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

Document Version	3.6.0
Document Status	Final
Part of Release	3.2
Revision	3

Document Change History			
Date	**Version**	**Changed by**	**Description**
28.02.2014	3.6.0	AUTOSAR Release Management	• Introduced new constraints • Improved documentation • Improved the upstream mapping • Miscellaneous fixes and improvements in the metamodel

26.03.2012	3.5.0	AUTOSAR Administration	• Explained the general approach of bit counting • Made Nm Attributes optional • Defined usage of "rxIdentifierRange" and "identifier" in CanFrameTriggering • Made Mappings and SoftwareComposition in System optional • Made "transferProperty" of "ISignalToIPduMapping" optional • Made dynamicPart in MultiplexedPdu optional • Added Constraints 3020, 3024, 3025, 3034, 3035, 3039, 3040, 3041, 3042, 3043, 3044 • Added LinSlaveConfig class to the LinMaster • Clarified relationship between CommConnectorPorts and Triggering elements

31.03.2011	3.4.0	AUTOSAR Administration	• Added missing FlexRayTp, LinTp and CanTp configuration parameters. • Added possibility to have ports on top level software composition in ECU Extract. • Support of new COM transfer properties • Implemented the network management user data concept • Implemented support for communication with Complex Device Drivers • Added ComProcessingGw and ComProcessingRx attributes to EcuInstance • Support for EndToEndProtection • Added support for mode dependent routing (PduRIPduGroup) • Added element RxIdentifierRange to CanFrameTriggering • Feature support of FlexRay 3.0 Hardware • Support for FlexRay ISO 10681-2 TP • Support for treating byte arrays with primitive type mapping • Support for Partial Networking

16.09.2010	3.3.0	AUTOSAR Administration	• Clarified semantics of Transfer Property for signal groups • Clarified semantics of ByteOrder attributes • Updated upstream template mapping of GdMaxMicrotick • Added the new transfer property TriggeredOnChange to ComTransferProperty • Added missing FlexRayNm and CanNm parameters • Clarified the usage of EcuPorts in Ecu Extract • Made Flexray channel specific attributes optional
22.10.2009	3.2.0	AUTOSAR Administration	• Clarified semantics of references to "ComIPduGroup" • Added TransferProperty attribute to ISignalToIPduMapping element. • Added extension that allows the specification of ranges for CAN Communication Controller Timing attributes • Adapted IPdu Multiplexer model to allow the segmentation of the static and dynamic part. • Added LinErrorResponse settings • Added version number attributes to the System class • Added relationships between ISignalTriggering, IPduTriggering, FrameTriggering • Added support for low-level routing of NPdu's • Updated description and model of the "EventControlledTiming" • Modeling of Priorities in Lin Substitution Frames • Added CanNm Id Range attributes to CanCluster

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.
01.07.2008	3.0.4	AUTOSAR Administration	• Adapt for Release 3.1
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. • Legal disclaimer revised • Release Notes added • "Advice for users" revised • "Revision Information" added
09.05.05	1.0.0	AUTOSAR Administration	Initial Release

Special Note:

The alignment between the System Template and the ECU Configuration Parameters is not formalized and finished at this time. Generation of the ECU extract is not verified.

Advice for users

AUTOSAR Specification Documents 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 Specification Documents for illustration purposes only, and they themselves are not part of the AUTOSAR Standard. Neither their presence in such Specification Documents, 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

References

[1] Template UML Profile and Modeling Guide
AUTOSAR_TemplateModelingGuide.pdf

[2] Model Persistence Rules for XML
AUTOSAR_ModelPersistenceRulesXML.pdf

[3] Methodology
AUTOSAR_Methodology.pdf

[4] AUTOSAR Template Modeling Patterns
AUTOSAR_TemplateModelingPatterns.pdf

[5] Software Component Template
AUTOSAR_SoftwareComponentTemplate.pdf

[6] Design Specification for the ECU Resource Template
AUTOSAR_ResourceTemplateECU.pdf

[7] Fibex - Field Bus Exchange Format, Version 2.0

[8] LIN Specification Package, Version 2.1

[9] CAN specifications

[10] MOST Specification, Version 2.5

[11] FlexRay Protocol Specification

[12] byteflight specifications

[13] Requirements on Basic Software: Layered Software Architecture
AUTOSAR_LayeredSoftwareArchitecture.pdf

[14] Specification of LIN Interface
AUTOSAR_SWS_LIN_Interface.pdf

[15] AUTOSAR RTE Software Specification
AUTOSAR_SWS_RTE.pdf

[16] Specification of the BSW Module Description Template
AUTOSAR_BSWMDTemplate.pdf

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

[18] Specification of Communication
AUTOSAR_SWS_COM.pdf

[19] Specification of ECU Configuration
AUTOSAR_ECU_Configuration.pdf

1 Introduction

1.1 Abbreviations

Abbreviation	*Meaning*
CAN	Controller Area Network
CAS	Collision Avoidance Symbol
CC	Communication Controller
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
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
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 "`Template UML Profile and Modeling Guide`" [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.

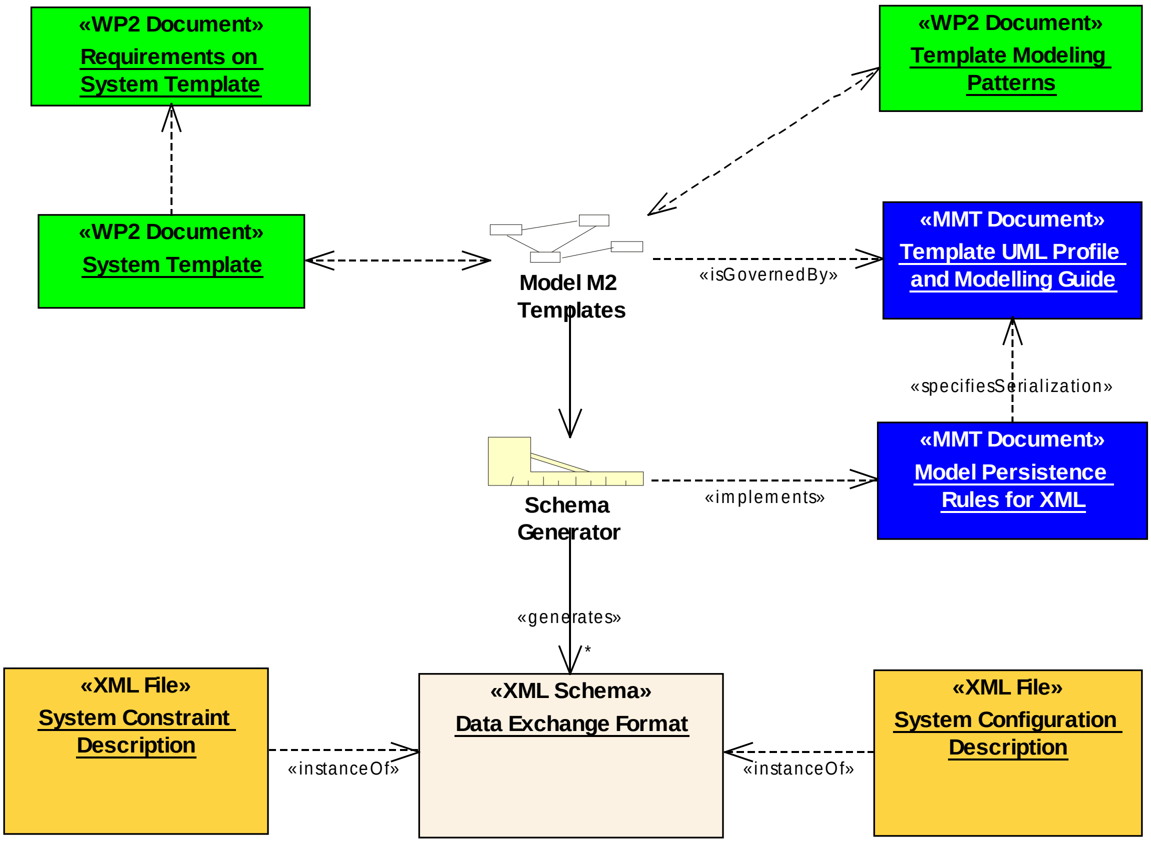

Figure 1.1: Methodology to define templates in AUTOSAR

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 `Template UML Profile and Modeling Guide` [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 "`Template Modeling Patterns`" are represented as predefined Classes in the meta-model which are incorporated in the generated schema. Examples for such patterns are the "common attributes" which are added to each generated class even if not explicitly inherited in the meta-model.

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.

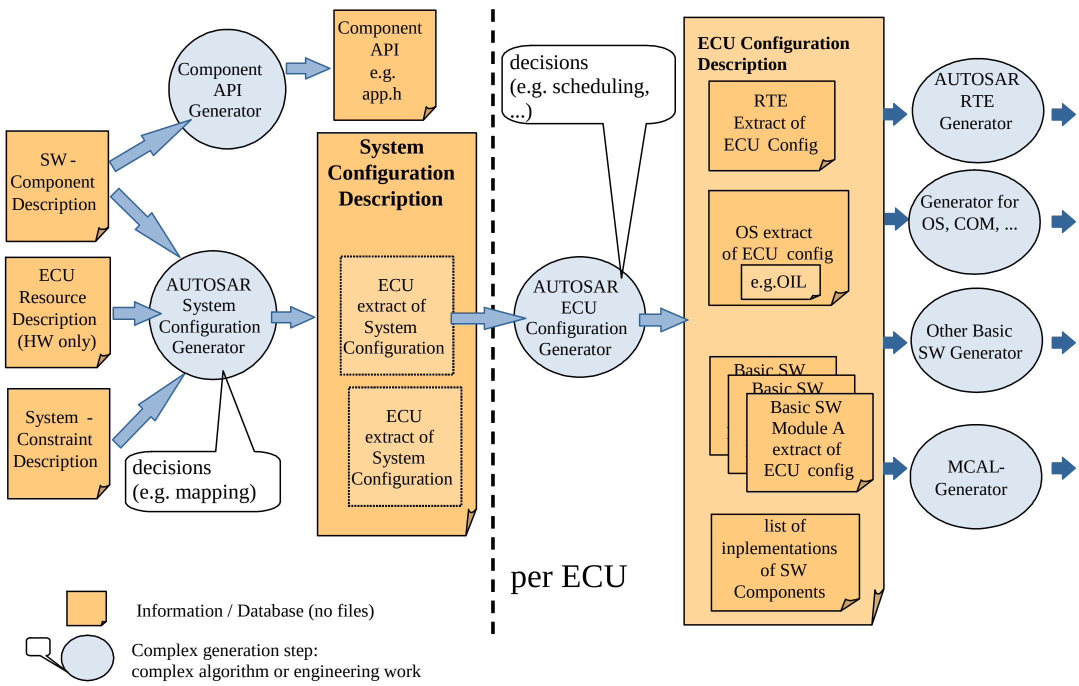

Figure 1.2: AUTOSAR Methodology

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.

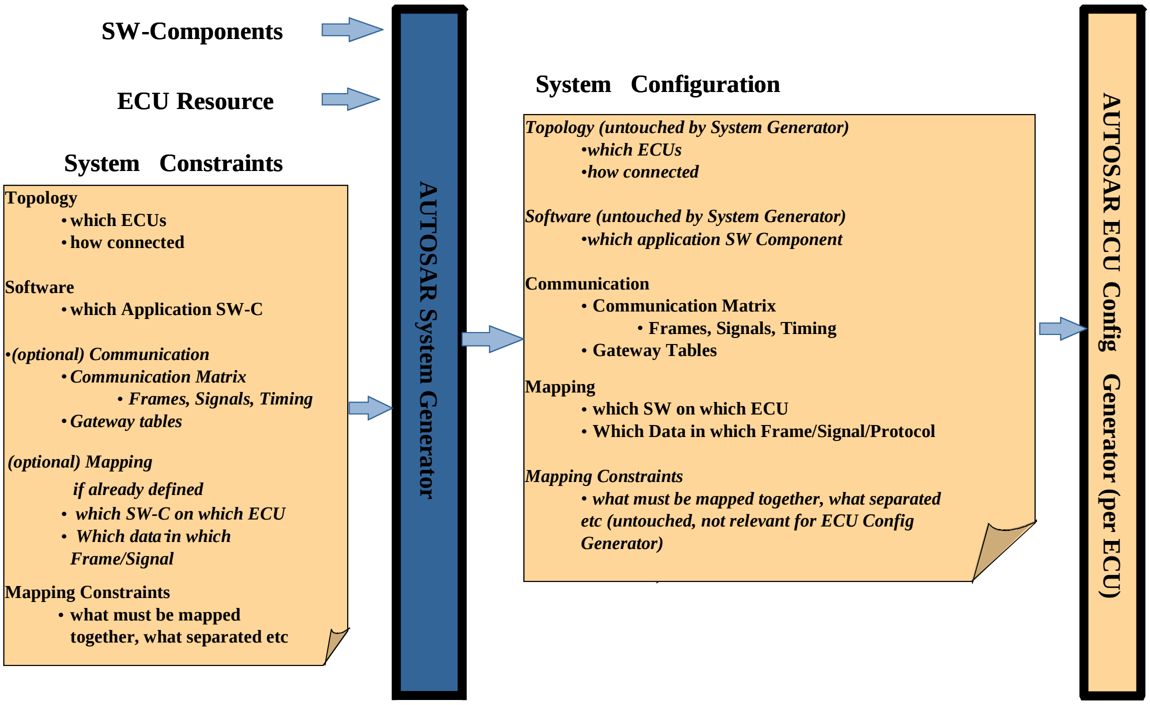

Figure 1.3: Scope of System Constraint Description and System Configuration Description

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 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.

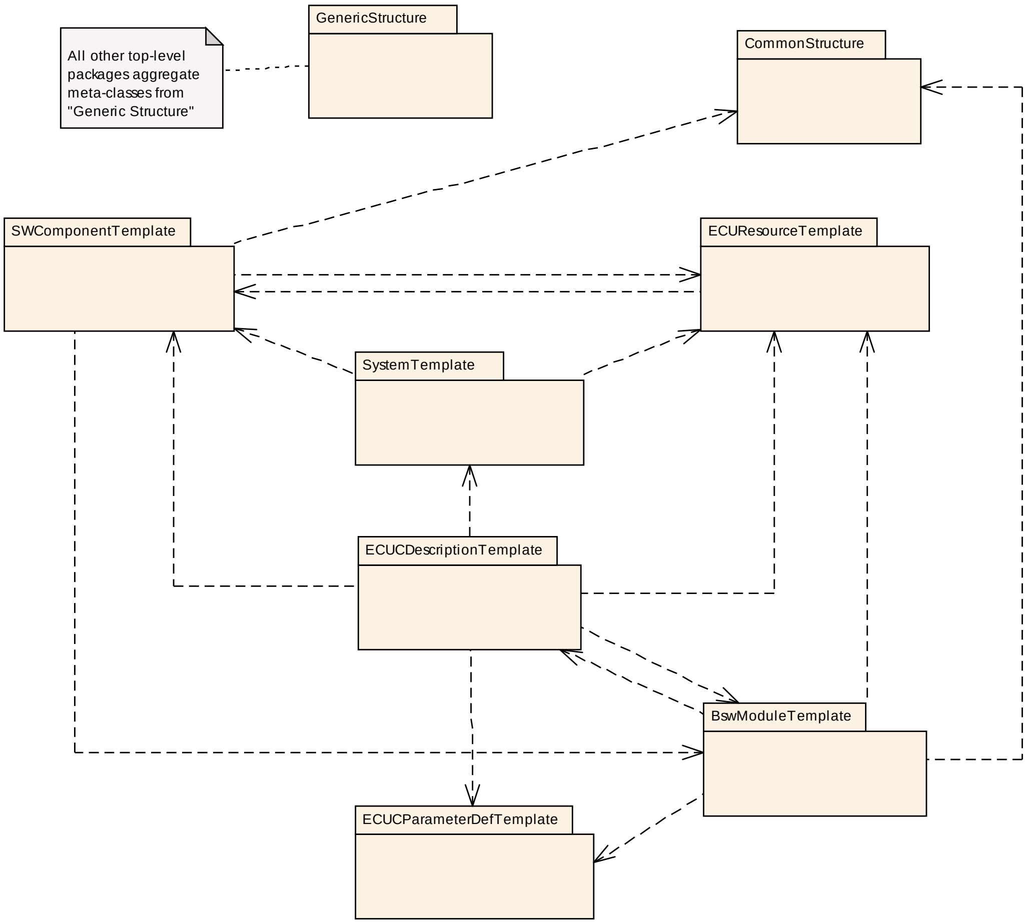

Figure 1.4: AUTOSAR Package Overview

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` [4], `SWComponentTemplate` [5] and `ECUResourceTemplate` [6].

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 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. ComponentType, PortInterface, AssemblyConnectorPrototype and DelegationConnectorPrototype. 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.

The package Generic Structure contains template independent definitions, e.g. the fact that template elements have unique identifiers. Furthermore, all templates need to follow the generic structure introduced in this part.

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].

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

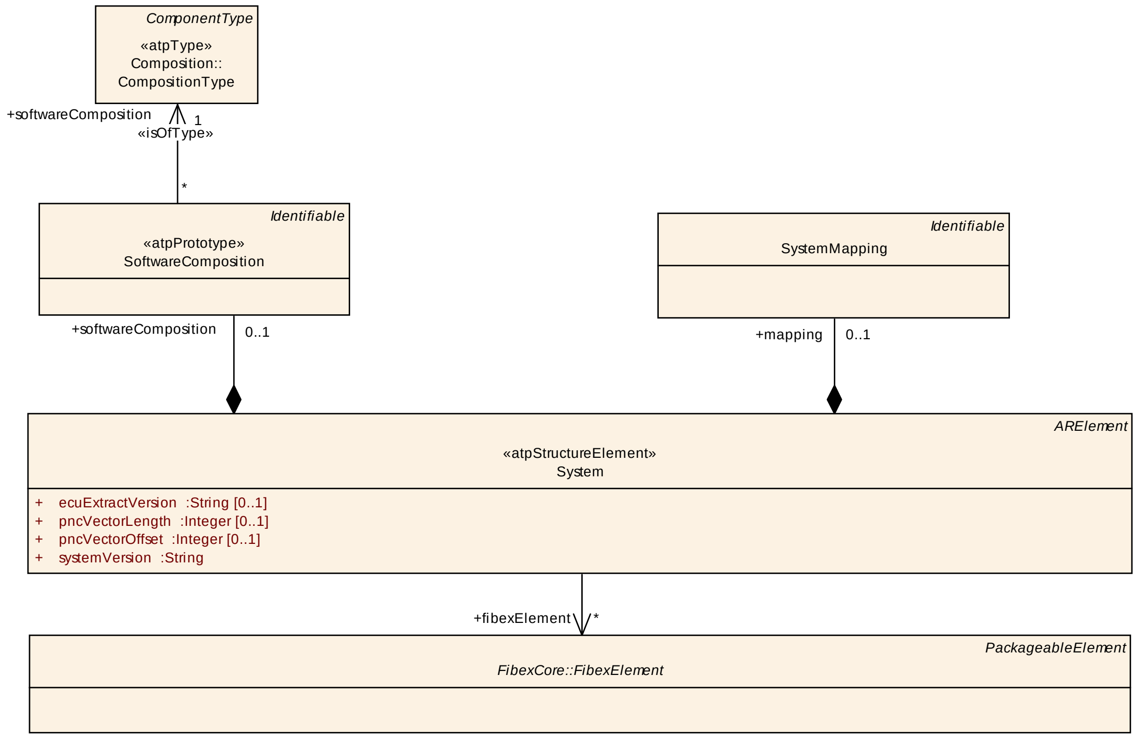

Figure 1.5: System Template Overview

[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].

System has relationships to all elements that define a system constraint description or system configuration description. It aggregates the SystemMapping and SoftwareComposition 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 SoftwareComposition element contains a reference to the top level software composition.

The System class contains a reference to FibexElements. 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 the next chapter.

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.			
Base	ARElement,ARObject,Identifiable,PackageableElement			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
ecuExtract Version	String	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.
mapping	SystemMapping	0..1	aggr	Aggregation of all mapping aspects (mapping of SW components to ECUs, mapping of data elements to signals, and mapping constraints).
pncVector Length	Integer	0..1	attr	Length of the partial networking request release information vector (in bytes).
pncVector Offset	Integer	0..1	attr	absolute offset (with respect to the NM-PDU) of the partial networking request release information vector that is defined in bytes as an index starting with 0.
softwareC omposition	SoftwareCompo sition	0..1	aggr	Aggregation of the top-level software composition, containing all software components in the System in a hierarchical structure.
systemVer sion	String	1	attr	Version number of the System Description.

Table 1.1: System

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. The attribute is marked as optional (multiplicity is 0..1). The reference is mandatory (lower multiplicity is 1).

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	aggrega	Attribute description
Role name	referenced class	1..*	referenc	Reference description

Table 1.2: Example of a class table

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 Template Modelling Guide [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 B.

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], byteflight [12]. 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 harmonisation 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.

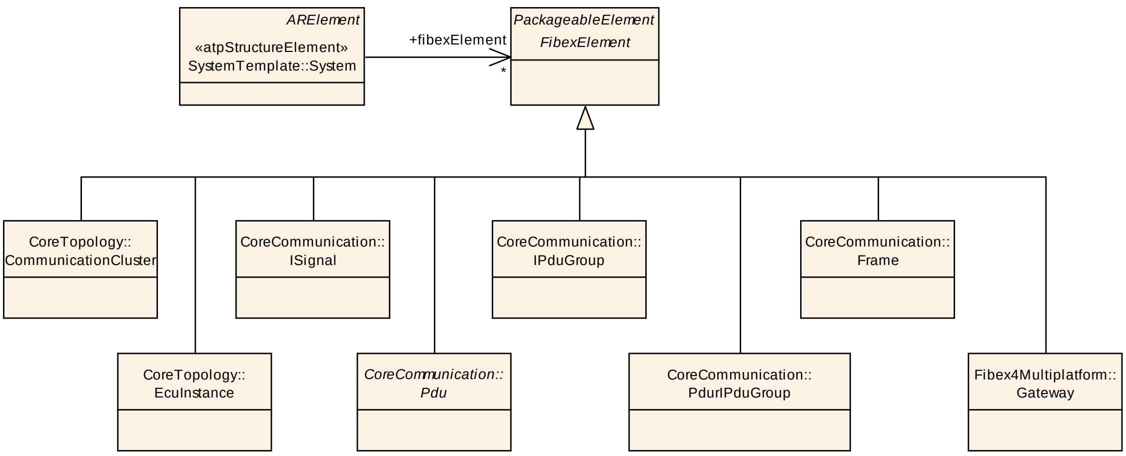

Figure 1.6: Fibex Elements

1.6 Document Conventions

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

1.7 Requirements Traceability

Requirement	Description	Satisfied by
SYSCT0001 Mixed Systems	The System Template has to cover resource requests of the basic SW and the RTE.	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)
SYSCT0002 Basic Software Resources and RTE Resources	System constraints, which arise through usage of mixed systems, must be treated by System Template.	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)
SYSCT0004 Variant handling	The System Template has to support variant handling.	not covered.
SYSCT0005 Timing requirements	The System Template has to describe timing requirements. Such timing requirements can be applied on frames, on signal paths, on single SW-C or on SW-C execution chains (including more than one ECU).	Frame Timing (chapter 5.6); Pdu Timing (chapter 5.10)
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. However, it doesn't describe the scheduling aspects nor the mapping of software components to individual microcontrollers residing in one ECU.	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)

Requirement	Description	Satisfied by
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 1 frame.	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 (chapter Communication 5)
SYSCT0015 Bus bandwidth	The System Template shall support bandwidth calculation as a constraint for the definition of the Communication Matrix.	chapter Topology (2); chapter Communication (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).	chapter Signal Path Constraint (4.2.2)
SYSCT0017 Mapping of signals to the same physical line	MThe 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)

Requirement	Description	Satisfied by
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.	FlexRayspecific 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 9)
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)

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.

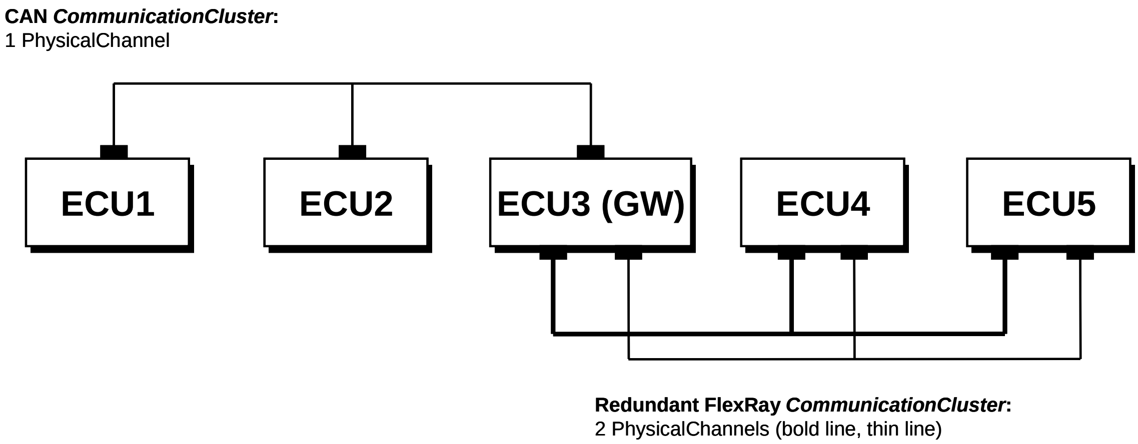

Figure 2.1: Example for a Communication Cluster within a physical network topology

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 SoftwareComposition 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.

2.1.1 ECU Instance

ECUInstance describes the presence of an Electronic Control Unit 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 [6]: 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.			
Base	ARObject,FibexElement,Identifiable,PackageableElement			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
associated IPduGroup	IPduGroup	*	ref	With this reference it is possible to identify which IPduGroups are applicable for which CommunicationConnector/ ECU.
associated PdurIPduG roup	PdurIPduGroup	*	ref	With this reference it is possible to identify which PduR IPdu Groups are applicable for which CommunicationConnector/ ECU.
comConfig urationId	Integer	0..1	attr	This ID is returned by a call to Com_GetConfigurationId()
comProces singPeriod	Float	0..1	attr	The COM scheduling time is used in order to be able to calculate the worst case bus timing. The processing period shall be specified AUTOSAR conform in seconds.
comProces singPeriod Gw	Float	0..1	attr	Optional signal Routing processing period of the COM scheduling in order to be able to calculate the worst case bus timing. Only applicable if a different processing period for Tx and Routing shall be respected. If not present the "comProcessingPeriod" attribute shall be used for Tx and Gateway operation. The processing period shall be specified AUTOSAR conform in seconds.
comProces singPeriod Rx	Float	0..1	attr	Optional Rx processing period of the COM scheduling in order to be able to calculate the worst case bus timing. Only applicable if a different processing period for Tx and Rx shall be respected. If not present the "comProcessingPeriod" attribute shall be used for Tx and Rx operation. The processing period shall be specified AUTOSAR conform in seconds.
commCont roller	Communication Controller	1..*	aggr	CommunicationControllers of the ECU.
connector	Communication Connector	*	aggr	All channels controlled by a single controller.
diagnostic Address	Integer	0..1	attr	An ECU specific ID for responses of diagnostic routines.
pduRConfi gurationId	Integer	0..1	attr	unique PDURconfiguration identifier

Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
responseAddress	Integer	*	attr	This attribute is obsolete and will be removed in future. **Tags:** atp.Status=obsolete
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.
wakeUpOverBusSupported	Boolean	1	attr	Driver support for wakeup over Bus.

Table 2.1: EcuInstance

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	**CommunicationController**			
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.			
Base	ARObject,Identifiable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
wakeUpByControllerSupported	Boolean	1	attr	Defines whether the ECU shall be woken up by this CommunicationController. TRUE: wake up is possible FALSE: wake up is not supported Note: If wakeUpByControllerSupported is set to TRUE the feature shall be supported by both hardware and basic software.

Table 2.2: 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**			
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			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
channel	PhysicalChannel	1	ref	Reference to the channel to which the ECU is connected.
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.
nmAddress	Integer	0..1	attr	An ECUs NM address on the referenced channel.
nmEnabled	Boolean	0..1	attr	Specifies whether Network Management shall be enabled for this CommunicationConnector. If set to true the NM-Attributes of CommunicationCluster and CommunicationConnector shall be provided.

Attribute	Datatype	Mul.	Kind	Note
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 2.3: 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 2.4: PncGatewayTypeEnum

Note: Use-case for the relation of several `CommunicationConnector`s 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.

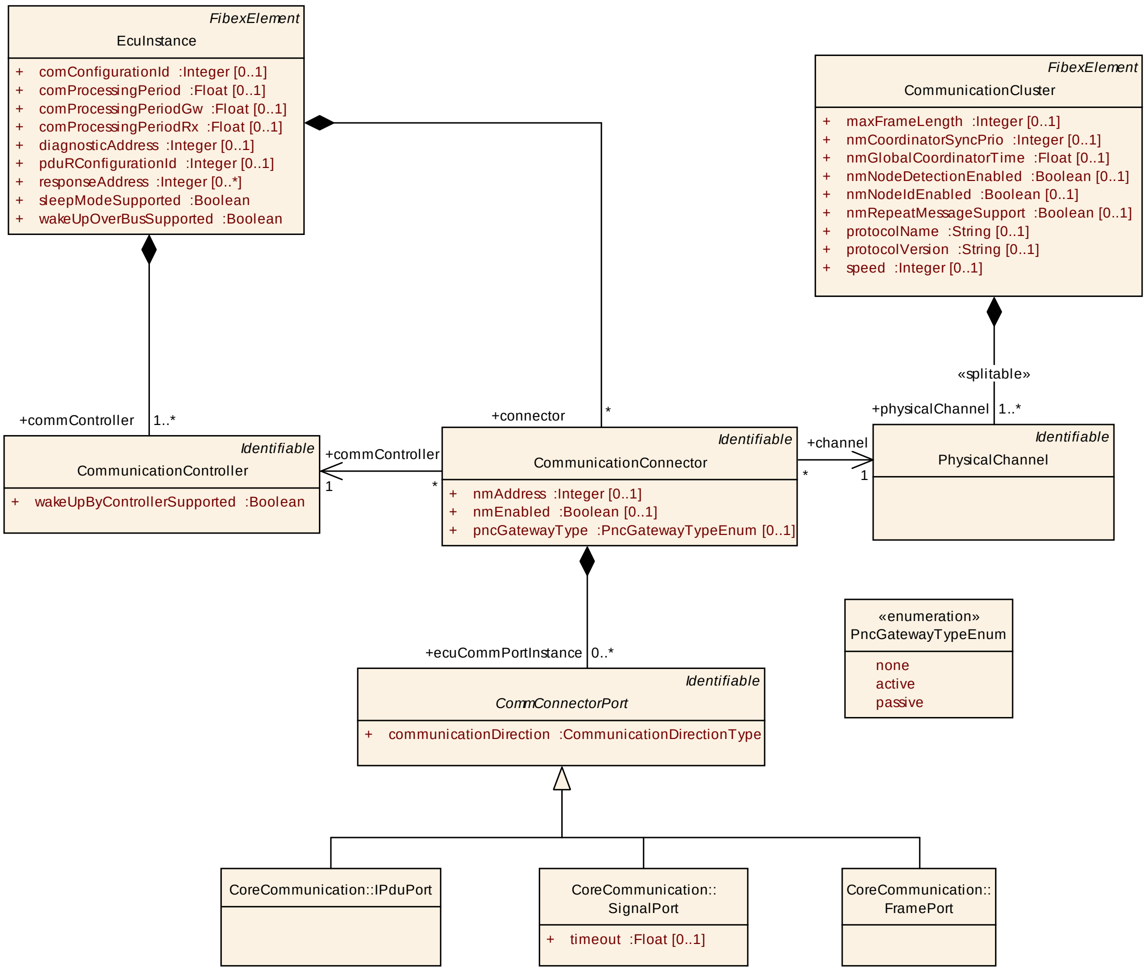

Figure 2.2: Topology elements (Topology)

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	**CommunicationCluster**			
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.			
Base	ARObject,FibexElement,Identifiable,PackageableElement			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
maxFrameLength	Integer	0..1	attr	Maximal supported length in bytes for frames in this cluster.
nmCoordinatorSyncPrio	Integer	0..1	attr	Priority of the Nm coordinator.
nmGlobalCoordinatorTime	Float	0..1	attr	This attribute defines the maximum shutdown time of a connected and coordinated NM-Cluster.
nmNodeDetectionEnabled	Boolean	0..1	attr	Enable/disable the node detection functionality.
nmNodeIdEnabled	Boolean	0..1	attr	Enable/disable the source node identifier.
nmRepeatMessageSupport	Boolean	0..1	attr	switch for enabling support for repeat message
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
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 bits per second

Table 2.5: 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**			
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			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
frameTriggerings	FrameTriggering	*	aggr	One frame triggering is defined for exactly one channel. Channels may have assigned an arbitrary number of frame triggerings. **Stereotypes:** atpSplitable
iPduTriggering	IPduTriggering	*	aggr	One I-Pdu triggering is defined for exactly one channel. Channels may have assigned an arbitrary number of I-Pdu triggerings. **Stereotypes:** atpSplitable
iSignalTriggering	ISignalTriggering	*	aggr	One ISignalTriggering is defined for exactly one channel. Channels may have assigned an arbitrary number of ISignaltriggerings. **Stereotypes:** atpSplitable
tpAddress	TpAddress	*	aggr	Collection of TpAddresses.
tpChannel	TpChannel	*	aggr	Optional configuration of Tranport Protocol channels.
tpNode	TpNode	*	aggr	Senders and receivers of TP messages.

Table 2.6: PhysicalChannel

2.3 Specialized Attributes of the Topology Entities

According to their characteristic features, different communication standards like FlexRay, CAN, LIN 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` and `LinCluster`.

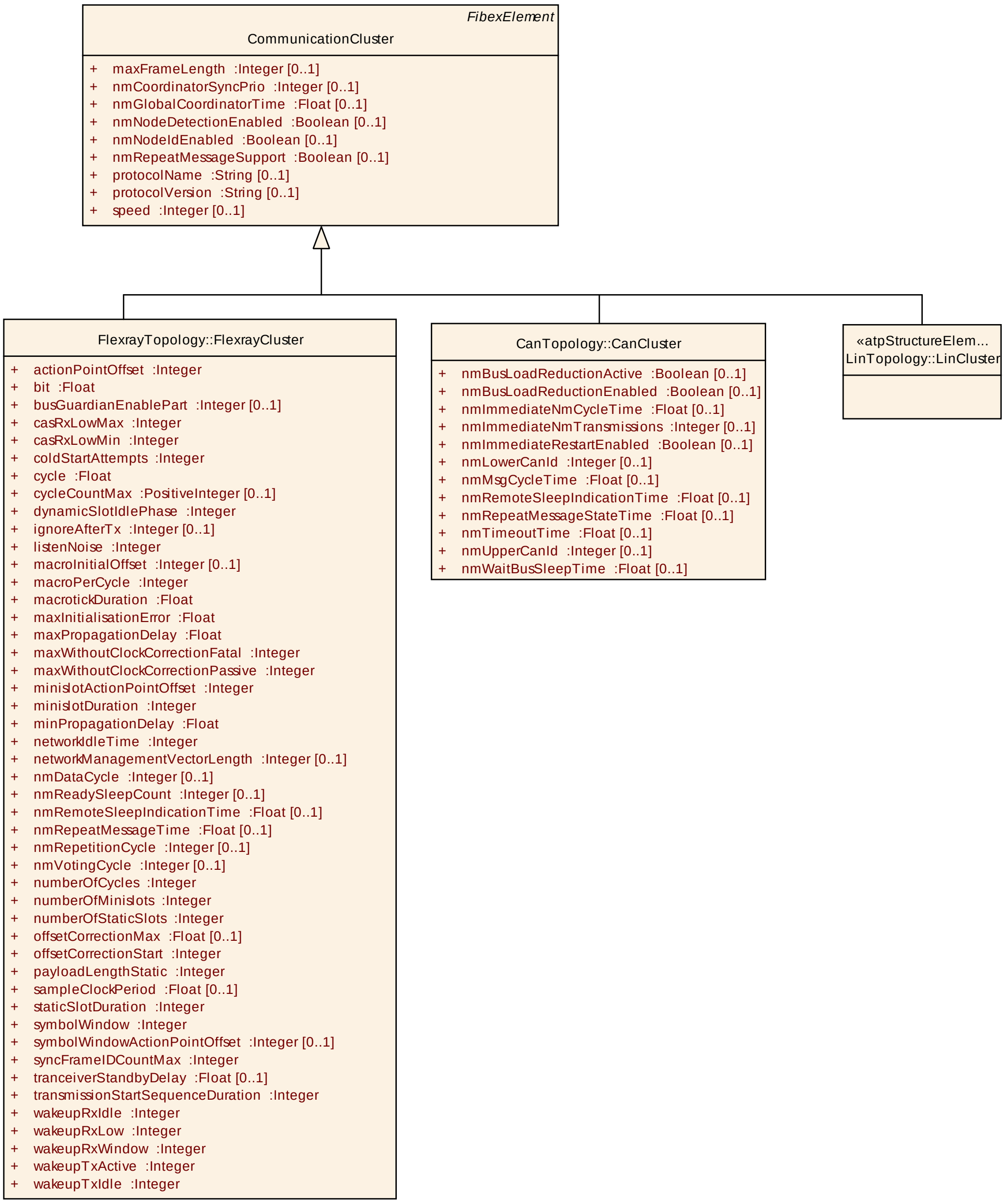

Figure 2.3: Specialized `CommunicationCluster` attributes (TopologyAttributeRefinement)

2.3.1 CAN

Modeling of the Can bus is supported in the System Template by the means of two specialized meta-model classes, `CANCluster` and `CanCommunicationController` (Figure 2.4).

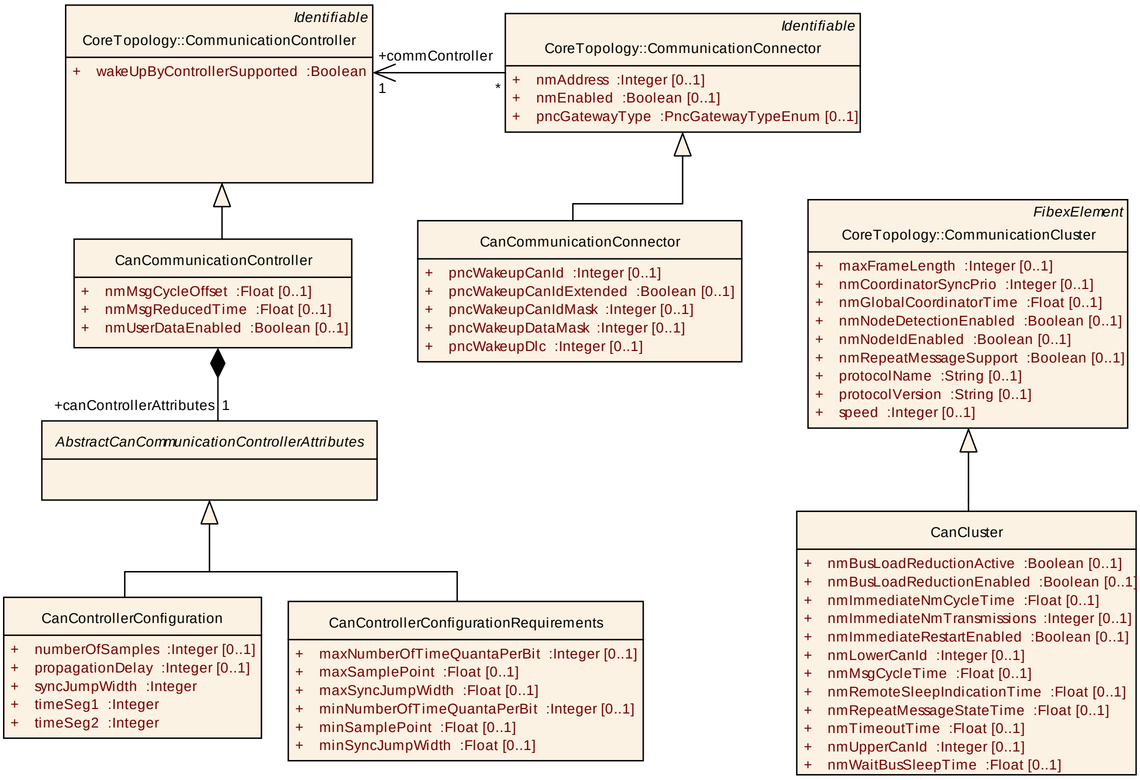

Figure 2.4: Can bus elements (Fibex4Can_Topology)

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.

Class	**CanCluster**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	CAN specific attributes **Tags:** xml.namePlural=CAN-BUSES			
Base	ARObject,CommunicationCluster,FibexElement,Identifiable,PackageableElement			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
nmBusLoadReductionActive	Boolean	0..1	attr	It determines if bus load reduction for the respective NM channel is active or not. True: active False: inactive
nmBusLoadReductionEnabled	Boolean	0..1	attr	switch for enabling busload reduction support.
nmImmediateNmCycleTime	Float	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	Integer	0..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 immediate NM PDUs is defined by nmImmediateNmCycleTime.
nmImmediateRestartEnabled	Boolean	0..1	attr	Enables the asynchronous transmission of a CanNm PDU upon bus-communication request in Prepare-Bus-Sleep mode.
nmLowerCanId	Integer	0..1	attr	This attribute can be used together with the nmUpperCanId attribute to define a range of CanIds. Can Frames which will arrive in the given Id Range will be handled as Nm Pdus. 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.
nmMsgCycleTime	Float	0..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.
nmRemoteSleepIndicationTime	Float	0..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.
nmRepeatMessageStateTime	Float	0..1	attr	It defines how long the NM shall stay in the Repeat Message State (in seconds)
nmTimeoutTime	Float	0..1	attr	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.

Attribute	Datatype	Mul.	Kind	Note
nmUpperCanId	Integer	0..1	attr	This attribute can be used together with the nmLowerCanId attribute to define a range of CanIds. Can Frames which will arrive in the given Id Range will be handled as Nm Pdus. 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.
nmWaitBusSleepTime	Float	0..1	attr	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.

Table 2.7: CanCluster

2.3.1.2 CAN Communication Controller

`CanCommunicationController` is a specialization of the `CommunicationController` class. It contains the specific CAN controller attributes needed for configuring the Can stack in an ECU connected to a certain CAN cluster. 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	**CanCommunicationController**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	CAN bus specific communication port attributes.			
Base	ARObject,CommunicationController,Identifiable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
canControllerAttributes	AbstractCanCommunicationControllerAttributes	1	aggr	CAN Bit Timing configuration
nmMsgCycleOffset	Float	0..1	attr	Node specific time offset in the periodic transmission node. It determines the start delay of the transmission. Specified in seconds.
nmMsgReducedTime	Float	0..1	attr	Node specific bus cycle time in the periodic transmission mode with bus load reduction. Specified in seconds.
nmUserDataEnabled	Boolean	0..1	attr	Switch for enabling user data support.

Table 2.8: CanCommunicationController

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 2.9: 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**
numberOfSamples	Integer	0..1	attr	Number of samples. Possible values are 1 or 3.
propagationDelay	Integer	0..1	attr	The propagation time segment in quanta.
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.

Attribute	Datatype	Mul.	Kind	Note
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 2.10: 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 2.11: CanControllerConfigurationRequirements

2.3.1.3 CAN Communication Connector

`CanCommunicationConnector` adds the CAN specific attributes to the `CommunicationConnector`.

Class	CanCommunicationConnector			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	CAN bus specific communication connector attributes.			
Base	ARObject,CommunicationConnector,Identifiable			
Attribute	Datatype	Mul.	Kind	Note
pncWakeupCanId	Integer	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. true: extended Can identifier is used false: standard Can identifier is used
pncWakeupCanIdMask	Integer	0..1	attr	Bit mask for CAN Identifier used to configure the CAN Transceiver for partial network wakeup.
pncWakeupDataMask	Integer	0..1	attr	Bit mask for CAN Payload used to configure the CAN Transceiver for partial network wakeup. Note that this data mask is calculated over the whole payload (8 Byte) of the NmPdu ignoring the leading bytes which do not contain pncVector information. The number of leading bytes which shall be ignored is equivalent to the value of System.pncVectorOffset.
pncWakeupDlc	Integer	0..1	attr	Data Length of the remote data frame used to configure the CAN Transceiver for partial network wakeup in Bytes.

Table 2.12: CanCommunicationConnector

2.3.2 FlexRay

Modelling of FlexRay clusters is supported in the System Template by the means of four specialized meta-model classes, `FlexrayCluster`, `FlexrayCommunicationConnector`, `FlexrayCommunicationController` and `FlexrayPhysicalChannel`. (Figure 2.5).

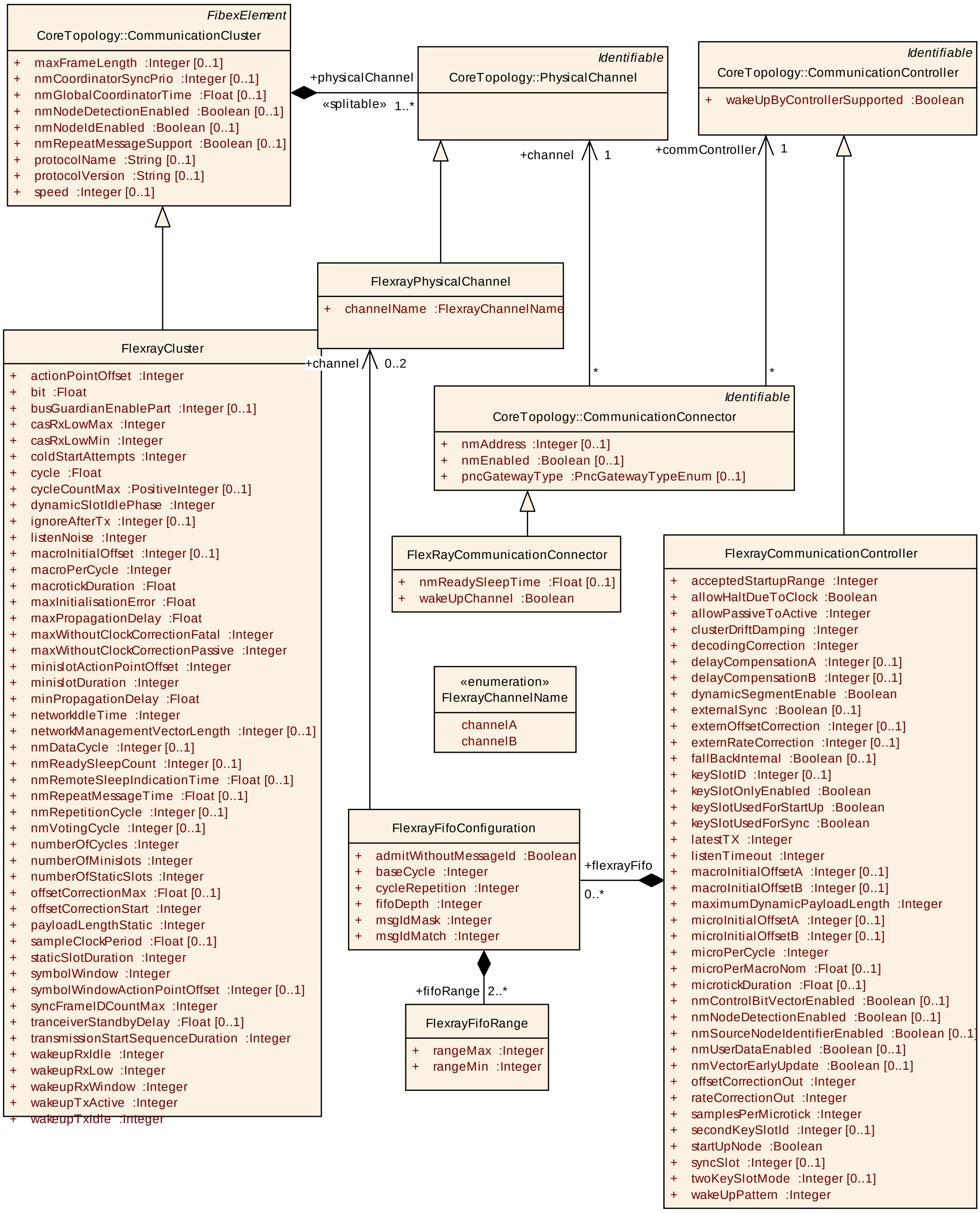

Figure 2.5: FlexRay cluster elements (Fibex4FlexRay_Topology)

2.3.2.1 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	**FlexrayCluster**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::FlexrayTopology			
Note	FlexRay specific attributes to the physicalCluster			
Base	ARObject,CommunicationCluster,FibexElement,Identifiable,PackageableElement			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
actionPointOffset	Integer	1	attr	The offset of the action point in networks
bit	Float	1	attr	Nominal bit time (= 1 / fx:SPEED). gdBit = cSamplesPerBit * gdSampleClockPeriod. Unit: seconds (gdBit)
busGuardianEnablePart	Integer	0..1	attr	Bus Guardian Inter Slot Gap (ISG) part that follows a guarded schedule element. Unit macroticks
casRxLowMax	Integer	1	attr	Upper limit of the Collision Avoidance Symbol (CAS) acceptance window. Unit:bitDuration
casRxLowMin	Integer	1	attr	Lower limit of the Collision Avoidance Symbol (CAS) acceptance window. Unit:bitDuration
coldStartAttempts	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	Float	1	attr	Length of the cycle. Unit: seconds
cycleCountMax	PositiveInteger	0..1	attr	Maximum cycle counter value in a given cluster. Remark: Set to 63 for FlexRay Protocol 2.1 Rev. A compliance.
dynamicSlotIdlePhase	Integer	1	attr	The duration of the dynamic slot idle phase in minislots.
ignoreAfterTx	Integer	0..1	attr	Duration for which the bitstrobing is paused after transmission [gdBit].
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
macroInitialOffset	Integer	0..1	attr	Please note that this attribute is deprecated and will be removed in future. Use FlexrayCommunicationController.macroInitialOffsetA resp. FlexrayCommunicationController.macroInitialOffsetB instead. Number of macroticks which describe the distance between the static slot boundary and the closed macrotick boundary of the secondary time reference point using the initial configured macrotick length. **Tags:** atp.Status=obsolete

Attribute	Datatype	Mul.	Kind	Note
macroPerCycle	Integer	1	attr	The number of macroticks in a communication cycle
macrotickDuration	Float	1	attr	Duration of the cluster wide nominal macrotick, expressed in seconds
maxInitialisationError	Float	1	attr	The maximum error that a node may have after initialization. Unit: seconds
maxPropagationDelay	Float	1	attr	Maximum propagation delay of a Cluster (in seconds).
maxWithoutClockCorrectionFatal	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.
maxWithoutClockCorrectionPassive	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.
minPropagationDelay	Float	1	attr	Minimum propagation delay of a Cluster (in seconds).
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	0..1	attr	Length of the Network Management vector on a cluster. Unit: Bytes
nmDataCycle	Integer	0..1	attr	Number of FlexRay Communication Cycles needed to transmit the Nm Data PDUs of all FlexRay Nm Ecus of this FlexRayNmCluster.
nmReadySleepCount	Integer	0..1	attr	The value of this parameter influences the shutdown behavior of the FlexRay NM. FrNm switches to bus sleep mode at the end of the nmReadySleepCount +1 repetition cycle without any NM vote. If this parameter is available in the system template the parameter nmReadySleepTime shall not be available.
nmRemoteSleepIndicationTime	Float	0..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.

Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
nmRepeatMessageTime	Float	0..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	0..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	0..1	attr	Number of FlexRay CommunicationCycles needed to transmit the Nm vote of Pdus of all FlexRay NmEcus of this FlexRayNmCluster.
numberOfCycles	Integer	1	attr	Total number of cycles until a temporal transmission pattern is repeated. The CycleCounter of an AbsolutelyScheduledTiming is evaluated against this parameter.
numberOfMinislots	Integer	1	attr	number of Minislots in the dynamic segment.
numberOfStaticSlots	Integer	1	attr	The number of static slots in the static segment.
offsetCorrectionMax	Float	0..1	attr	Cluster global magnitude of the maximum permissible offset correction value Unit: seconds (gOffsetCorrectionMax)
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.
sampleClockPeriod	Float	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	0..1	attr	Number of macroticks the action point offset is from the beginning of the symbol window [Macroticks].
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.
tranceiverStandbyDelay	Float	0..1	attr	The duration of timer t_TrcvStdbyDelay in seconds. The granularity of this parameter shall be restricted to full FlexRay cycles (cycle). The transceiver status setting to STANDBY shall be delayed by this value. Not specifying a value or a value of 0 shall imply that the timer is not used.

Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
transmissionStartSequenceDuration	Integer	1	attr	Number of bits in the Transmission Start Sequence [gdBits].
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.
wakeupRxLow	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.
wakeupRxWindow	Integer	1	attr	The size of the window used to detect wakeups [gdBit]. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolRxWindow.
wakeupTxActive	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. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolTxLow.
wakeupTxIdle	Integer	1	attr	Number of bits used by the node to transmit the 'idle' part of a wakeup symbol. Unit: gDbit. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolTxIdle.

Table 2.13: FlexrayCluster

2.3.2.2 FlexRay Communication Controller

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

Class	**FlexrayCommunicationController**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::FlexrayTopology			
Note	FlexRay bus specific communication port attributes.			
Base	ARObject,CommunicationController,Identifiable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
acceptedStartupRange	Integer	1	attr	Expanded range of measured clock deviation allowed for startup frames during integration. Unit:microtick

Attribute	Datatype	Mul.	Kind	Note
allowHaltDueToClock	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).
allowPassiveToActive	Integer	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
clusterDriftDamping	Integer	1	attr	The cluster drift damping factor used in clock synchronization rate correction in microticks
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.
dynamicSegmentEnable	Boolean	1	attr	Boolean flag that configures the Bus Guardian Schedule Monitoring Service to expect transmissions within the dynamic segment.
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.

Attribute	Datatype	Mul.	Kind	Note
keySlotID	Integer	0..1	attr	ID of the slot used to transmit the startup frame, sync frame, or designated single slot frame. If the attributes keySlotUsedForStartUp, keySlotUsedForSync, or keySlotOnlyEnabled are set to true the key slot value is mandatory.
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	Upper limit for the start up listen timeout and wake up listen timeout. Unit: Microticks
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

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
microPerMacroNom	Float	0..1	attr	Please note that this attribute is deprecated and will be removed in future. Old description: Number of microticks per nominal macrotick that all implementations must support. **Tags:** atp.Status=obsolete
microtickDuration	Float	0..1	attr	Duration of a microtick. This attribute can be derived from samplePerMicrotick and gdSampleClockPeriod. Unit: seconds
nmControlBitVectorEnabled	Boolean	0..1	attr	Enables control bit vector support.
nmNodeDetectionEnabled	Boolean	0..1	attr	Enables the Request Repeat Message Request support. Only valid if nmNodeIdEnabled is set to true.
nmSourceNodeIdentifierEnabled	Boolean	0..1	attr	Switch for enabling SourceNodeIdentifier support.
nmUserDataEnabled	Boolean	0..1	attr	Switch for enabling user data support.
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)
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	1	attr	Number of samples per microtick
secondKeySlotId	Integer	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.
startUpNode	Boolean	1	attr	Indicates that the node is a startup node (startup frame configured; connected to gChannels)
syncSlot	Integer	0..1	attr	The number of the static slot in which a sync frame shall be sent, if a sync frame shall be sent

Attribute	Datatype	Mul.	Kind	Note
twoKeySlot Mode	Integer	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.
wakeUpPa ttern	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 2.14: FlexrayCommunicationController

2.3.2.3 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			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
nmReadySleepTime	Float	0..1	attr	The value of this parameter influences the shutdown behavior of the FlexRay NM. FrNm switches to bus sleep mode nmReadySleepTime seconds after the completion of the last repetition cycle containing a NM vote. If this parameter is available in the system template the parameter nmReadySleepCount shall not be available.
wakeUpChannel	Boolean	1	attr	Referenced channel used by the node to send a wakeup pattern. (pWakeupChannel)

Table 2.15: FlexRayCommunicationConnector

[constr_3508] Value of nmReadySleepTime ⌈ The nmReadySleepTime value shall be a multiple of cycle * nmRepetitionCycle. ⌋

2.3.2.4 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,PhysicalChannel			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
channelName	FlexrayChannelName	1	attr	Name of the channel (Channel A or Channel B).
pduPool	FlexrayIsoTpPduPool	*	aggr	Optional configuration of FlexRay TP Pdu Pools (shall only be used for Flexray ISO TP configuration).
tpConnectionControl	FlexrayIsoTpConnectionControl	*	aggr	Optional configuration of FlexRay ISO TP Connection Controls (shall only be used for Flexray ISO TP configuration).
tpEcus	FlexrayIsoTpEcu	*	aggr	Optional collection of TP Ecus with ECU specific Flexray ISO Tp configuration parameters.

Table 2.16: FlexrayPhysicalChannel

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

Table 2.17: FlexrayChannelName

2.3.3 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.

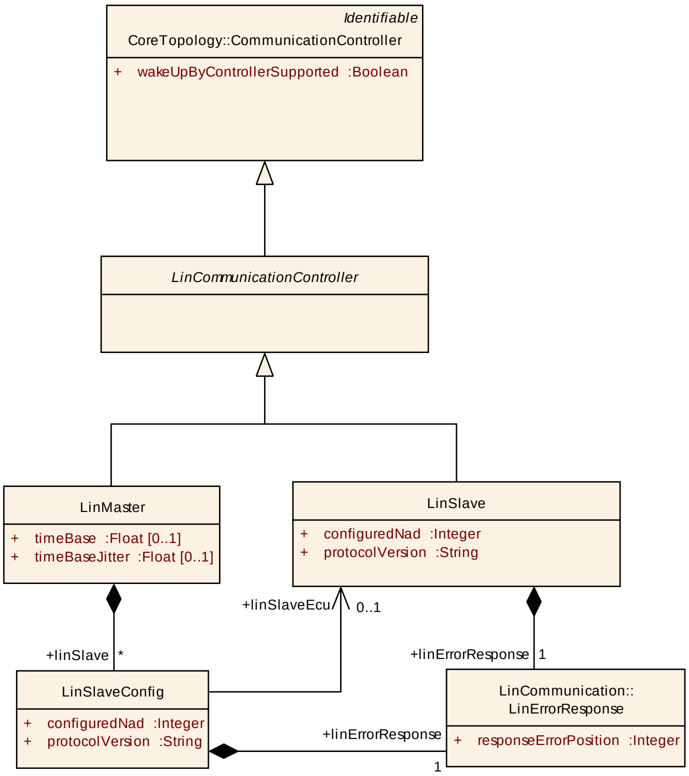

Figure 2.6: Specialized LINCommunicationController attributes (Fibex4Lin_Topology)

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 3.0 ([13], [14]).

2.3.3.1 LIN Cluster

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

Class	**LinCluster**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	LIN specific attributes			
Base	ARObject,CommunicationCluster,FibexElement,Identifiable,PackageableElement			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
scheduleTable	LinScheduleTable	*	aggr	Schedule tables organize the Timings of the frames for LIN.

Table 2.18: LinCluster

2.3.3.2 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	**LinCommunicationController (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	LIN bus specific communication port instance attributes.			
Base	ARObject,CommunicationController,Identifiable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Table 2.19: LinCommunicationController

2.3.3.3 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	**LinMaster**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	Describing the properties of the refering ecu as a LIN master.			
Base	ARObject,CommunicationController,Identifiable,LinCommunicationController			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
linSlave	LinSlaveConfig	*	aggr	LinSlaves that are handled by the LinMaster.
timeBase	Float	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.

Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
timeBaseJitter	Float	0..1	attr	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 2.20: LinMaster

Class	**LinSlaveConfig**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	Node attributes of LIN slaves that are handled by the LinMaster. In the System Description LIN slaves may be described as non AUTOSAR ECUs (linSlaveEcu reference). But in an Ecu Extract of the LinMaster the LinSlaveEcus will not be available. The information that is described here is necessary in the ECU Extract for the configuration of the LinMaster. The values of attributes of LinSlaveConfig and LinSlave shall be identical for each LinSlaveConfig that points to a LinSlave. Please note that this causes redundancy which is intended to support flexible development methodology.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
configuredNad	Integer	1	attr	To distinguish LIN slaves that are used twice or more within the same cluster.
linErrorResponse	LinErrorResponse	1	aggr	Each slave node shall publish one response error in one of its transmitted unconditional frames.
linSlaveEcu	LinSlave	0..1	ref	Reference to the LinSlaveEcu.
protocolVersion	String	1	attr	Version number for a communication protocol.

Table 2.21: LinSlaveConfig

[constr_3034] Values of `LinSlaveConfig` and `LinSlave` attributes ⌈ The values of attributes of `LinSlaveConfig` and `LinSlave` shall be identical for each `LinSlaveConfig` that points to a `LinSlave`. ⌋

Please note that this causes redundancy which is intended to support flexible development methodology.

[TPS_SYST_01046] ShortNames of `LinSlaveConfig` and `LinSlave` ⌈ The `shortName`s of a pair of `LinSlaveConfig` and `LinSlave` do not have to be identical. ⌋

2.3.3.4 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	**LinSlave**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	Describing the properties of the refering ecu as a LIN slave.			
Base	ARObject,CommunicationController,Identifiable,LinCommunicationController			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
configuredNad	Integer	1	attr	To distinguish LIN slaves that are used twice or more within the same cluster.
linErrorResponse	LinErrorResponse	1	aggr	Each slave node shall publish one response error in one of its transmitted unconditional frames.
protocolVersion	String	1	attr	Version number for a communication protocol.

Table 2.22: 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 2.23: LinErrorResponse

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 2.7). It can be specified which `ECU` hardware is realizing each given `ECUInstance`, providing the means for algorithms to map software components onto the systems `ECUInstance`. By specifying which hardware `ECUCommunicationPort` on a `CommunicationPeripheral` 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).

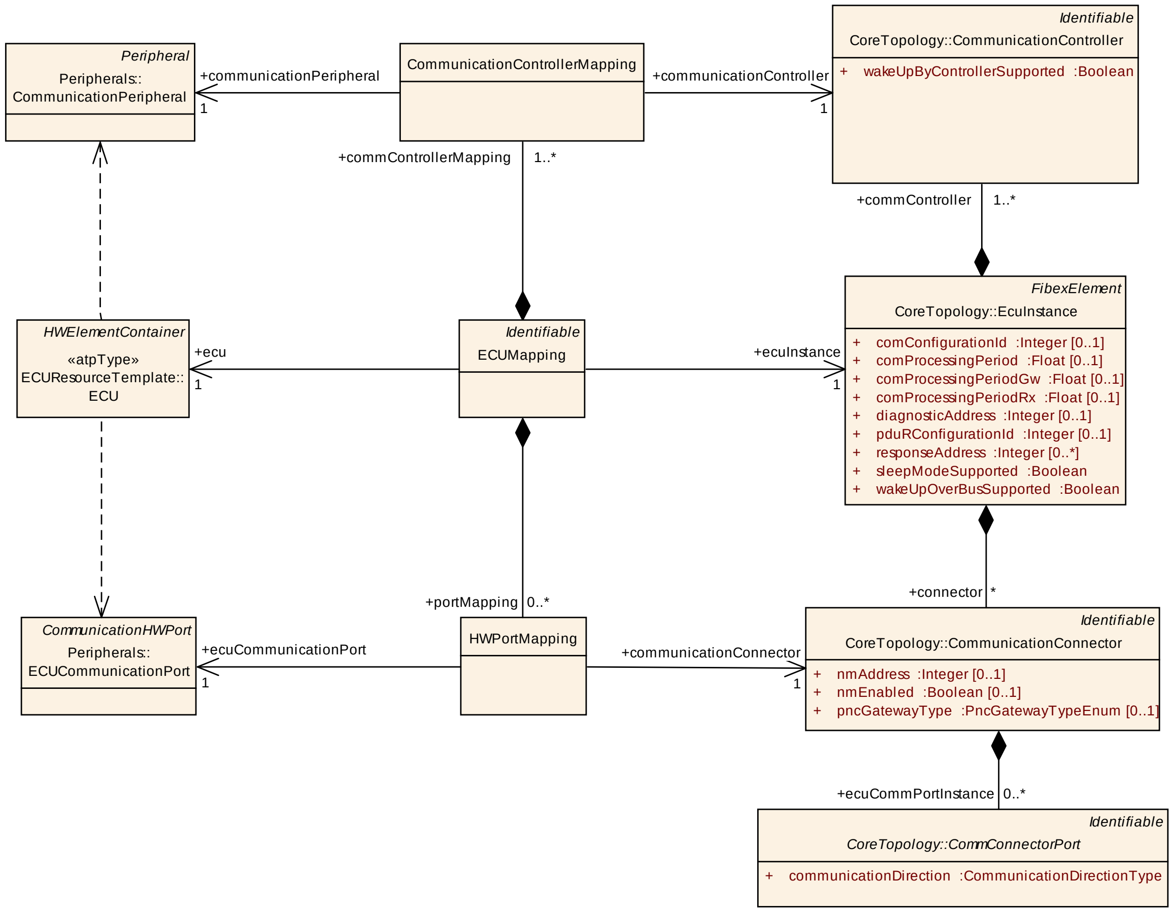

Figure 2.7: Mapping of topology description elements in the System Template onto hardware elements defined in the ECU Resource Template (ECUResourceMapping)

2.4.1 ECU Mapping

`ECUMapping` allows to assign an `ECU` hardware type to an `ECUInstance` used in a physical topology. `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`.

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			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
commControllerMapping	CommunicationControllerMapping	1..*	aggr	The ECUMapping contains the mapping of all CommunicationControllers of the ECU.
ecu	ECU	1	ref	Reference to the Ecu description in the ECU Resource Template
ecuInstance	EcuInstance	1	ref	Reference to the EcuInstance in the System Template
portMapping	HWPortMapping	*	aggr	The ECUMapping contains the mapping of all HW Ports of the ECU.

Table 2.24: ECUMapping

2.4.2 Communication Controller Mapping

`CommunicationControllerMapping` specifies the `CommunicationPeripheral` hardware 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
communicationPeripheral	CommunicationPeripheral	1	ref	

Table 2.25: CommunicationControllerMapping

2.4.3 HW-Port Mapping

`HWPortMapping` specifies the `ECUCommunicationPort` hardware to realize the specified `CommunicationConnector` 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 ECUCommunicationPort hardware (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
ecuCommunicationPort	ECUCommunicationPort	1	ref	Reference to the Peripheral in the ECU Resource Template

Table 2.26: HWPortMapping

3 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 5.2) that is going to be exchanged between mapped Software Components onto different ECUs is a consequence of a connection between such application Software Components.

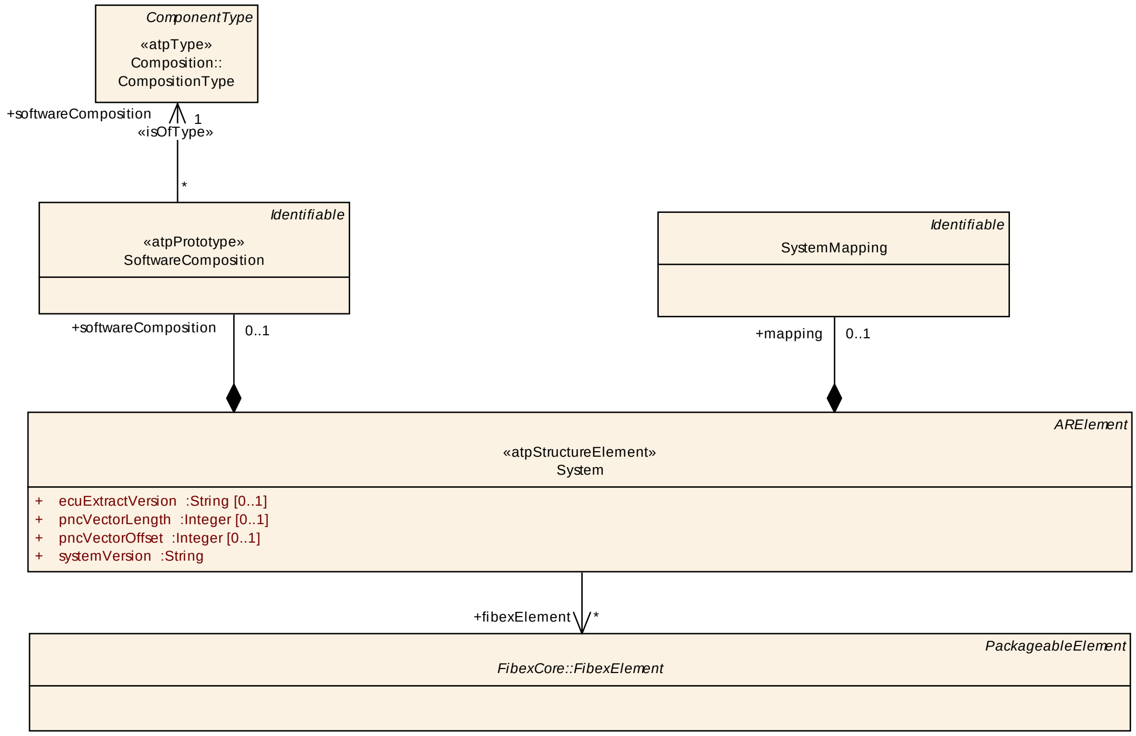

Figure 3.1: Inclusion of a (top-level) Software Composition into an AUTOSAR system (SystemTemplate)

In AUTOSAR, Software Components can either be atomic (`AtomicSoftwareComponentType`) or may consist of a composition of other Software Components `CompositionType` [5]. In order to assemble non-trivial applications from AUTOSAR components, such compositions can be built up hierarchically, until the outermost `CompositionType` 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 Ecu Extract outside ports for the outermost composition are allowed. Since the Ecu Extract represents the view on one Ecu, there is the need to define the communication of this extract with the outside world. Two approaches are available how the external communication of an ECU 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 Ecu 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 `CompositionType` as its application software system input by owning exactly one `SoftwareComposition` class, which points to the `CompositionType` forming the input via its <<isOfType>> relationship as shown in Figure 3.1.

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

Class	**SoftwareComposition**			
Package	M2::AUTOSARTemplates::SystemTemplate			
Note	The Top-level software composition, containing all software components in the System in a hierarchical structure. The contained ComponentPrototypes are fully specified by their ComponentTypes (including PortPrototypes, PortInterfaces, DataElementPrototypes, InternalBehavior etc.), and their ports are interconnected using ConnectorPrototypes.			
Base	ARObject,Identifiable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
softwareComposition	CompositionType	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: SoftwareComposition

4 Mapping

A central part of the system generation process is the mapping of software components (ComponentPrototypes) to ECUs, and the subsequent mapping of the communication between these software components to bus frames. Input to the software component mapping are the software composition, 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.

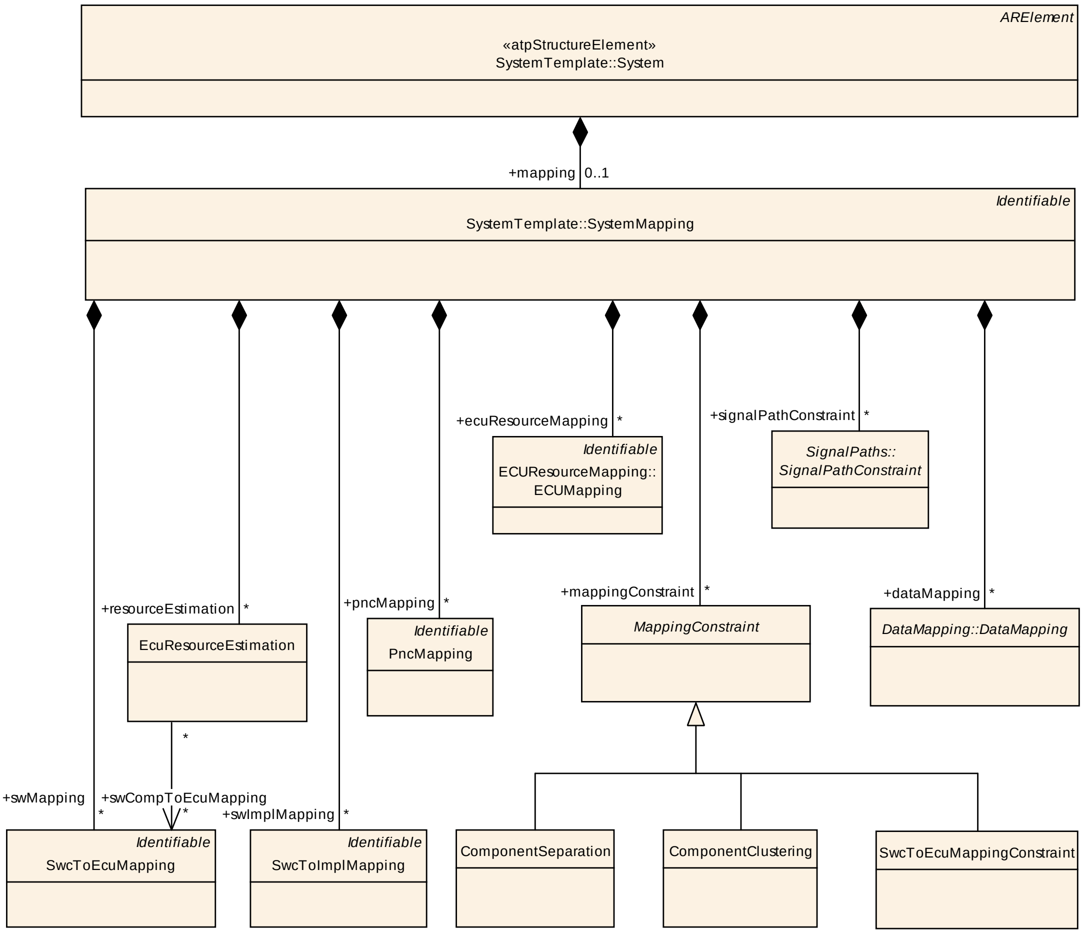

Figure 4.1: Mapping Overview (Mapping)

The following mappings are defined:

- The SwCompToEcuMapping meta-class maps one or several ComponentPrototypes to ECUs. In the System Constraint Description it is possible to predefine the mapping of ComponentPrototypes 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 ComponentPrototypes (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 ComponentPrototypes 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 data elements and operations 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			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
dataMapping	DataMapping	*	aggr	The data mappings defined.
ecuResourceMapping	ECUMapping	*	aggr	
mappingConstraint	MappingConstraint	*	aggr	Constraints that limit the mapping freedom for the mapping of SW components to ECUs.
pncMapping	PncMapping	*	aggr	
resourceEstimation	EcuResourceEstimation	*	aggr	Resource estimations for this set of mappings, zero or one per ECU instance.
signalPathConstraint	SignalPathConstraint	*	aggr	Constraints that limit the mapping freedom for the mapping of data elements to signals.
swImplMapping	SwcToImplMapping	*	aggr	The mappings of AtomicSoftwareComponent Instances to Implementations.
swMapping	SwcToEcuMapping	*	aggr	The mappings of SW components to ECUs.

Table 4.1: SystemMapping

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 `ComponentPrototypes` to one ECU instance. Figure 4.2 shows this structure. The predefinition will force the system generator to use the specified mapping.

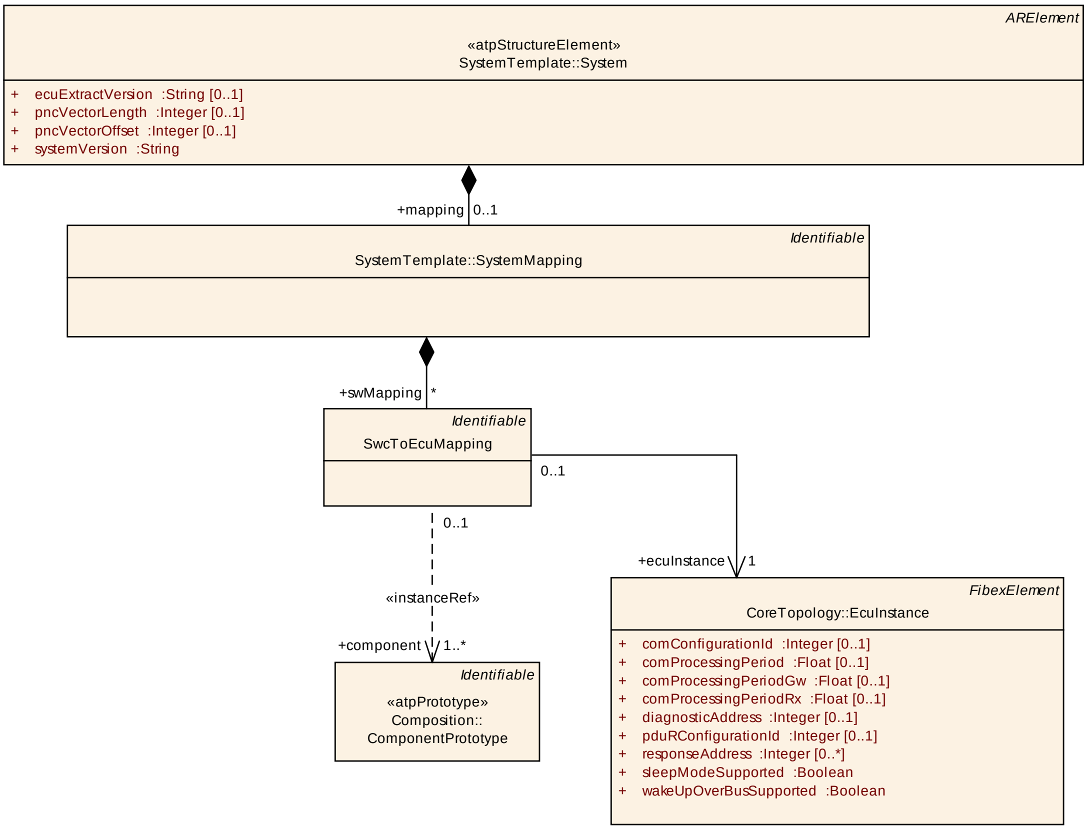

Figure 4.2: SW component to ECU mapping (SwcToEcuMapping)

The following table describes the `SwcToEcuMapping` in detail.

Class	**SwcToEcuMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Note	Map software components to a specific ECU Instance.			
Base	ARObject,Identifiable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
component	ComponentPrototype	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 ComponentPrototype INSIDE the Composition to another ECU Instance the inner mapping overrides the outer mapping.
ecuInstance	EcuInstance	1	ref	EcuInstance is a reference to an ECU Instance description

Table 4.2: SwcToEcuMapping

4.1.2 Software Component to Implementation Mapping

As several implementations may exist for the same AtomicSoftwareComponentType, 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 ComponentPrototype relating to the same AtomicSoftwareComponentType. This is illustrated in Figure 4.3.

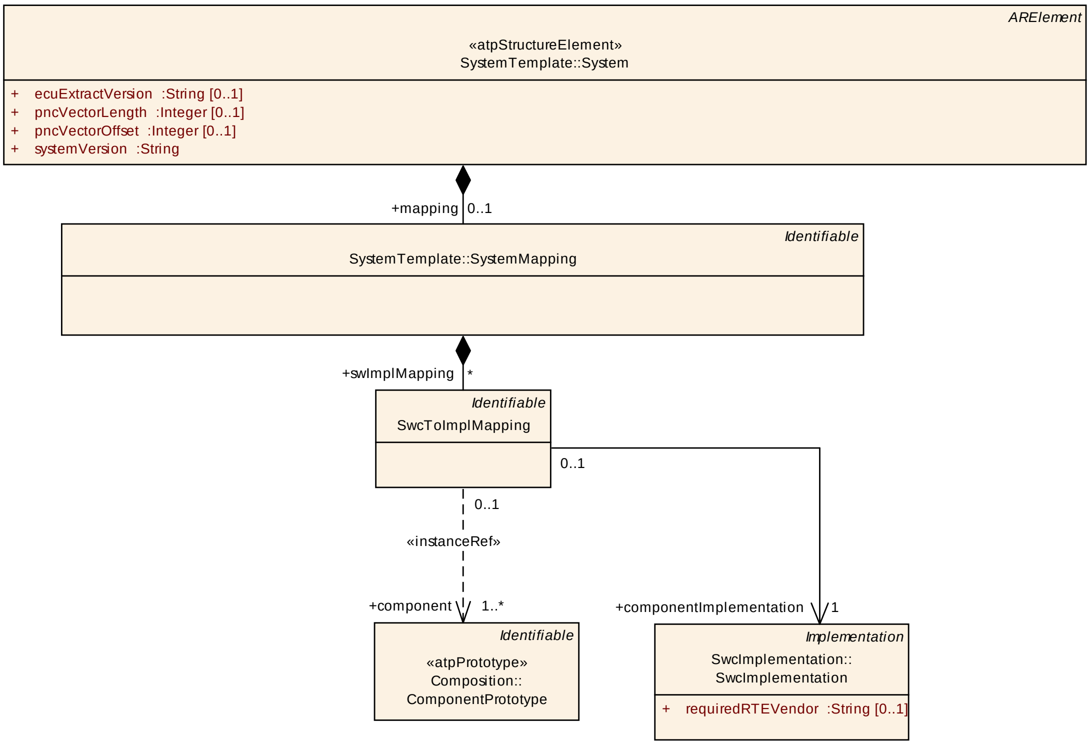

Figure 4.3: SW Component to Implementation mapping (SwcToImplMapping)

The following table contains the detailed description of SwcToImplMapping:

Class	**SwcToImplMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Note	Map instances of an AtomicSoftwareComponentType to a specific Implementation.			
Base	ARObject,Identifiable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
component	ComponentPrototype	1..*	iref	Reference to the software component instances that are being mapped to the specified Implementation. The targeted ComponentPrototype needs be of the AtomicSoftwareComponentType 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***

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`.

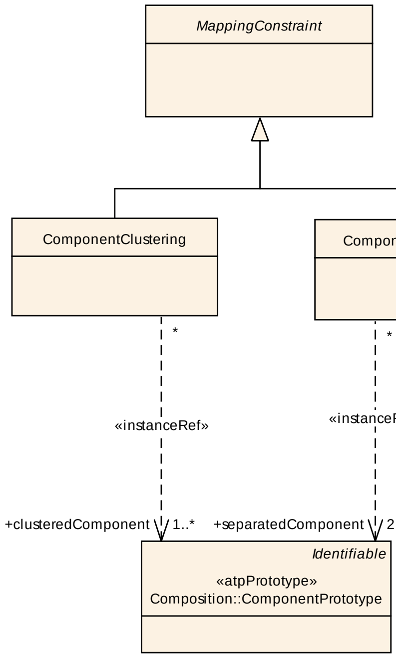

Figure 4.4: Details on ComponentClustering and ComponentSeparation (SwcClustering)

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 `ComponentPrototype` 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***

Attribute	Datatype	Mul.	Kind	Note
clusteredComponent	ComponentPrototype	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].

A *separation* constraint is part of a `MappingConstraint` element and it must refer to two `ComponentPrototype` 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	ComponentPrototype	2	iref	The two components that have to be mapped to different ECUs

Table 4.6: ComponentSeparation

[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.

4.1.3.3 SwcToEcuMappingConstraint

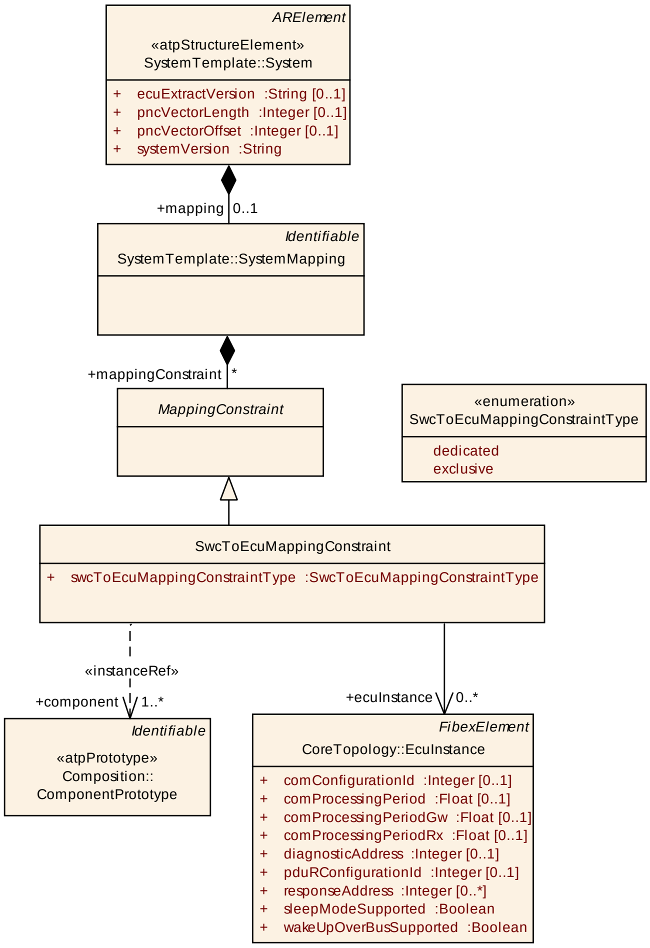

Figure 4.5: Dedicated and exclusive Mapping of SWC to ECUs

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	ComponentPrototype	1..*	iref	
ecuInstance	EcuInstance	*	ref	If the dedicated mapping is described, the ComponentPrototypes can only be mapped to these referenced ECUInstances. If the exclusive mapping is described, the ComponentPrototypes 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 DataElementPrototypes or OperationPrototypes of SW Component Ports to SystemSignals.

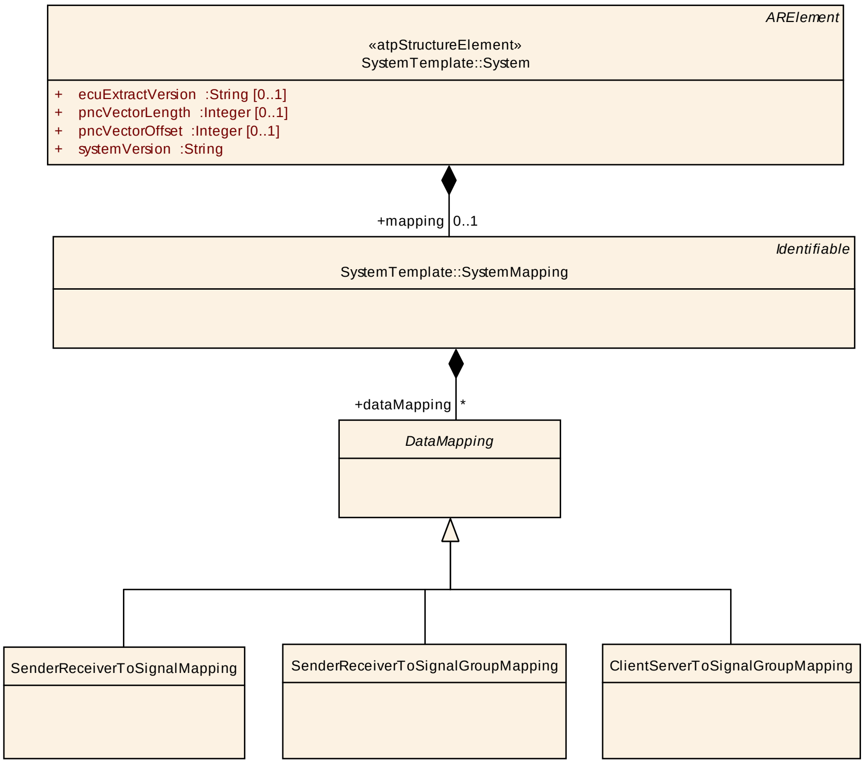

Figure 4.6: Overview: Data Mapping Description (DataMappingOverview)

SystemSignals represent DataElementPrototypes and OperationPrototypes in the communication description. This chapter describes how the DataElementPrototypes and OperationPrototypes 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. According to the COM Specification, signal groups without signals are allowed. These have a "signalLength" = 0. In this case there shall be an "update-bit" configured.			
Base	ARElement,ARObject,AbstractSignal,Identifiable,PackageableElement			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
dataType	Datatype	0..1	ref	Optional reference to a SystemSignal's datatype in case the System Description doesn't use a complete Software Component Description (VFB View). This supports the inclusion of legacy system signals. This reference can be used to configure the "ComSignalDataInvalidValue" and the Data Semantics. If a full DataMapping exist for the SystemSignal this information is additionally available from the mapped DataElement. In this case the referenced datatypes needs to be compatible.
initValue	ValueSpecification	0..1	ref	Optional reference to a SystemSignal'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 reference 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 SystemSignal length of zero bits is allowed.

Table 4.9: SystemSignal

[constr_3074] No `TransmissionAcknowledgementRequest` for multiple senders ⌈ If more than one `SenderComSpec` exist (in different `PortPrototype`s on atomic level) that refer to dataElements effectively mapped to the same `SystemSignal` it is not allowed that any `SenderComSpec` aggregates `transmissionAcknowledge`. ⌋

Please note that the term "'effectively mapped"' refers to the fact that the DataMapping can refer to a dataElement in a "delegation" PortPrototype on the surface of a rootSwComposition of an Ecu Extract OR to PortPrototypes inside the rootSwComposition. Both ways shall be considered.

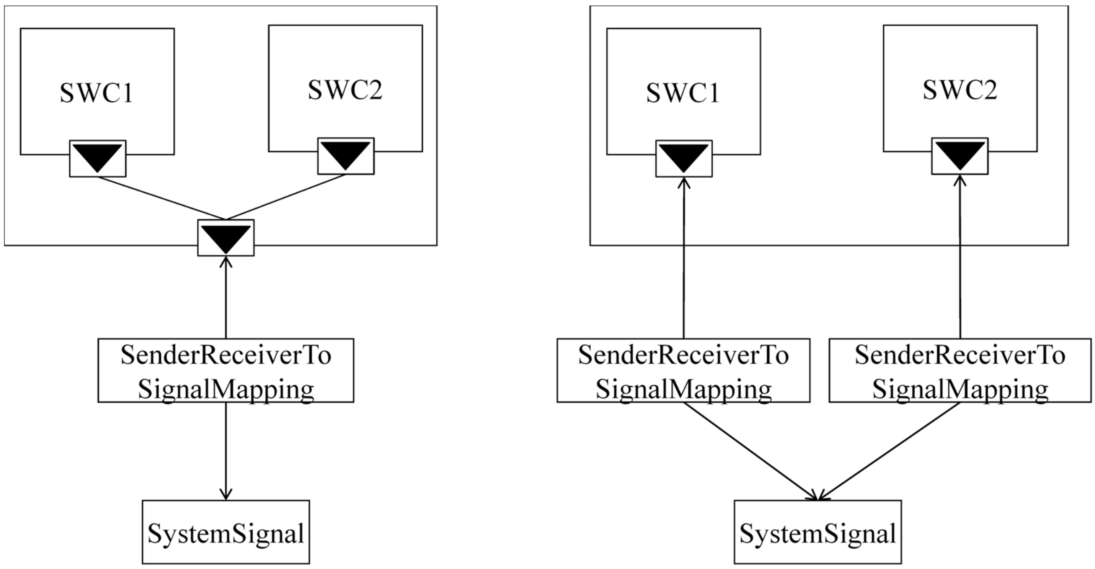

Figure 4.7: Example for data elements that are effectively mapped to the same SystemSignal

[constr_3072] Allowed SystemSignal dataType references ⌈ The dataType that is referenced by the SystemSignal may only be one of the following alternatives:

- a PrimitiveType
- an ArrayType that boils down to a UInt8 data type, i.e. where the reference ArrayType.element.type refers to one of the following alternatives:
  - an OpaqueType where $0 <$ numberOfBits ≤ 8
  - an IntegerType where $0 \leq$ lowerLimit.value ≤ 255 **and** $0 \leq$ upperLimit.value ≤ 255
  - a CharType
  - a StringType
  - a BooleanType

⌋

In case that a DataElementPrototype is transferred over the network a SystemSignal is being defined representing the DataElementPrototype on the network. In case of 1:n communication the DataElementPrototype in the ProvidePort of the ComponentPrototype is still mapped to only one SystemSignal.

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

4.2.1 Mapping of Data Prototypes on System Signals

This chapter describes how DataPrototypes, 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 SoftwareComposition is expressed by using AssemblyConnectorPrototypes and DelegationConnectorPrototypes to connect the PPortPrototypes and RPortPrototypes of ComponentPrototypes with each other on the VFB-level.

Ultimately, each chain of ConnectorPrototypes 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 DataElement or ArgumentPrototype contained in these Ports referenced by the ConnectorPrototype 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 DataElement 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 DataElement.

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

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

1) For each SystemSignal that is not part of a SystemSignalGroup: in a complete System Description exactly one data mapping shall be defined (P-Port or R-Port). Preference: P-Port

2) For each `SystemSignal` that is part of exactly one `SystemSignalGroup` and is not transmitted additionally as standalone `SystemSignal`: in a complete System Description exactly one data mapping shall be defined (P-Port or R-Port). Preference: P-Port

3) For each `SystemSignalGroup`: in a complete System Description exactly one `DataMapping` shall be defined (P-Port or R-Port). Preference: P-Port

In a complete System Description, it is sufficient to refer to the `DataElementPrototype` 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 `ProvidePorts`.

4) For a `SystemSignal` which is

- part in several `SystemSignalGroup`s
- part in at least one `SystemSignalGroup` and at the same time is transmitted additionally as standalone `SystemSignal`

several data mappings may be defined.

Note: because of the defined assignment of `ISignal` pointing to a `SystemSignalGroup` it is possible to assign the occurrence of each `SystemSignal` (in- and out-side of `SystemSignalGroups`) to the corresponding `ISignal`.

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

In an ECU extract of the system description, where only the relevant information for an individual ECU is 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. Therefore a data mapping can be provided on `ProvidePorts` and on `RequirePorts`.

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

The ECU Extract is introduced to allow a collaboration between an OEM and a Supplier. 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. In such a scenario often only the outer shell of a Software Composition (an empty composition) is defined by an OEM and is delivered to the supplier. The supplier adds the substructure to the Composition by adding atomic `ComponentPrototypes` and `ConnectorPrototypes`. But the supplier must respect the predefined data mapping on the Software Composition. For the RTE generation only the mapping on the atomic SWCs possesses validity. Therefore the existing data mappings on compositions needs to be transfered to the atomic SWCs.

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

The `DataElementPrototype` meta-class is defined in the SW Component Template. The datatype of the data element 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 `DataElementPrototypes` with primitive datatypes and the `SystemSignal` (see Figure 4.8). 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.

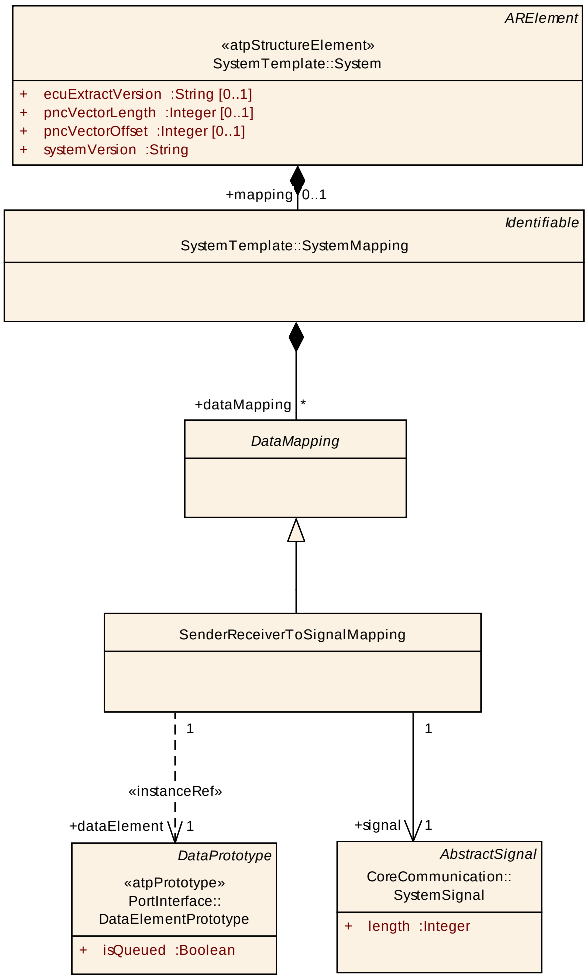

Figure 4.8: Mapping of data elements with primitive datatypes (SenderRecPrimitiveTypeMapping)

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 DataElementPrototype in the ProvidePort of the ComponentPrototype is still mapped to only one SystemSignal.			
Base	ARObject,DataMapping			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
dataEleme nt	DataElementPr ototype	1	iref	Reference to the data element, which ought to be sent over the Communication bus. This DataElement is described in the Software Component Template.
signal	SystemSignal	1	ref	Reference to the system signal used to carry the data element.

Table 4.10: SenderReceiverToSignalMapping

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

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

The RTE is required to treat AUTOSAR signals transmitted using sender-receiver communication consistently. To achieve this, the "signal group" mechanisms shall be utilized. The complex data type shall be decomposed into single signals. As this set of single signals has to be treated consistent, it is placed in a "signal group". There is one exception to this rule: it is allowed to map an array DataPrototype consisting of UINT8 elements to exactly one SystemSignal via the SenderReceiverToSignalMapping. A UINT8 element may be a String or an ArrayType that contains ArrayElements of IntegerType with range 0..255.

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

If the "signal group" mechanisms is used each PrimitiveType in the complex data type will be one SystemSignal in the System Description. The relationship between the SystemSignals and the DataElementPrototypes is provided in the SenderReceiverToSignalGroupMapping (see Figure 4.9).

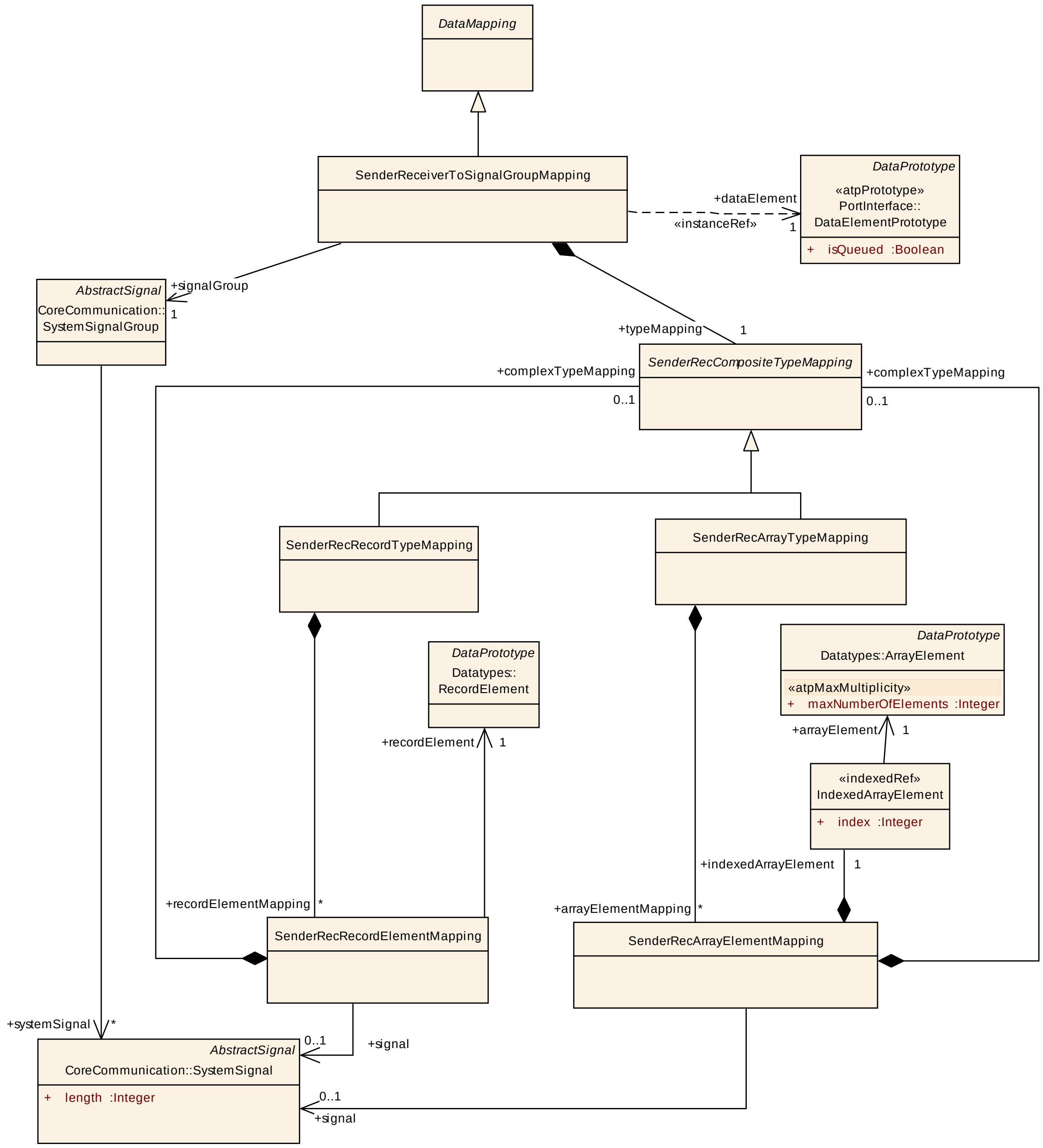

Figure 4.9: Mapping of data elements with composite datatypes (SenderRecComposite-TypeMapping)

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	DataElementPrototype	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	

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 compositeType is an Array, the "ArrayTypeMapping" will be used.			
Base	ARObject,SenderRecCompositeTypeMapping			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
arrayElementMapping	SenderRecArrayElementMapping	*	aggr	Each ArrayElement must be mapped on a SystemSignal.

Table 4.13: SenderRecArrayTypeMapping

Class	**SenderRecRecordTypeMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	If the compositeType is a Record, the "RecordTypeMapping" will be used.			
Base	ARObject,SenderRecCompositeTypeMapping			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
recordElementMapping	SenderRecRecordElementMapping	*	aggr	Each RecordElement 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 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**
complexTypeMapping	SenderRecCompositeTypeMapping	0..1	aggr	This aggregation will be used if the element is composite.
recordElement	RecordElement	1	ref	Reference to a RecordElement in the context of the dataElement or in the context of a composite element.
signal	SystemSignal	0..1	ref	Reference to the system signal used to carry the primitive RecordElement.

Table 4.15: SenderRecRecordElementMapping

Class	**SenderRecArrayElementMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	The ArrayElement 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 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.
signal	SystemSignal	0..1	ref	Reference to the system signal used to carry the primitive ArrayElement.

Table 4.16: SenderRecArrayElementMapping

4.2.1.3 Mapping of Client Server Operations to Signal Groups

The Client/Server interfaces aggregate a number of 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 datatype of an argument may be a primitive one or a composite one. Each primitive argument will be mapped directly onto one `SystemSignal`. The complex data type must be decomposed into single signals.

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

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 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.

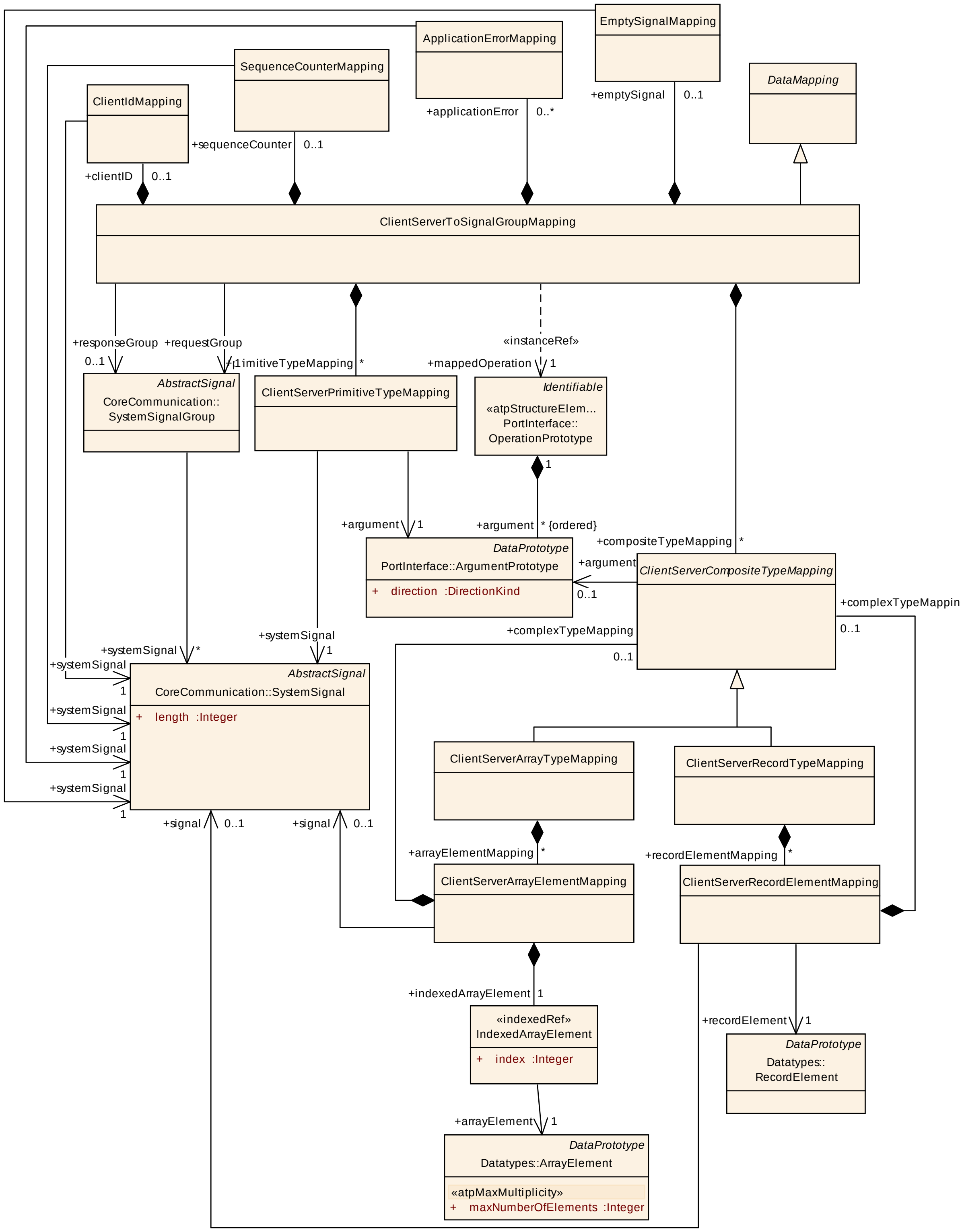

Figure 4.10: Operation Mapping (ClientServerOperationMapping)

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	*	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	According to the COM Specification, signal groups without signals are allowed. These have a "signalLength" = 0. In this case there shall be an "update-bit" configured.
mappedOperation	OperationPrototype	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	1	ref	Reference to the signal group which contains the references to request signals used to transport the OUT arguments of the operation or the empty signal if the operation doesn't have OUT arguments.
responseGroup	SystemSignalGroup	0..1	ref	Reference to the signal group which contains the references to response signals used to transport the IN 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 4.17: ClientServerToSignalGroupMapping

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.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
systemSignal	SystemSignal	1	ref	Reference to the SystemSignal with the ClientID.

Table 4.18: 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.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
systemSignal	SystemSignal	1	ref	Reference to the SystemSignal with the SequenceCounter.

Table 4.19: SequenceCounterMapping

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

Table 4.20: ApplicationErrorMapping

Class	**EmptySignalMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	According to the COM Specification, signal groups without signals are allowed. These have a "signalLength" = 0. In this case there shall be an "update-bit" configured.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
systemSignal	SystemSignal	1	ref	Reference to a SystemSignal with "signalLength" = 0 and an UpdateBit.

Table 4.21: 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	ArgumentPrototype	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 4.22: 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	ArgumentPrototype	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 4.23: ClientServerCompositeTypeMapping

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

Table 4.24: ClientServerArrayTypeMapping

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

Table 4.25: ClientServerRecordTypeMapping

Class	ClientServerArrayElementMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	The ArrayElement 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 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.
signal	SystemSignal	0..1	ref	Reference to the system signal used to carry the primitive ArrayElement.

Table 4.26: ClientServerArrayElementMapping

Class	ClientServerRecordElementMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	Mapping of a primitive record element to a SystemSignal. 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
complexTypeMapping	ClientServerCompositeTypeMapping	0..1	aggr	This aggregation will be used if the element is composite.
recordElement	RecordElement	1	ref	Reference to a RecordElement in the context of the mappedOperation or in the context of a composite element.
signal	SystemSignal	0..1	ref	Reference to the system signal used to carry the primitive RecordElement.

Table 4.27: 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.

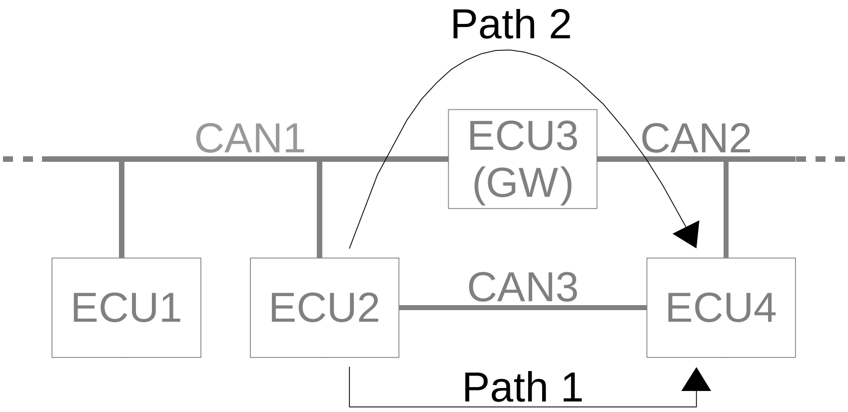

Figure 4.11: Example for a Communication Path

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.

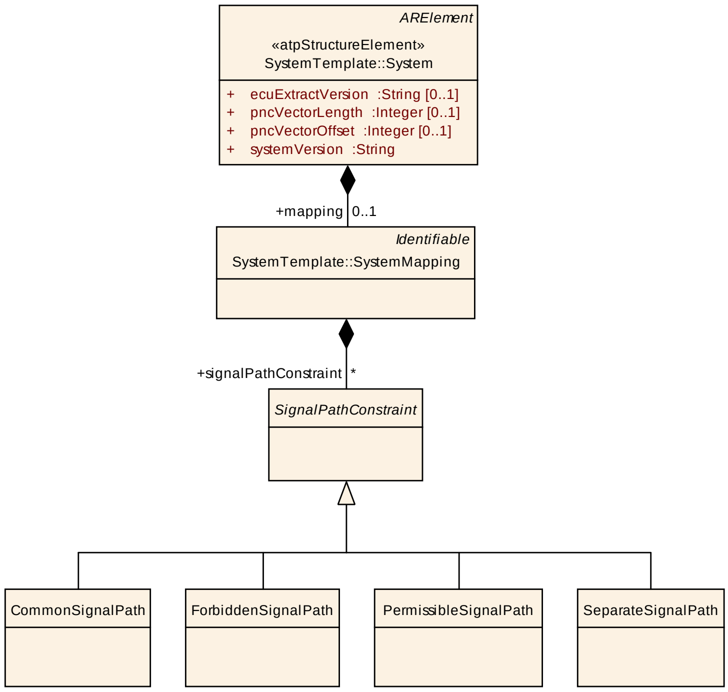

Figure 4.12: Communication Path Description (SignalPathConstraints)

4.2.2.1 CommonSignalPath

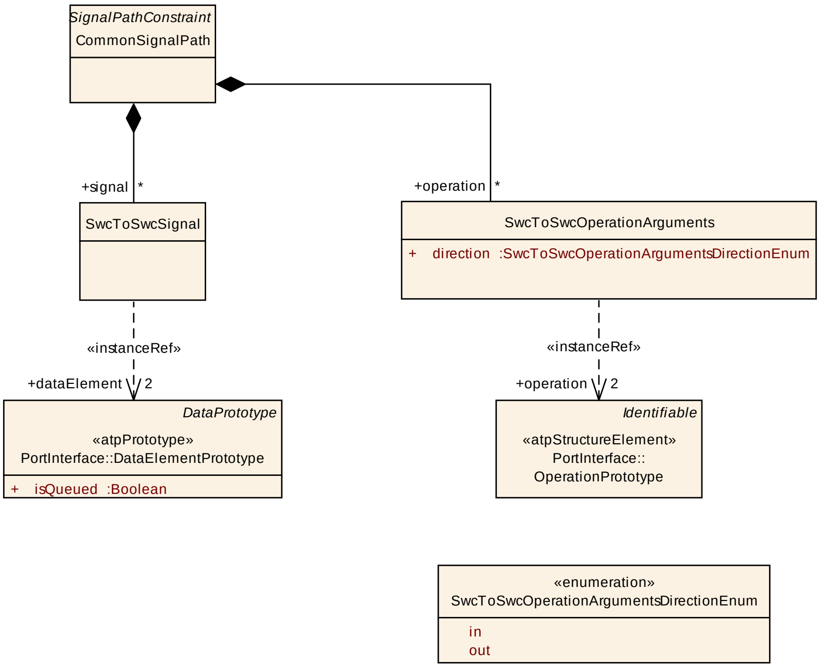

Figure 4.13: Description of signals that must take the same way in the topology (CommonSignalPath)

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 4.28: 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	DataElementPrototype	2	iref	Reference to a data element on the PPort and to the same data element on the RPort.

Attribute	Datatype	Mul.	Kind	Note

Table 4.29: 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	OperationPrototype	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 4.30: 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 4.31: SwcToSwcOperationArgumentsDirectionEnum

4.2.2.2 ForbiddenSignalPath

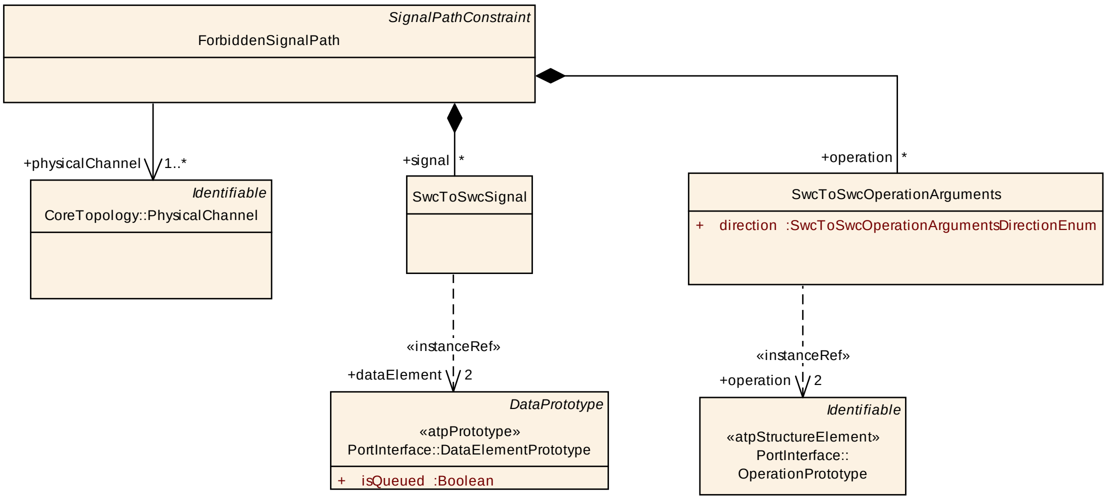

Figure 4.14: Description of the signal path that a signal must not take in the topology (ForbiddenSignalPath)

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 4.32: ForbiddenSignalPath

4.2.2.3 PermissibleSignalPath

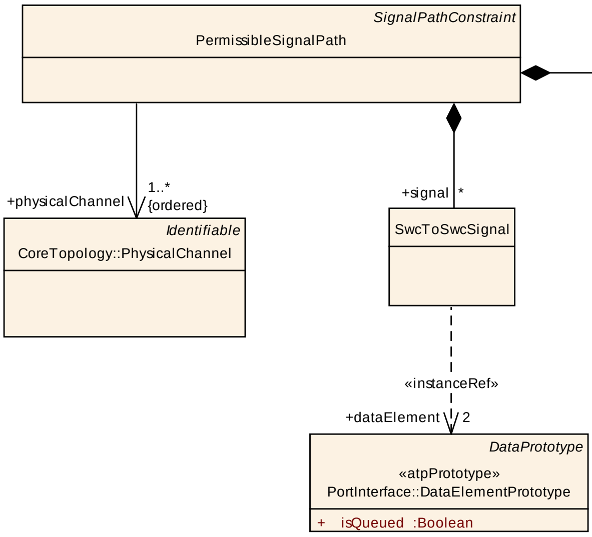

Figure 4.15: Description of the signal path that a signal must take in the topology (PermissibleSignalPath)

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	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 4.33: PermissibleSignalPath

4.2.2.4 SeparateSignalPath

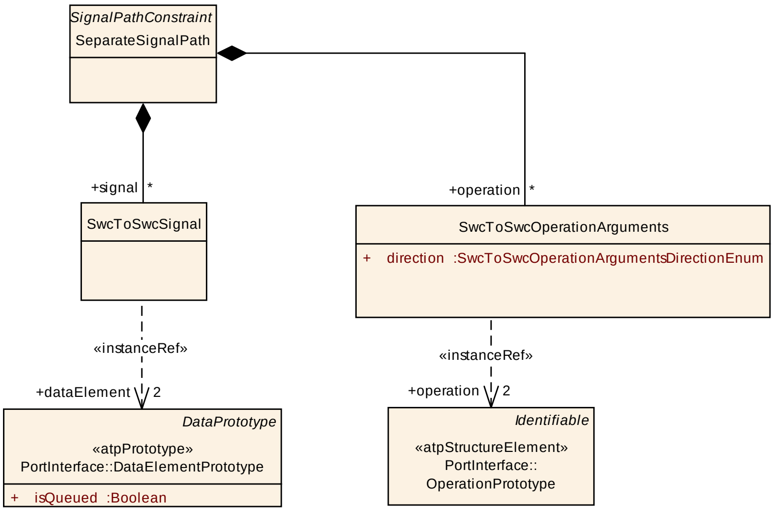

Figure 4.16: Description of signals that must not take the same way in the topology (SeparateSignalPath)

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).			
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 4.34: 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.

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

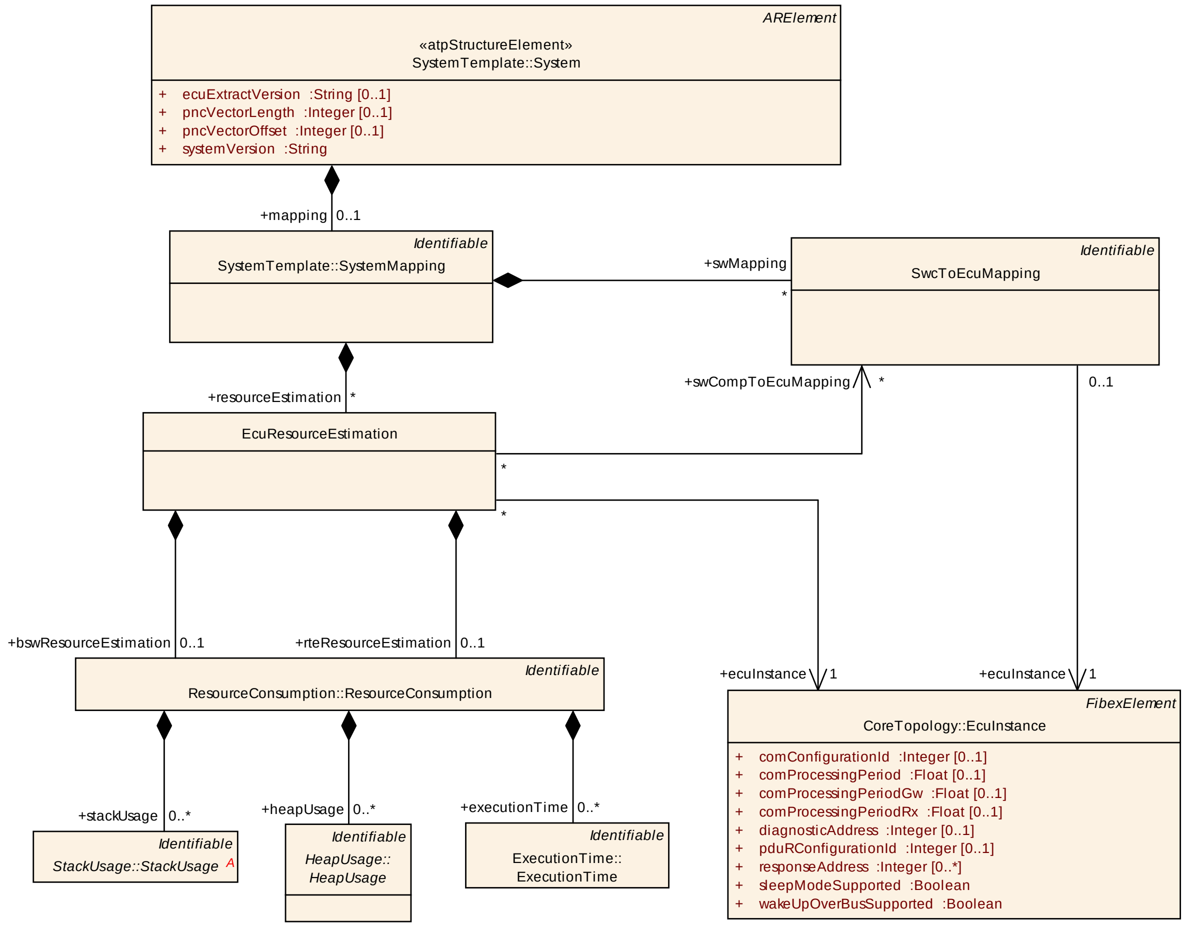

Figure 4.17: ECU resource estimations (ResourceEstimation)

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.
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 4.35: EcuResourceEstimation

Class	**ResourceConsumption**			
Package	M2::AUTOSARTemplates::CommonStructure::ResourceConsumption			
Note	Description of consumed resources by one implementation of a software.			
Base	ARObject,Identifiable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
executionTime	ExecutionTime	*	aggr	Collection of the execution time descriptions for the runnable entities of this implementation.
heapUsage	HeapUsage	*	aggr	Collection of the heap memory allocated by this implementation.
objectFileSection	MemorySection	1..*	aggr	Provides additional information to the sections of the object-file containing the implementation of the SW-Component
stackUsage	StackUsage	*	aggr	Collection of the stack memory usage for each runnable entity of this implementation.

Table 4.36: ResourceConsumption

The element `ResourceConsumption` and the subelements `heapUsage`, `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 [5].

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

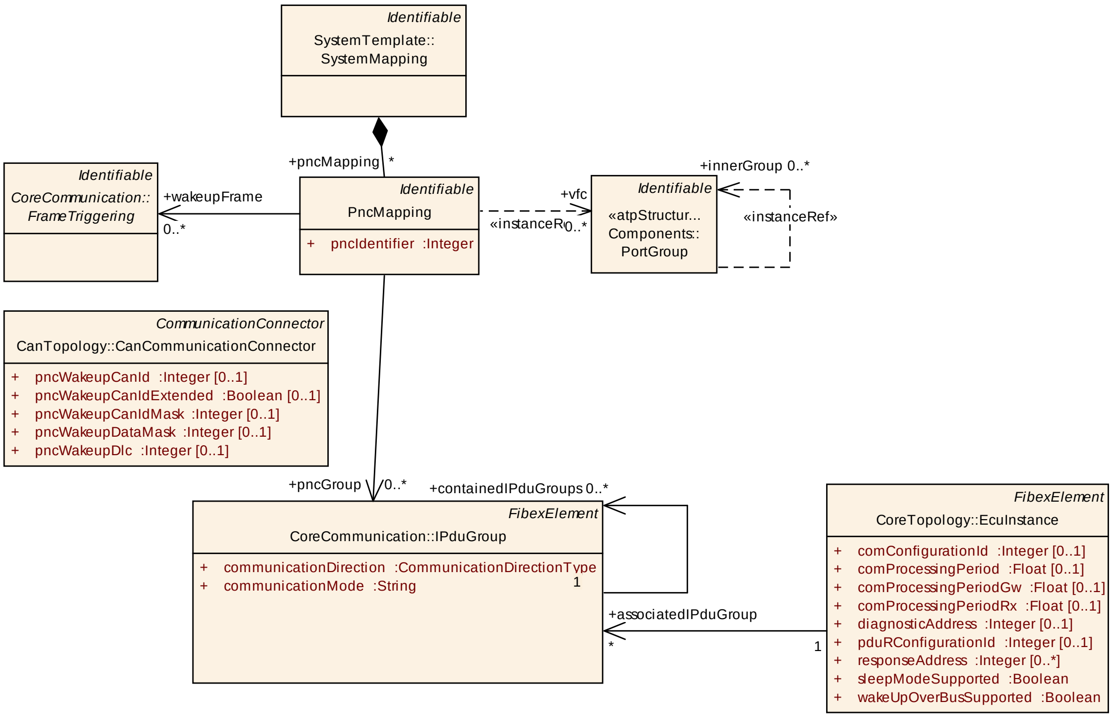

Figure 4.18: Mapping of Virtual Function Clusters onto Partial Network Clusters

Class	**PncMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
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,Identifiable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Attribute	Datatype	Mul.	Kind	Note
pncGroup	IPduGroup	*	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	Integer	1	attr	Identifier of the Partial Network Cluster. This number represents the absolute bit position of this Partial Network Cluster in the NM Pdu.
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.
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 4.37: PncMapping

[constr_3039] `pncIdentifier` range ⌈ The `pncIdentifier` value shall be in the range of 8..63. ⌋

The runtime information that is used to coordinate the request/release information of all partial networks is called `pncVector`. If there is an `ISignalToIPduMapping` aggregated by `NmPdu` that fully matches the interval defined by `System.pncVectorOffset` and `System.pncVectorLength` then the corresponding `ISignal` represents the `pncVector`.

If such an `ISignalToIPduMapping` does not exist a `ComSignal` that represents the `pncVector` shall be created in the Base Ecu Configuration Value description. The `ComBitPosition` and `ComSignalLength` of this `ComSignal` shall be derived from `pncVectorOffset` and `pncVectorLength` respectively. Since the `ComSignalType` of the `pncVector` `ComSignal` is UINT8_N the `ComSignalEndianness` shall be configured as OPAQUE (see SWS_Com_00553), the `ComSignalEndianness` shall be set to PENDING and the `ComSignalInitValue` to 0.

In the system description the `NmPdu`s are described based on the actual network interaction (i.e. an ECU sends one `NmPdu` per network and receives a set of `NmPdu`s). Those `NmPdu` with user data have, according to section 5.1, `IPduTriggering`s and assigments to `IPduPort`s defined.

The `NmPdu`s contributing to partial networking also have the Nm user data layout specified to contain the `pncVector` according to [constr_3043].

Those Pdus which are used to perform the ECU internal communication between the basic software modules (like *EIRA*, *ERA*, *IRA*) are not described in the system description and need to be introduced to the ECU Configuration.

[constr_3040] Restriction of pncIdentifier values ⌈ The pncIdentifier value shall be within the range described by pncVectorOffset and pncVectorLength. ⌋

[constr_3041] pncVectorOffset range ⌈ The pncVectorOffset value shall be in the range of 1..7. ⌋

[constr_3042] pncVectorLength range ⌈ The pncVectorLength value shall be in the range of 1..6. ⌋

[constr_3043] pncVector configuration in AUTOSAR Com ⌈ The pncVector shall be configured as UINT8_N signal in AUTOSAR Com. ⌋

Attributes used to configure the Partial Network Wakeup of one specific Ecu are described in chapter 2.3.1.3.

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 2.0 [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.

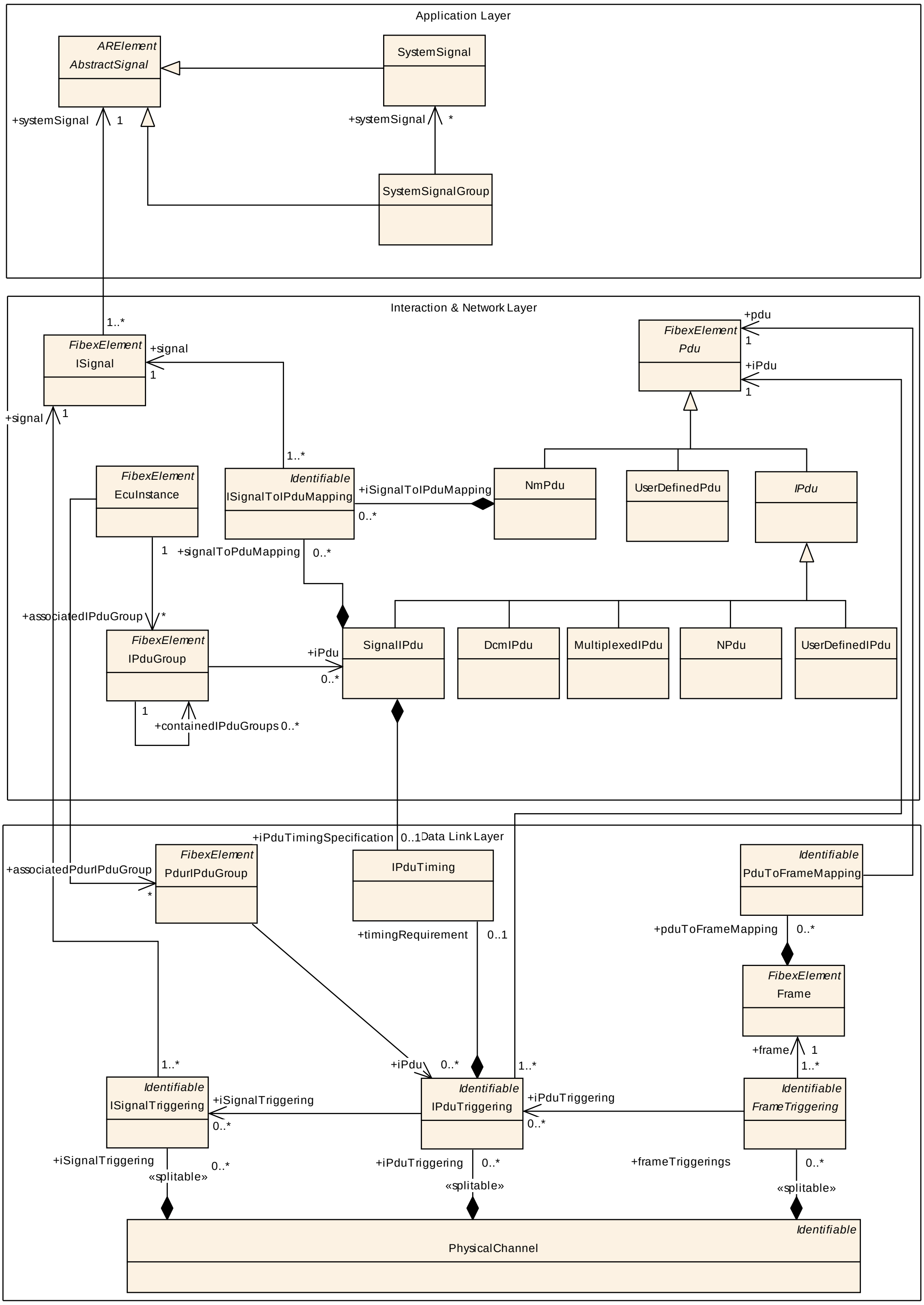

Figure 5.1: Communication Overview (FibexCore: Communication)

A `Frame` is a piece of information that is exchanged over the communication channels. A frame has a payload section of a certain length in bytes, which contains an arbitrary number of non-overlapping PDUs (I-Pdus, N-Pdus 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. [1]
- 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). Parts of the PCI are described in the System Template by the Frame Triggering element. Thus, the L-Pdu is described in the System Template by the Frame and the FrameTriggering element.

In case no multiplexing is performed the I-PDUs of COM are passed via the PDU Router directly to the communication interfaces. Therefore the maximum length of an I-PDU depends of the maximum length of the L-PDU of the underlying communication interface. For CAN and LIN the maximum L-PDU length is 8 bytes. For FlexRay the maximum L-PDU length is 254 bytes. Only the I-PDUs from the DCM are transported via the Transport Protocol. The Transport Protocols are described in more detail in chapter 5.12.

If multiplexing is performed an IPdu is routed between the IPdu Multiplexer and the Interface 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 5.5.

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 5.10.

[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. The Transport Protocols are described in more detail in chapter 5.12.

5.1 Triggerings and Ports

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

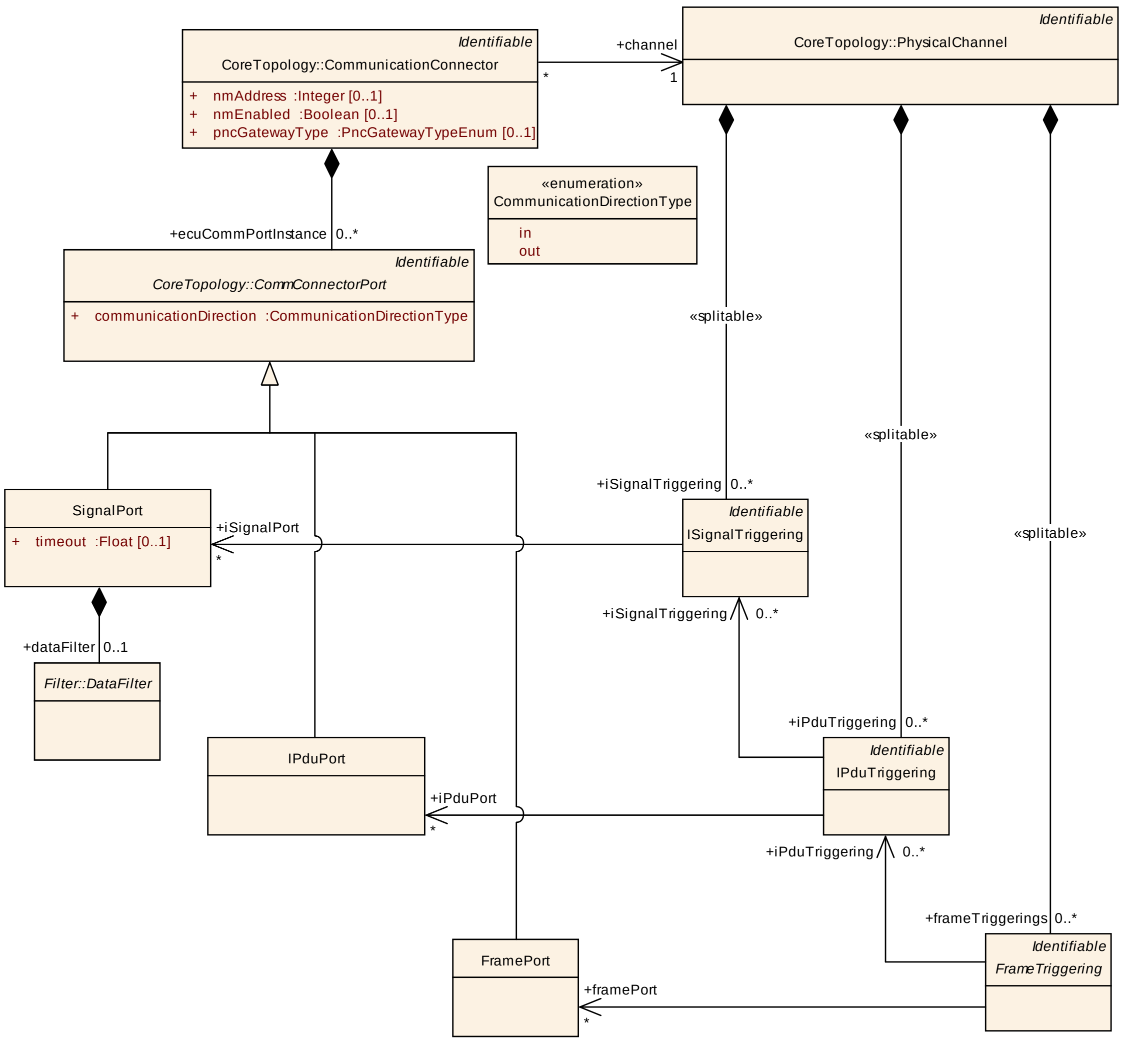

Figure 5.2: Communication Matrix (FibexCore: CommunicationMatrix)

A `FrameTriggering` need to fulfill requirements for contained Pdus that are defined by the corresponding `IPduTriggerings`. And the `IPduTriggering` 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`, `IPduTriggering` 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. The IPduTriggering can be used for the specification of timing requirements for FlexRay and Lin. This timing requirements needs to be fulfilled by the timing specification on the Frame.

In the System Template the FrameTriggering, IPduTriggering and ISignalTriggering are part of the PhysicalChannel definition thus defining the network view on the communication. Whether an ECU actually participates in the communication (Tx or Rx) is defined via the associations between the Triggering elements and Port elements of the CommunicationConnector. Figure 5.2 shows the relationship between the CommConnectorPort sub elements and the FrameTriggering, IPduTriggering and ISignalTriggering.

The following rules apply for the creation of Triggerings and Ports on the sender side:

- Application sends Signal
  - ISignalTriggering and SignalPort shall be created
  - optional DataMapping between SystemSignal and DataElementPrototype shall be created (in case of legacy signals this data mapping is not available).
- COM Signal Gateway
  - ISignalTriggering and SignalPort shall be created.
  - SignalMapping between two ISignalTriggerings shall be created (see chapter 6.3 for more details).
- Signal part of a Pdu but NOT sent by Application or Signal Gateway
  - Pdu part of Pdu Gateway
    - ISignalTriggerings for all contained signals shall be created (without connections to SignalPort).
    - IPduMapping between two IPduTriggerings shall be created (see chapter 6.2 for more details).
  - Pdu sent by Com
    - ISignalTriggerings for all contained signals shall be created (without connections to SignalPort).
- Neither Signal, Pdu nor Frame sent by the ECU
  - ISignalTriggering, PduTriggering and FrameTriggering on the PhysicalChannel shall be created. The connections to SignalPort, IPduPort, FramePort shall be skipped.

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 IPduTriggering and IPduPort shall be defined.
- DcmIPdus shall have IPduTriggering 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 IPduTriggering 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 IPduTriggering 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 IPduTriggering and IPduPort.

The following rule applies to the creation of ISignalTriggering and SignalPort:

[TPS_SYST_01058] Pdu Gateway where an Ecu only routes a IPduTriggering without being interested in the content ⌈ In case of a Pdu Gateway where an Ecu only routes a IPduTriggering without being interested in the content, the reference between the ISignalTriggerings (that are referred to by the IPduTriggering in the role iSignalTriggering) and the respective SignalPorts shall not be created. ⌋

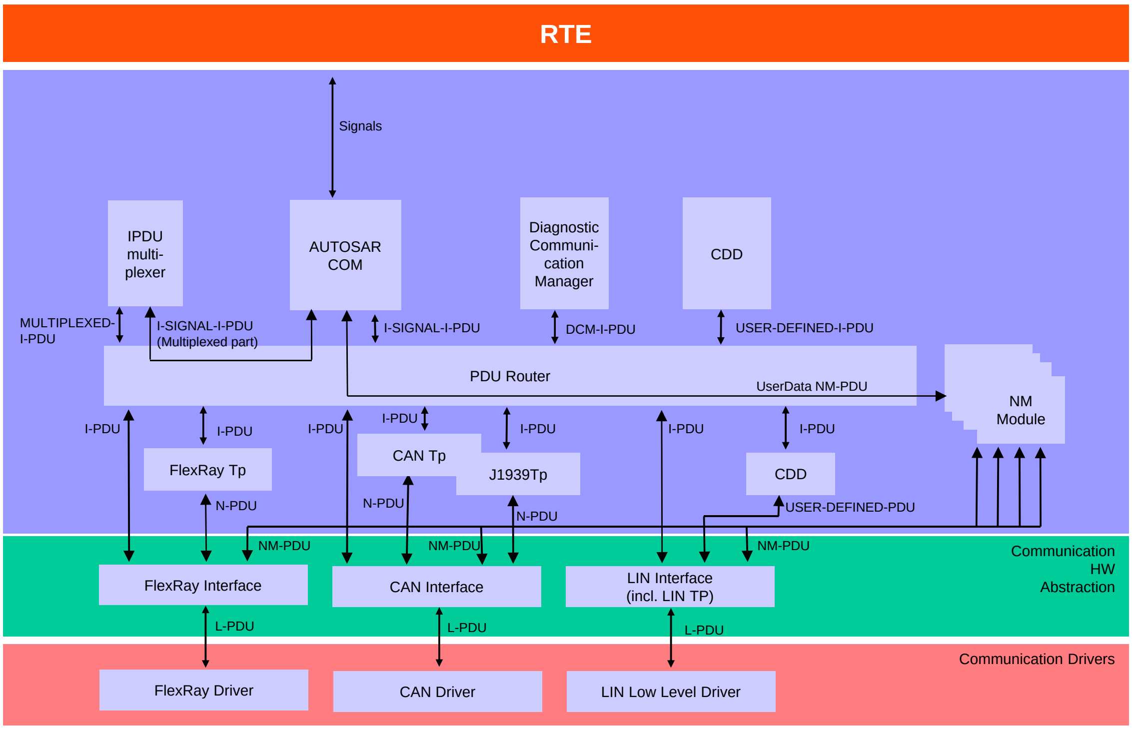

Figure 5.3: AUTOSAR Layered Architecture

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 a SignalPort, IPduPort or FramePort with the corresponding Triggering shall be created. A SignalPort 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			
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			
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 an IPduTriggering.			
Base	ARObject,CommConnectorPort,Identifiable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*

Table 5.3: IPduPort

Class	**SignalPort**			
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			
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	Float	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.

Attribute	Datatype	Mul.	Kind	Note

Table 5.4: SignalPort

[TPS_SYST_01059] Relationship between `FrameTriggering` and `CommConnectorPort` ⌈ For the reference between `FrameTriggering` and `FramePort` two approaches are supported:

- One to One relationship between `FrameTriggering` and `FramePort` per `EcuInstance`
- One `FramePort` per `communicationDirection` per `EcuInstance` exists and is referenced by all applicable `FrameTriggering`s (n to 1).

⌋

[TPS_SYST_01060] Relationship between `IPduTriggering` and `CommConnectorPort` ⌈ For the reference between `IPduTriggering` and `IPduPort` two approaches are supported:

- One to One relationship between `IPduTriggering` and `IPduPort` per `EcuInstance`
- One `IPduPort` per `communicationDirection` per `EcuInstance` exists and is referenced by all applicable `IPduTriggering`s (n to 1).

⌋

[TPS_SYST_01061] Relationship between `ISignalTriggering` and `CommConnectorPort` ⌈ For the reference between `ISignalTriggering` and `SignalPort` two approaches are supported:

- One to One relationship between `ISignalTriggering` and `SignalPort` per `EcuInstance`
- One `SignalPort` per `communicationDirection` per `timeout` per `EcuInstance` exists and is referenced by all applicable `IPduTriggering`s (n to 1).

⌋

Data Filters are described in more detail in the Software Component Template Specification [5].

5.2 ISignals

`SystemSignals` represent `DataElementPrototypes` and `OperationPrototypes` in the communication description and are defined independently of frames and communication clusters.

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` 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 a `ISignal` within an `SignalIPdu`.

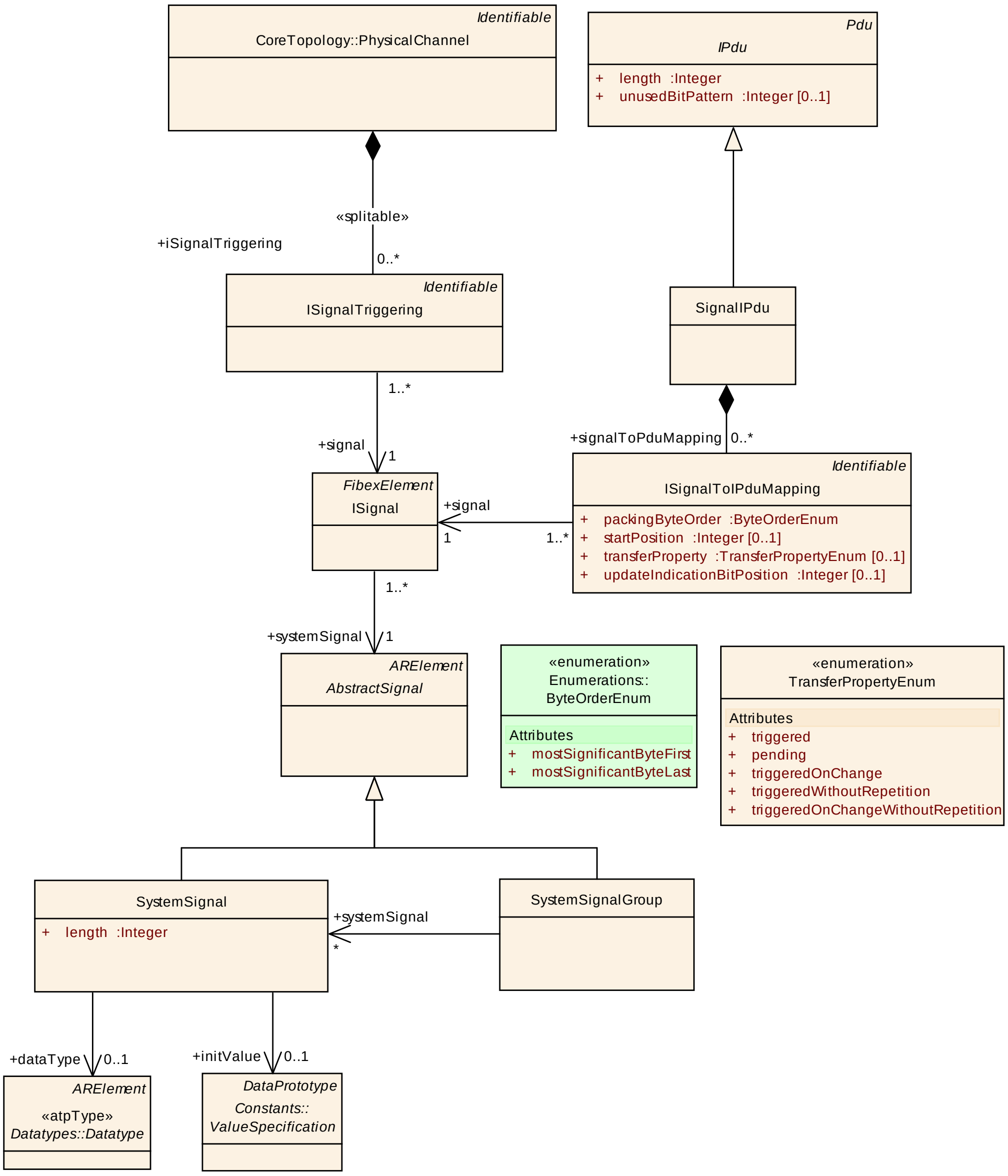

Figure 5.4: ISignals and the mapping into IPdus (FibexCore: SignalOverview)

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 atomic transmission of data. A signal group shall be transmitted and received atomically; 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.

A Com Signal Group must be defined in the Com Module for each `SystemSignalGroup` that contains `SystemSignals` that are transmitted or received by the regarded ECU. The Com group signals that are included within a Com signal group must be defined in the Com Module for each `ISignal` which has a reference to a System Signal that is associated by the `SystemSignalGroup`.

A `SystemSignal` contains an optional reference to a SystemSignal's datatype and to a initvalue in case the System Description doesn't use a complete Software Component Description (VFB View). This supports the inclusion of legacy system signals.

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. 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 the case of the SystemSignalGroup an ISignal must be created for the SystemSignalGroup and for each SystemSignal contained in the SystemSignalGroup. If a mapping for the SystemSignalGroup is defined, only the UpdateIndicationBitPosition is relevant, and the startPosition shall be ignored.			
Base	ARObject,FibexElement,Identifiable,PackageableElement			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
systemSignal	AbstractSignal	1	ref	Reference to the System Signal that is supposed to be transmitted in the ISignal.

Table 5.5: ISignal

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.			
Base	ARElement,ARObject,AbstractSignal,Identifiable,PackageableElement			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
systemSignal	SystemSignal	*	ref	Reference to a set of signals that must always be kept together.

Table 5.6: SystemSignalGroup

Class	**ISignalToIPduMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	An ISignalToIPduMapping describes the mapping of ISignals to SignalIPdus and defines the position of the ISignal within an SignalIPdu. 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 SignalIPdu. In case the ISignal/ISignalGroup is part of the Signal Gateway several ISignalToIPduMappings of the same ISignal are supported.			
Base	ARObject,Identifiable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
packingByteOrder	ByteOrderEnum	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) and "Big Endian" (MostSignificantByteFirst) can be selected. The value of this attribute impacts the absolute position of the signal into the SignalIPdu (see the startPosition attribute description).
signal	ISignal	1	ref	Reference to a ISignal that is mapped into the SignalIPdu. Several ISignalToPduMappings to the same ISignal are only relevant when the ECU handles the signal gateway.

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 SystemSignalGroup is defined, only the UpdateIndicationBitPosition is relevant, and the startPosition shall be ignored.
transferProperty	TransferPropertyEnum	0..1	attr	The triggered, 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 IPdu. The immediate transmission of the IPdu is caused even if only one Signal of an IPdu has the transferProperty triggered, triggeredWithoutRepetition, triggeredOnChange, triggeredOnChangeWithoutRepetition and all other Signals have the transferProperty pending. Also for ISignals which refer to GroupSignals of a SystemSignalGroup this attribute is relevant and shall be evaluated: • If none of the ISignals belonging to the GroupSignals of a SystemSignalGroup have a transferProperty defined the transferProperty of the ISignal referring to the SystemSignalGroup is considered. • If at least one of the ISignals belonging to the GroupSignals of a SystemSignalGroup has a transferProperty defined all ISignals belonging to the GroupSignals of a SystemSignalGroup shall have a transferProperty defined as well. All of the transferProperties of the ISignals belonging to the GroupSignals of a SystemSignalGroup 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. 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 SignalIPdu still undergoes a change.

Table 5.7: 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 5.8: TransferPropertyEnum

[constr_3024] Usage of `triggeredWithoutRepetition` and `triggeredOnChangeWithoutRepetition` is not allowed for signal groups and group signals. ⌈ The values `triggeredWithoutRepetition` and `triggeredOnChangeWithoutRepetition` shall not be used if the `ISignalToIPduMapping` refers to an `ISignal` that contains a reference to a `SystemSignalGroup` or to a `SystemSignal` that is part of a `SystemSignalGroup` (group signal). ⌋

5.2.1 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).

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

Example 5.5 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 5.5: PackingByteOrder Example

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

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

5.3 PDUs

The PDU Router deploys AUTOSAR COM and DCM `I-PDUs` onto different communication protocols. The PDU Router also determines if a transport protocol has to be used or not. [3] This information can be derived from the System Template.

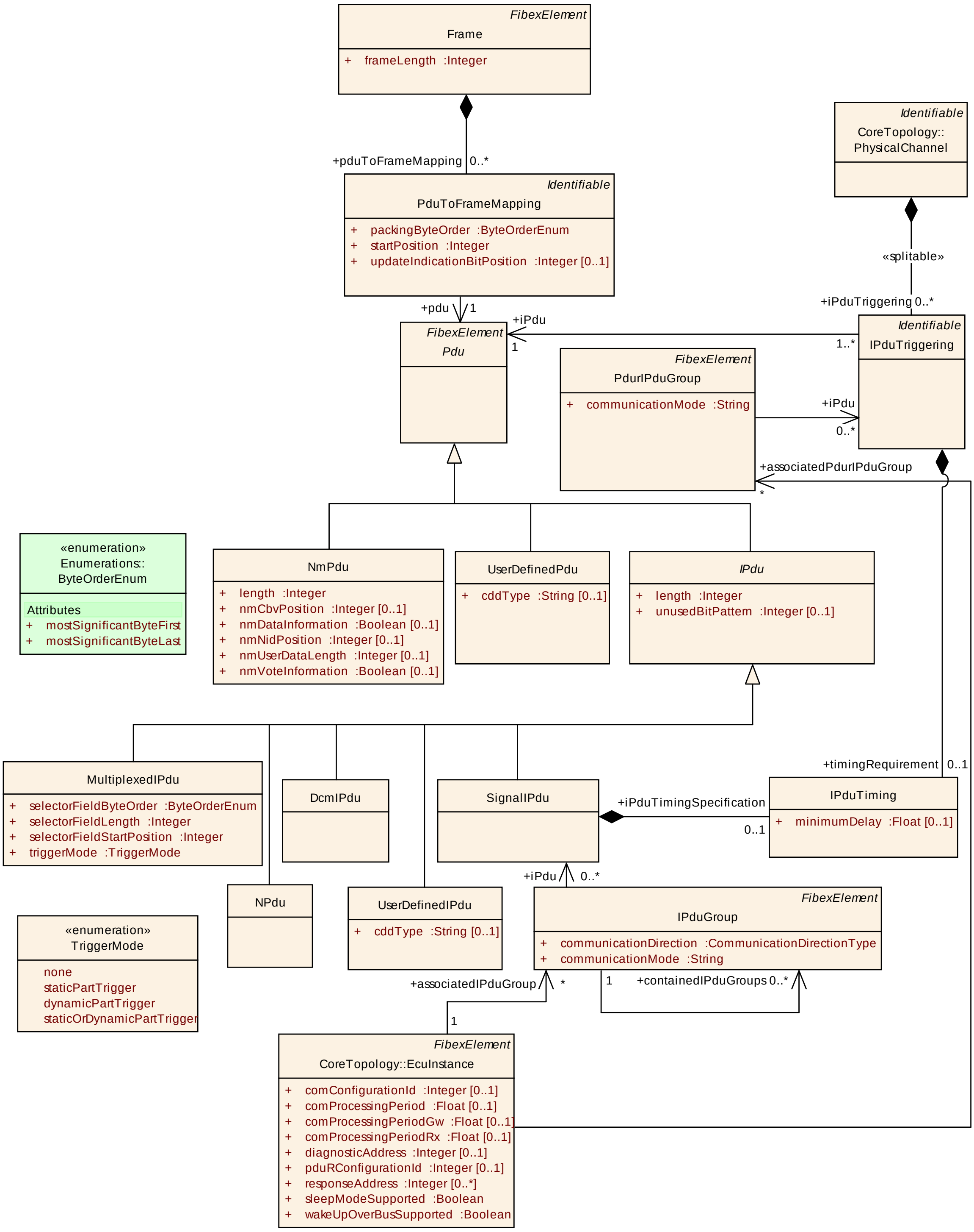

Figure 5.6: Pdus and the mapping into Frames (FibexCore: PDUOverview)

[3] There is one special gateway use case where a NPdu is routed by the Pdu Router. More details can be found in chapter 5.12.

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 SignalIPdu. This timing description can be used for the Configuration of COM Transmission Modes. The IPduTriggering describes on which channel the IPdu is transmitted. The element can also be used for the specification of timing requirements for FlexRay and Lin. This 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,FibexElement,Identifiable,PackageableElement			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Table 5.9: 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 Archtecture the NPdu is not a spezialisation of an IPdu. The NPdu is located under the IPdu to support the low-level routing of NPdu's. More details can be found in the NPdu class description.			
Base	ARObject,FibexElement,Identifiable,PackageableElement,Pdu			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
length	Integer	1	attr	The size of the IPDU in bits. The IPdu length of zero bytes is allowed.
unusedBit Pattern	Integer	0..1	attr	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.

Table 5.10: IPdu

Class	**SignalIPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Represents the I-PDU's handled by Com. The SignalIPdu 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.			
Base	ARObject,FibexElement,IPdu,Identifiable,PackageableElement,Pdu			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
iPduTiming Specification	IPduTiming	0..1	aggr	Timing specification for Com IPdus (Transmission Modes). This information is mandatory for the sender in a System Extract. This information may be omitted on receivers in a System Extract.
signalToPduMapping	ISignalToIPduMapping	*	aggr	Definition of SignalToIPduMappings included in the SignalIPdu.

Table 5.11: SignalIPdu

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 I-PDUs. 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.			
Base	ARObject,FibexElement,IPdu,Identifiable,PackageableElement,Pdu			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*

Table 5.12: NPdu

Class	**NmPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Network Management I-Pdu			
Base	ARObject,FibexElement,Identifiable,PackageableElement,Pdu			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
iSignalToIPduMapping	ISignalToIPduMapping	*	aggr	This optional aggregation is used to describe NmUserData that is transmitted in the NmPdu. The counting of the startPosition starts at the beginning of the NmPdu regardless whether Cbv or Nid are used.
length	Integer	1	attr	The size of the NmPDU in bits.
nmCbvPosition	Integer	0..1	attr	Defines the position of the control bit vector within the NM PDU (Bitpositon). If this attribute is not configured the Control Bit Vector is not used.
nmDataInformation	Boolean	0..1	attr	Defines if the Pdu contains NM Data. If the NmPdu does not aggregate any ISignalToIPduMappings it still may contain UserData that is set via Nm_SetUserData(). If the ISignalToIPduMapping exists then the nmDataInformation shall be ignored.
nmNidPosition	Integer	0..1	attr	Defines the bitposition of the source node identifier within the NM PDU. If this attribute is not configured or the attribute CommunicationCluster.nmNodeIdEnabled is set to false then the Node Identification is not used.
nmUserDataLength	Integer	0..1	attr	Defines the length in Bytes of the user data contained in the NM PDU.
nmVoteInformation	Boolean	0..1	attr	Defines if the Pdu contains NM Vote information.

Table 5.13: NmPdu

[constr_3035] User Data configuration in case NID/CBV are enabled ⌈If NID/CBV are enabled (`nmCbvPosition` and `nmNidPosition` are configured), there shall not be any user data configured at the position of the respective NID/CBV bytes. ⌋

[constr_3044] CBV configuration in case partial network is used ⌈ In case a partial network is used the control bit vector (CBV) shall be defined in Byte 0 of the `NmPdu`. ⌋

[constr_3069] Allowed `nmNidPosition` values ⌈ The value of `nmNidPosition` shall only be set to either bit 0 (byte 0) or bit 8 (byte 1). ⌋

[constr_3070] Allowed `nmCbvPosition` values ⌈ The value of `nmCbvPosition` shall only be set to either bit 0 (byte 0) or bit 8 (byte 1). ⌋

[constr_3071] Values of `nmCbvPosition` and `nmNidPosition` shall never have the same value ⌈ `nmCbvPosition` and `nmNidPosition` shall never have the same value. ⌋

Please note that in AUTOSAR only FrNm is able to send out `NmPdu`s with and without voting information:

[constr_3073] `nmVoteInformation` only valid for FrNm ⌈ The `nmVoteInformation` attribute is only valid for FrNm. ⌋

Class	**DcmIPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Represents the I-PDU's handled by Dcm.			
Base	ARObject,FibexElement,IPdu,Identifiable,PackageableElement,Pdu			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Table 5.14: 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, Xcp without the need of pdu routing) then this Pdu element shall be used to describe the communication. UserDefinedPdu is not suitable to be routed using the Pdu Router.			
Base	ARObject,FibexElement,Identifiable,PackageableElement,Pdu			
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 5.15: 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, Xcp with the need to be routed through the Pdu Router) then this IPdu element shall be used to describe the communication. UserDefinedIPdu is suitable to be routed using the Pdu Router.			
Base	ARObject,FibexElement,IPdu,Identifiable,PackageableElement,Pdu			
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 5.16: UserDefinedIPdu

Class	**IPduGroup**			
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 I-Pdu group contains either Com I-Pdus or I-Pdu groups. When an I-Pdu group containing one or more other I-Pdu groups is started the contained I-Pdu groups shall also be started. When an I-Pdu group containing one or more other I-Pdu groups is stopped the contained I-Pdu groups shall also be stopped. Only a two level hierarchy of I-Pdu groups is allowed. An I-Pdu group that is part of an I-Pdu group must not contain I-Pdu groups. In the COM SRS document it is stated that "every IPdu must belong to exactly 1 I-Pdu group." This is true from a dedicated ECUs point of view, however in the system description handling a number of ECUs several IPdu Groups may reference to the same SignalIPdu.			
Base	ARObject,FibexElement,Identifiable,PackageableElement			
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 Send or Receive).
communicationMode	String	1	attr	This attribute defines the use-case for this IPduGroup (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.
containedIPduGroups	IPduGroup	*	ref	An I-PDU group can be included in other I-Pdu groups.
iPdu	SignalIPdu	*	ref	Reference to a set of SignalIPdus, which are contained in the I-Pdu Group.

Table 5.17: IPduGroup

With the association between an IPduGroup and an EcuInstance it is possible to identify which IPduGroups are applicable for which CommunicationConnector/Ecu. Only top level IPduGroups shall be referenced by an EcuInstance. If an IPduGroup contains other IPduGroups than these contained IPduGroups shall not be referenced by the EcuInstance. Contained IPduGroups are associated to an EcuInstance via the top level IPduGroup.

Class	**PduToFrameMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	A PduToFrameMapping defines the composition of Pdus in each frame. 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.			
Base	ARObject,Identifiable			
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).
pdu	Pdu	1	ref	Reference to a I-Pdu, N-Pdu or NmPdu that is transmitted in the Frame.
startPosition	Integer	1	attr	This attribute describes the bitposition of a Pdu within a Frame. Please 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. The Pdus are byte aligned in a Frame and only the values 0, 8, 16, 24,... (for little endian) and 7, 15, 23, ... (for big endian) are allowed.

Attribute	Datatype	Mul.	Kind	Note
updateIndicationBitPosition	Integer	0..1	attr	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. 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 5.18: PduToFrameMapping

Class	IPduTiming			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	AUTOSAR COM provides the possibility to define two different TRANSMISSION MODES for each I-PDU. The Transmission Mode of an I-PDU that is valid at a specific point in time is selected using the values of the signals that are mapped to this I-PDU. For each I-PDU 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**
cyclicTiming	CyclicTiming	0..1	aggr	If the COM Transmission Mode is true the timing can be aggregated directly by the IPduTriggering. Additionally a Cyclic Timing can be defined as a Timing Requirement (for Lin, FlexRay). Timing Requirements are aggregated by the IPduTriggering/IPduTiming element.
eventControlledTiming	EventControlled Timing	0..1	aggr	If the COM Transmission Mode is true the timing can be aggregated directly by the IPduTiming. Additionally an EventControlledTiming can be defined as a Timing Requirement (for Lin, FlexRay). Timing Requirements are aggregated by the IPduTriggering/IPduTiming element.
minimumDelay	Float	0..1	attr	Minimum Delay in seconds between successive transmissions of this I-PDU, independent of the Transmission Mode.

Attribute	Datatype	Mul.	Kind	Note
requestControlledTiming	RequestControlledTiming	0..1	aggr	A RequestControlled Timing can be defined as a Timing Requirement. Timing Requirements are aggregated by the IPduTriggering/IPduTiming element.
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. In case only the TRUE transmission mode is used there is no need for the "TransmissionModeDeclaration" and its sub-structure.

Table 5.19: IPduTiming

Class	IPduTriggering			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	The IPduTriggering describes on which channel the IPdu is transmitted. 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			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
iPdu	Pdu	1	ref	Reference to the Pdu for which the IPdu triggering is defined. One IPdu can be triggered on different channels. The Pdu routing by the PduR is only allowed for IPdus and not for NmPdus and UserDefinedPdus. 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	This relationship specifies explicitly which IPdus are received/sent by the connected ECU on the connected channel. This reference shall be provided to every IPduPort on every ECU in the System which sends and/or receives the IPdu.

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
iSignalTrig gering	ISignalTriggerin g	*	ref	This reference provides the relationship to the ISignalTriggerings that are implemented by the IPduTriggering. The reference is optional since no ISignalTriggering can be defined for DCM and Multiplexed Pdus.
timingReq uirement	IPduTiming	0..1	aggr	Describes timing requirements on an I-PDU, handled by the bus interface, (Flexray or LIN). For CAN the timing information must be equal to the timing specification on a signal IPdu.

Table 5.20: IPduTriggering

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

Table 5.21: 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`. ⌋

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.			
Base	ARObject,FibexElement,Identifiable,PackageableElement			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
communic ationMode	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	IPduTriggering	*	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.

Table 5.22: PdurIPduGroup

AUTOSAR Pdu Router provides a mechanism of enabling/disabling of routing path groups (PdurIPduGroup). With the association between an `PdurIPduGroup` and an `EcuInstance` it is possible to identify which `IPduGroups` are applicable for which CommunicationConnector/Ecu.

5.3.1 EndToEndProtection for ISignalIPduGroups

It is possible to protect the inter-ECU data exchange of safety-related signal groups which are mapped into `SignalIPdus` using protection mechanisms provided by E2E Library. It is possible to protect several signal groups in one `SignalIPdu` using several `EndToEndProtectionISignalIPdu` elements.

The `E2EProtectionISignalIPdu` element refers to the `ISignal` (representing a signal group) that is to be protected and to the `SignalIPdu` that transmits the protected `ISignal`. The `dataOffset` in the `E2EProtectionISignalIPdu` element defines the starting position of the Array representation of the `ISignal`.

The information how the referenced `ISignal` 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 `SignalIPdu` (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 `EndToEndProtectionDataElementPrototype` shall be defined. For more details, see SWC Template [5].

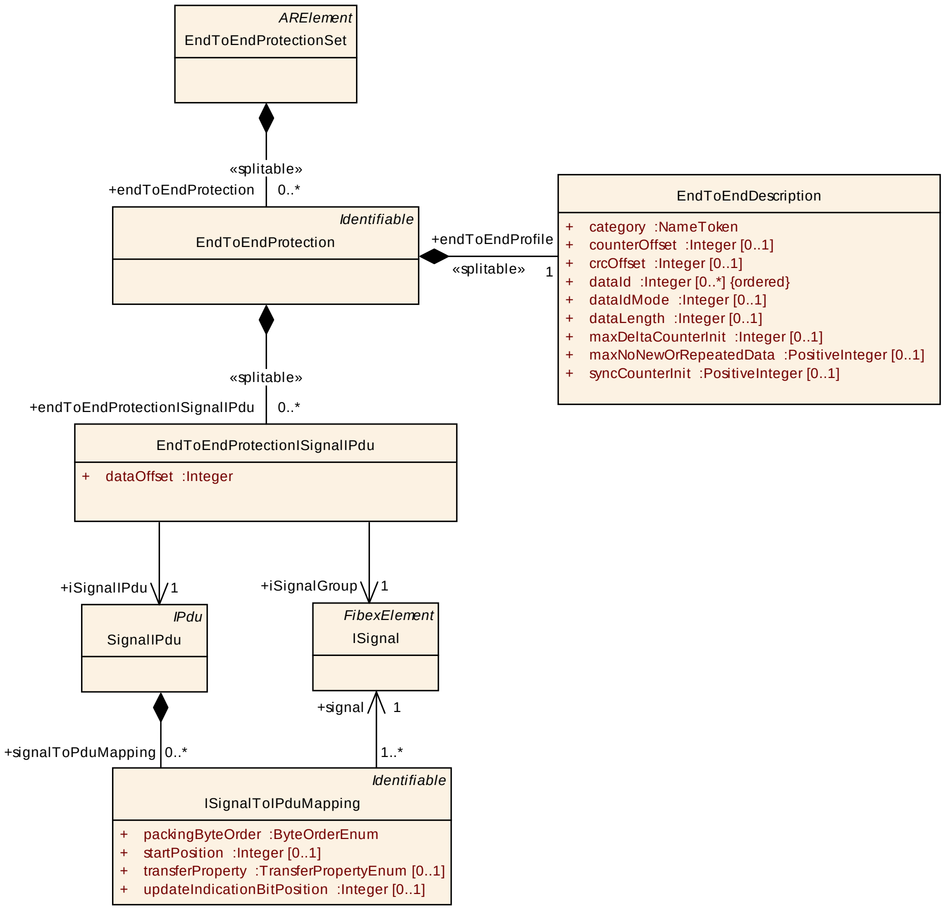

Figure 5.7: EndToEndProtection for COM IPdus

Class	**EndToEndProtectionSet**			
Package	M2::AUTOSARTemplates::SWComponentTemplate::EndToEndProtection			
Note	This represents a container for collection EndToEndProtectionInformation.			
Base	ARElement,ARObject,Identifiable,PackageableElement			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
endToEnd Protection	EndToEndProte ction	*	aggr	This is one particular EndToEndProtection. **Stereotypes:** atpSplitable

Table 5.23: 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			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
endToEnd Profile	EndToEndDesc ription	1	aggr	This represents the particular EndToEndDescription. **Stereotypes:** atpSplitable

Attribute	Datatype	Mul.	Kind	Note
endToEndProtectionDataElementPrototype	EndToEndProtectionDataElementPrototype	*	aggr	Defines to which DataElementPrototypes in the roles of one sender and one or more receivers this EndToEndprotection shall apply. In case several senders are defined to transport the data there may exist several EndToEndProtectionDataElementPrototype elements. It shall be possible to aggregate several EndToEndProtectiondataElementPrototype 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
endToEndProtectionISignalIPdu	EndToEndProtectionISignalIPdu	*	aggr	Defines to which ISignalIPdu - ISignal pair this EndToEndProtection shall apply. In case several ISignals are used to transport the data (e.g. fan-out in the RTE) there may exist several EndToEndProtectionISignalIPdu definitions. **Stereotypes:** atpSplitable

Table 5.24: EndToEndProtection

Class	**EndToEndProtectionISignalIPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::EndToEndProtection			
Note	It is possible to protect the inter-ECU data exchange of safety-related ISignal (refering to a SystemSignalGroup) at the level of COM IPdus using protection mechanisms provided by E2E Library. For each ISignal to be protected, a separate E2EProtectionISignalIPdu element must be created within the EndToEndProtectionSet. The E2EProtectionISignalIPdu element refers to the ISignal that is to be protected and to the ISignalIPdu that transmits the protected ISignal. The information how the referenced ISignal 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***
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	ISignal	1	ref	Reference to the ISignal (representing a signal group) that is to be protected.

Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
iSignalIPdu	SignalIPdu	1	ref	Reference to the SignalPdu that transmits the protected ISignal (group).

Table 5.25: EndToEndProtectionISignalIPdu

Class	**EndToEndProtectionDataElementPrototype**			
Package	M2::AUTOSARTemplates::SWComponentTemplate::EndToEndProtection			
Note	It is possible to protect the data exchanged between software components. For this purpose, for each communication to be protected, the user defines a separate EndToEndProtection (specifying a set of protection settings) and refers to a dataElement in the role of sender and to one or many dataElements in the role of receiver. For details, see End to End Library.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
receiver	DataElementPrototype	*	iref	This represnets the receiver. Note that 1:n communication is supported for this use case.
sender	DataElementPrototype	0..1	iref	This represents the sender. Can be optional if an ecu extract is provided and the sender is part of the extract.

Table 5.26: EndToEndProtectionDataElementPrototype

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	Integer	0..1	attr	Bit offset of Counter from the beginning of the Array representation of the Signal Group/DataElementPrototype (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

Attribute	Datatype	Mul.	Kind	Note
crcOffset	Integer	0..1	attr	Bit offset of CRC from the beginning of the Array representation of the Signal Group/DataElementPrototype (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
dataId	Integer	*	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	Integer	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	Integer	0..1	attr	This attribute represents the length of the Array representation of the Signal Group/DataElementPrototype including CRC and Counter in bits. **Tags:** xml.sequenceOffset=-80

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
maxDeltaCounterInit	Integer	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
maxNoNewOrRepeatedData	PositiveInteger	0..1	attr	The maximum amount of missing or repeated Data which the receiver does not expect to exceed under normal communication conditions.
syncCounterInit	PositiveInteger	0..1	attr	Number of Data required for validating the consistency of the counter that must be received with a valid counter (i.e. counter within the allowed lock-in range) after the detection of an unexpected behavior of a received counter.

Table 5.27: EndToEndDescription

The `maxDeltaCounterInit`, `maxNoNewOrRepeatedData` and `syncCounterInit` values can also be specified in the `ReceiverComSpec`. This allows the definition of receiver specific values. Values for `maxDeltaCounterInit`, `maxNoNewOrRepeatedData` and `syncCounterInit` that are defined in the `ReceiverComSpec` override the possible values in the `EndToEndDescription` class. More details can be found in the Software Component Template specification [5].

The supported E2E profiles (possible values of category in EndToEndDescription) are described in the Software Component Template [5] and the End to End Library [17].

5.4 Frames

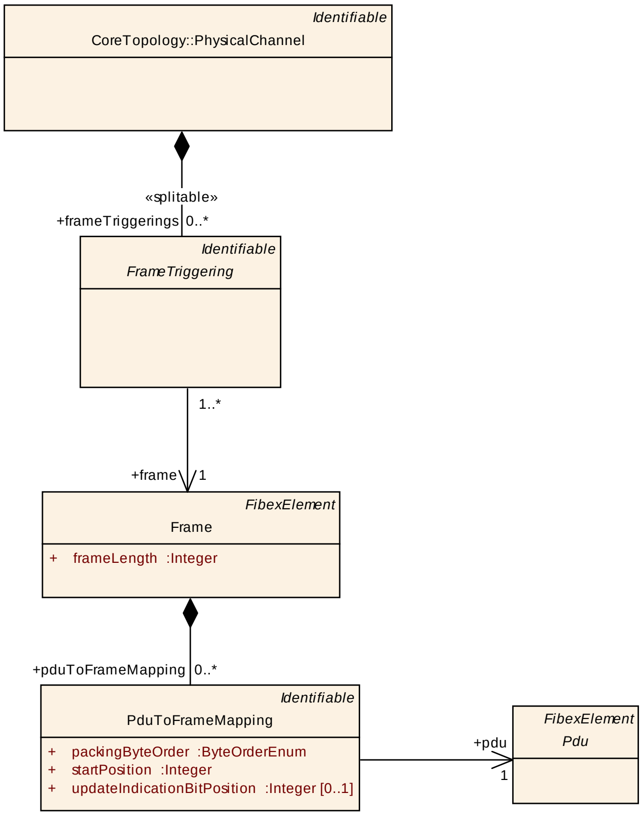

Figure 5.8: Frame Overview (FibexCore: FrameOverview)

`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.

Class	Frame			
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,FibexElement,Identifiable,PackageableElement			
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.

Table 5.28: 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 Frame Triggerings exist on more than one channel of the same cluster the fan-out/in is handled by the Bus interface.			
Base	ARObject,Identifiable			
Attribute	Datatype	Mul.	Kind	Note
frame	Frame	1	ref	One frame can be triggered on different channels. If a frame has no frame triggering, it wont be sent at all. A frame triggering has assigned exactly one frame, which it triggers.
framePort	FramePort	*	ref	This reference allows to specify explicitly which Frame is received/sent by the connected ECU on the connected channel. This reference shall be provided to every FramePort on every ECU in the System which sends and/or receives the Frame.
iPduTriggering	IPduTriggering	*	ref	This reference provides the relationship to the IPduTriggerings that are implemented by the FrameTriggering. The reference is optional since no IPduTriggering can be defined for NmPdus.

Table 5.29: FrameTriggering

5.5 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.

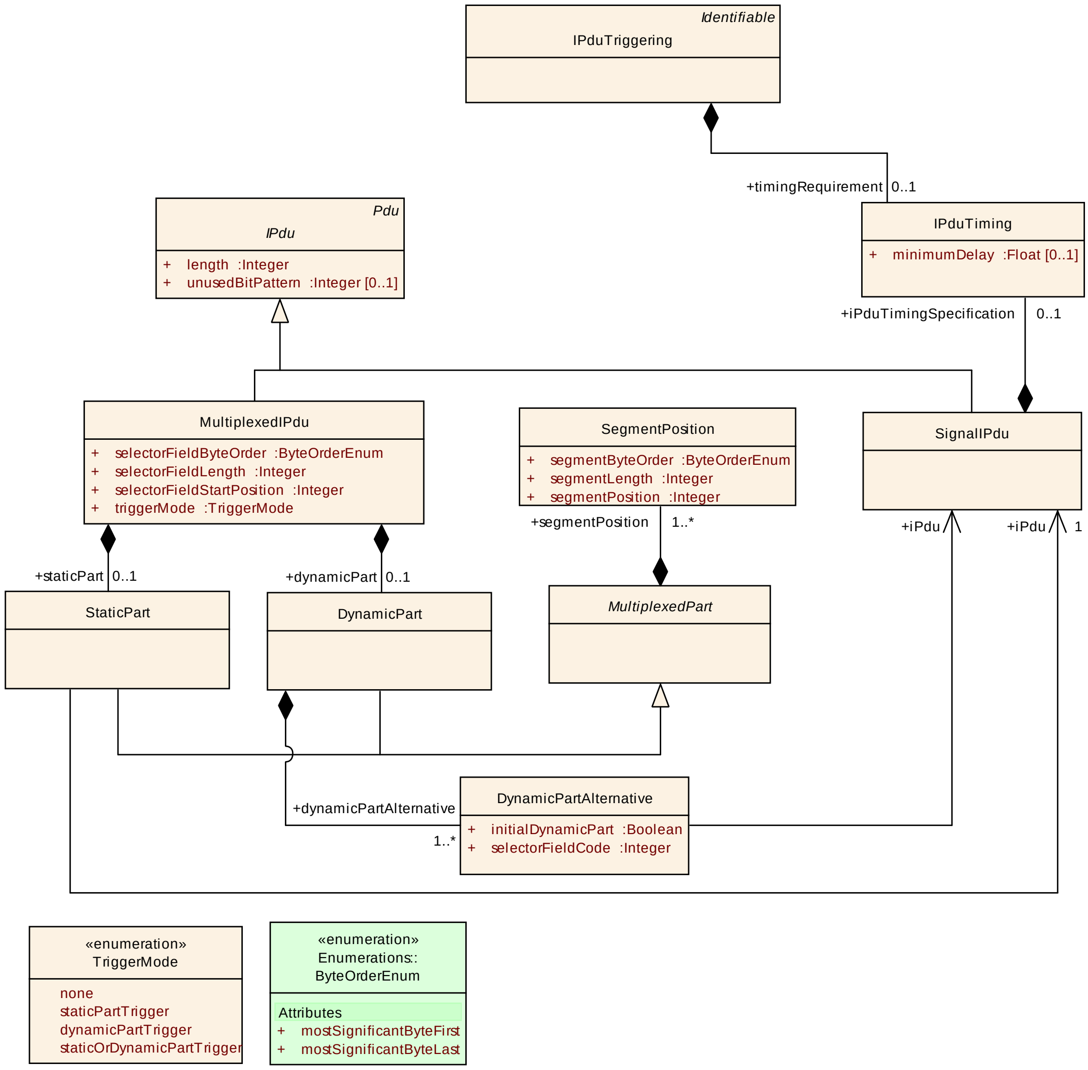

Figure 5.9: I-Pdu Multiplexer (FibexCore: IPDUMultiplexerOverview)

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.			
Base	ARObject,FibexElement,IPdu,Identifiable,PackageableElement,Pdu			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
dynamicPart	DynamicPart	0..1	aggr	According to the value of the selector field some parts of the MultiplexedIPdu have a different layout. In a complete System Description a MultiplexedIPdu shall contain a DynamicPart. The following use cases support the multiplicity to be 0..1: • If a MultiplexedIPdu 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 MulitplexedIPdu doesn't need to be described in the System Extract/Ecu Extract. • If a MultiplexedIPdu is received by an ECU which is only interested in the static part of the MultiplexedIPdu then the dynamicPart does not need to be described in the System Extract/Ecu Extract.
selectorFieldByteOrder	ByteOrderEnum	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.
selectorFieldLength	Integer	1	attr	The size in bits of the selector field shall be configurable in a range of one bit and eight bits.

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
selectorFieldStartPosition	Integer	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.
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.
triggerMode	TriggerMode	1	attr	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.

Table 5.30: MultiplexedIPdu

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 5.31: TriggerMode

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	SignalIPdu	1	ref	Reference to a Com IPdu which is routed to the IPduM module and is combined to a multiplexedPdu.

Table 5.32: 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 5.33: 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	SignalIPdu	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 dynamic part of the IPdu.

Table 5.34: 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 5.35: 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.

Table 5.36: SegmentPosition

Figure 5.10 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 5.10 the dynamic Part is segmented into two parts.

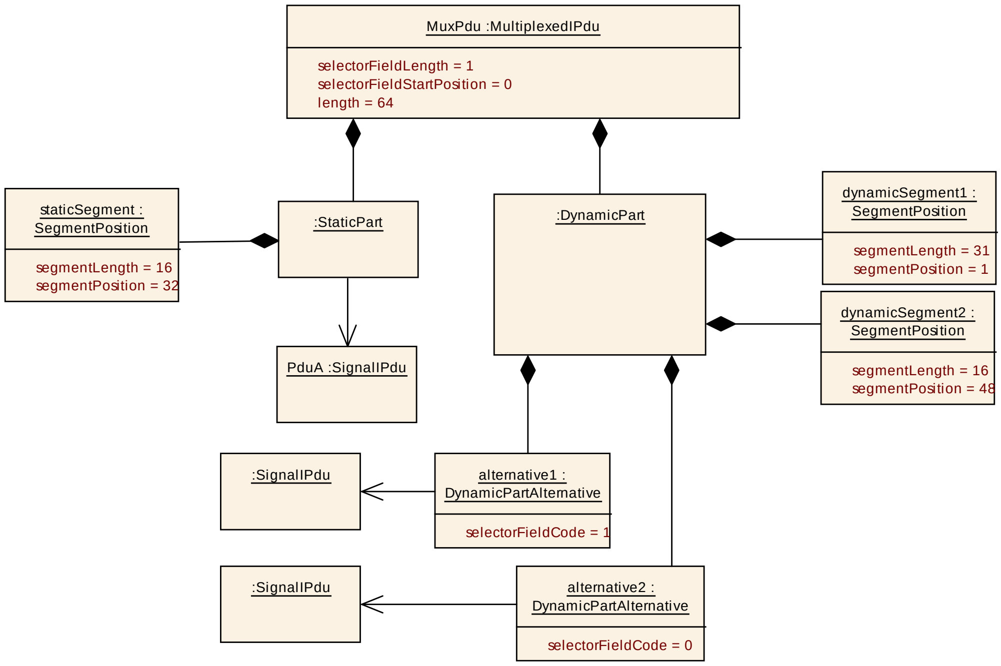

Figure 5.10: I-Pdu Multiplexer Example

5.6 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.

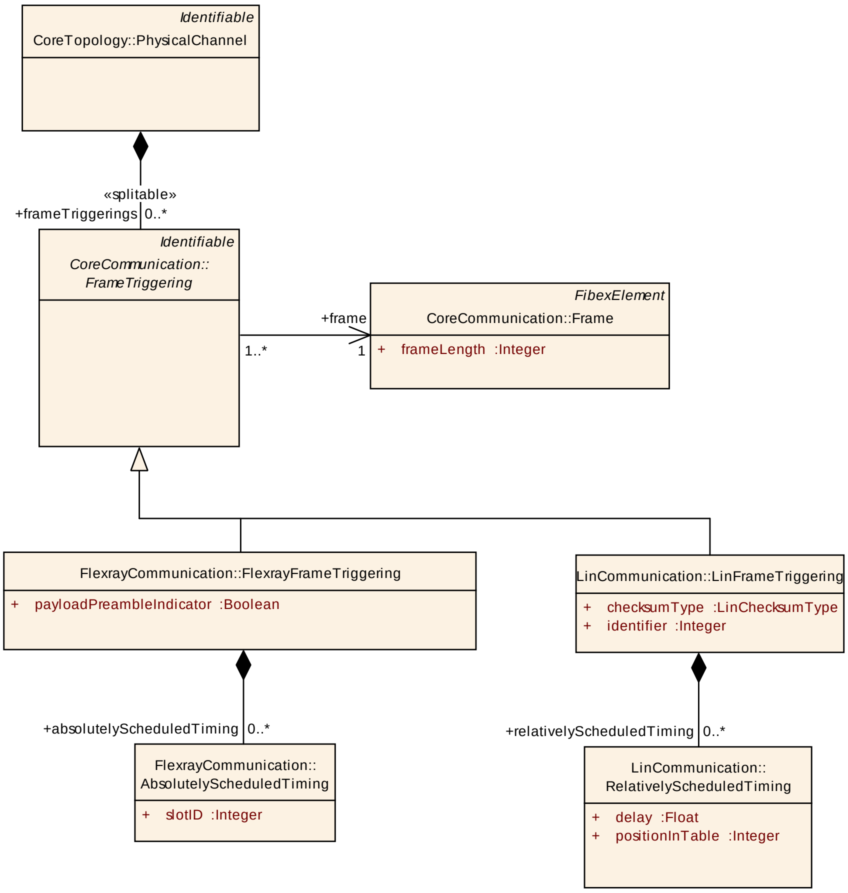

Figure 5.11: Frame Triggering

In FlexRay each frame is identified by its slot id and communication cycle. The `AbsolutelyScheduledTiming` is described in chapter 5.7. Schedule tables organize the Timings of the frames for LIN. This special type of timing is described by a `RelativelyScheduledTiming` (chapter 5.8).

5.7 FlexRay specific Frame Timing 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 `AbsolutelyScheduledTiming`.

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 `AbsolutelyScheduledTiming` 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 AbsolutelyScheduledTiming.

If the behavior of a FlexRay frame is cyclic or event triggered, a timing requirement can be specified in the `IPduTriggering`. This timing requirement must be fulfilled by the timing specification on the frame.

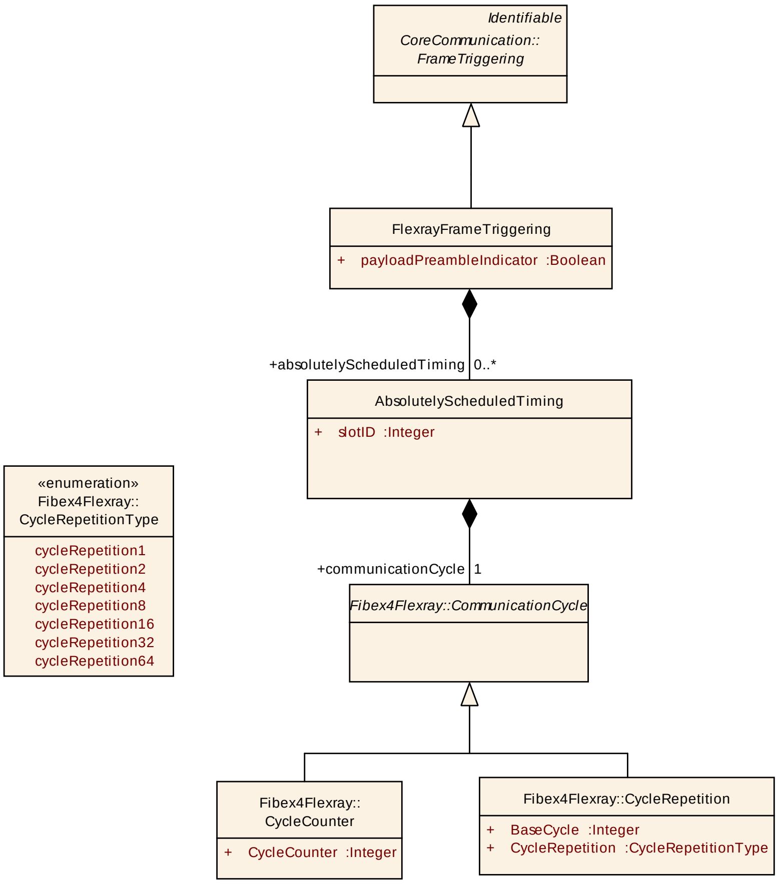

Figure 5.12: Absolutely Scheduled Timing (Fibex4FlexRay:AbsolutelyScheduledTiming)

Class	**FlexrayFrameTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::Flexray Communication			
Note	FlexRay specific attributes to the FrameTriggering			
Base	ARObject,FrameTriggering,Identifiable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
absolutely Scheduled Timing	AbsolutelySche duledTiming	*	aggr	Specification of a sending behaviour where the exact time for the frames transmission is guaranteed.
payloadPr eambleIndi cator	Boolean	1	attr	Switching the Payload Preamble bit.

Table 5.37: FlexrayFrameTriggering

Class	**AbsolutelyScheduledTiming**			
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	Integer	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 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 5.38: AbsolutelyScheduledTiming

Class	**CommunicationCycle (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray			
Note	The communication cycle where the frame is sent.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*

Table 5.39: CommunicationCycle

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

Class	**CycleCounter**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray			
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 5.40: CycleCounter

Class	**CycleRepetition**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray			
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 5.41: CycleRepetition

Enumeration	**CycleRepetitionType**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray
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	cycleRepetition value="1"
cycleRepetition16	cycleRepetition value="16"
cycleRepetition2	cycleRepetition value="2"
cycleRepetition32	cycleRepetition value="32"
cycleRepetition4	cycleRepetition value="4"
cycleRepetition64	cycleRepetition value="64"
cycleRepetition8	cycleRepetition value="8"

Table 5.42: CycleRepetitionType

5.8 Lin specific Frame Timing 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 Frames.

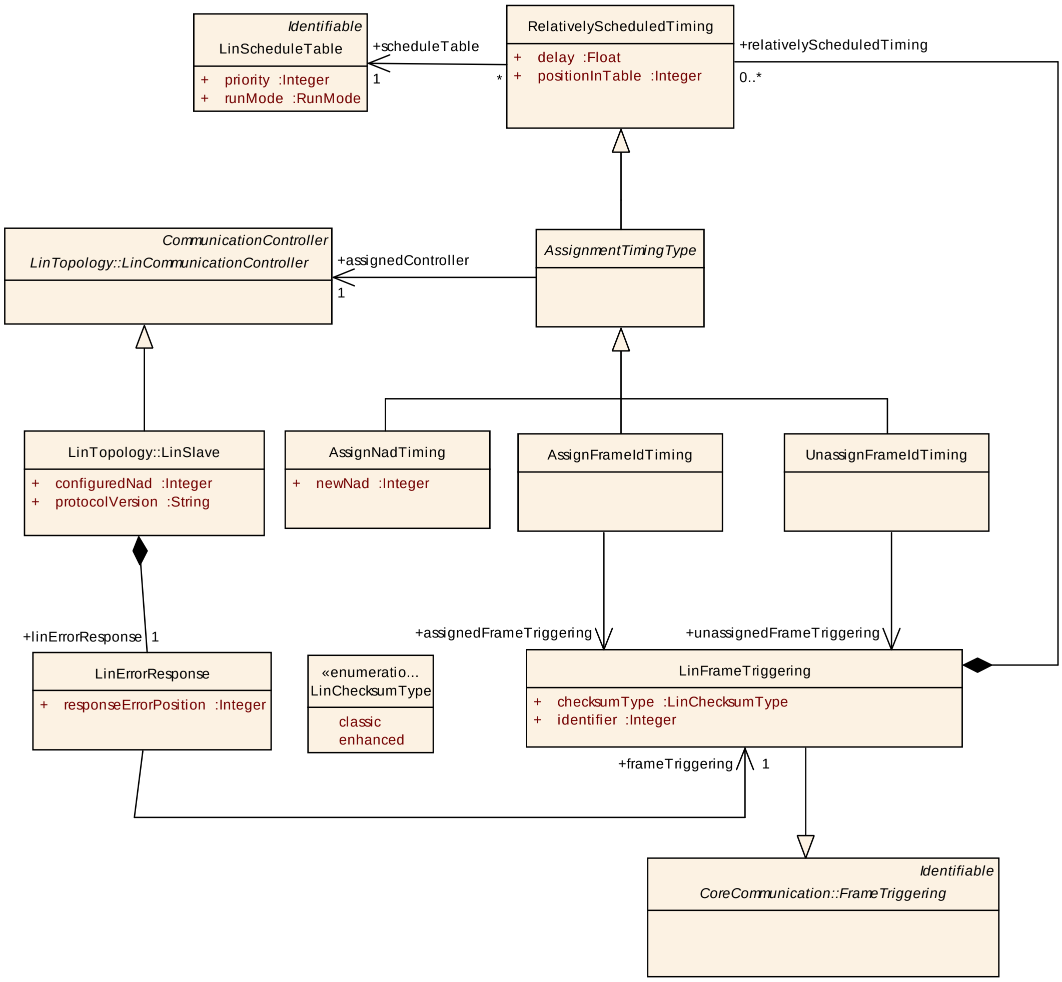

Figure 5.13: Relatively Scheduled and Assignment Timing (Fibex4Lin:AssignmentTiming)

In order to describe the LIN Communication the `RelativelyScheduledTiming` element is defined. 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.

Class	**LinFrameTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Lin specific attributes to the FrameTriggering			
Base	ARObject,FrameTriggering,Identifiable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
checksum Type	LinChecksumType	1	attr	Type of checksum that the frame is using.

Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
identifier	Integer	1	attr	To describe a frames identifier on the communication system, usualy with a fixed identifierValue.
relativelyScheduledTiming	RelativelyScheduledTiming	*	aggr	Specification of a sending behaviour where the transmission order is predefined.

Table 5.43: 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 5.44: LinChecksumType

Class	**RelativelyScheduledTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Specification of a sending behavior where the transmission order is predefined, e.g. used on LIN buses			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
delay	Float	1	attr	Relative delay between this frame and the start of the successor frame in the schedule table in seconds.
positionInTable	Integer	1	attr	Relative position of the frame described by this timing in the schedule table
scheduleTable	LinScheduleTable	1	ref	The master task transmits frame headers based on a schedule table. The master application may use different schedule tables and select among them.

Table 5.45: RelativelyScheduledTiming

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			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*

Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
priority	Integer	1	attr	Priority of the schedule table. The priority is used in the schedule table manager. The RUN_ONCE run mode schedules shall not have equal priority. Priority 0 is reserved for the NULL_SCHEDULE. Priority 255 is reserved for the RUN_CONTINUOUS run mode.
runMode	RunMode	1	attr	The schedule table can be executed in two different modes.

Table 5.46: LinScheduleTable

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

Table 5.47: RunMode

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/FIBEX the `AssignmentTiming` is introduced as a LIN specific extension. For the assignment a relation from `AssignmentTiming` to `CommunicationController` is needed. An additional relation to `FrameTriggering` is used for the assignment of the LIN identifier.

The assignment of node addresses (`AssignNadTiming`) is done in a slightly different way. Here only a reference to the `CommunicationController` is used.

Class	**AssignmentTimingType (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	A LIN specific extension of the common RelativelyScheduledTiming The extension describes the LIN specific assignment frames.			
Base	ARObject,RelativelyScheduledTiming			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
assignedController	LinCommunicationController	1	ref	The LIN slaves controller who is target of this assignment.

Table 5.48: AssignmentTimingType

Class	**UnassignFrameIdTiming**			
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,AssignmentTimingType,RelativelyScheduledTiming			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
unassignedFrameTriggering	LinFrameTriggering	1	ref	The frame whose identifier is reset by this assignment.

Table 5.49: UnassignFrameIdTiming

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

Table 5.50: AssignFrameIdTiming

Class	**AssignNadTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Schedule entry for an Assign NAD master request.			
Base	ARObject,AssignmentTimingType,RelativelyScheduledTiming			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
newNad	Integer	1	attr	The newly assigned NAD value (valid range 1..126)

Table 5.51: AssignNadTiming

With the FreeFormat a scheduling of fixed data content within a diagnostic frame is defined. For that specification `DataTiming` is introduced. More informations can be found in FIBEX [7]. In order to be consistent with the rest of the communication configuration, it is required that the diagnostic Lin Frames (Master Request Frame, Slave Request Frame) are explicitly modeled as `Frame` elements. `LinFrameTriggerings` dealing with diagnostic Frames thus reference this diagnostic frames. The defined diagnostic Frames does not contain `PduToFrameMappings`.

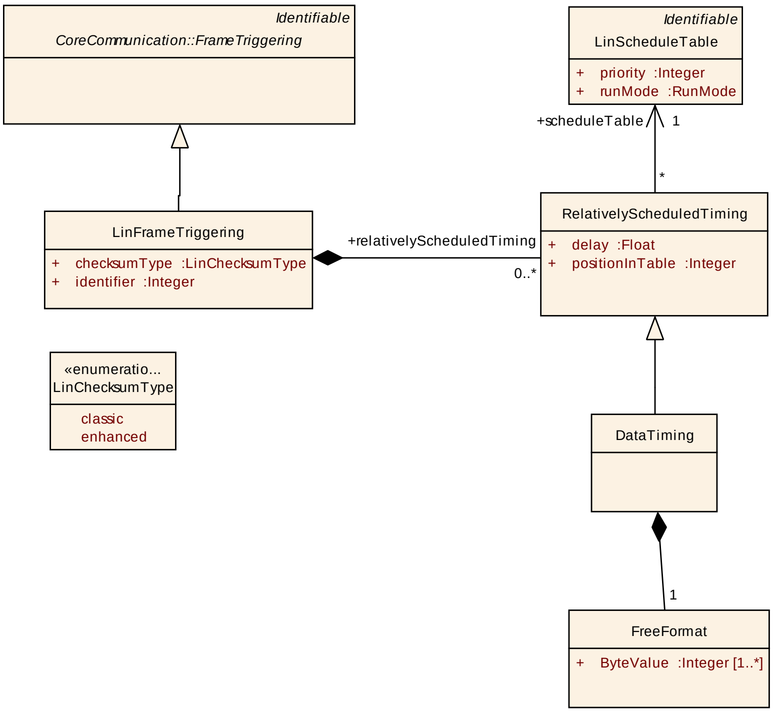

Figure 5.14: Free Format (Fibex4Lin:DataTiming)

Class	**DataTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	A LIN specific extension of the common RelativelyScheduledTiming. The extension maps the LIN specific free data frames into SystemTemplate. The base type keeps scheduling for those free data frames applicable.			
Base	ARObject,RelativelyScheduledTiming			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
freeFormat	FreeFormat	1	aggr	

Table 5.52: DataTiming

Class	**FreeFormat**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Representing freely defined data.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
ByteValue	Integer	1..*	attr	The integer Value of a freely defined data byte.

Table 5.53: FreeFormat

In LIN there are event triggered frames and sporadic frames. Both of them are abstract elements that represent a collection of unconditional frames. In System Template/-FIBEX that is described by the hierarchical link from a Frame (Substitution) to itself. Note that this is only needed for event triggered frames and sporadic frames and, therefore, is limited to two levels of hierarchy.

Sporadic frames and event triggered frames refer to a set of frames that may be sent alternatively within one time slot in a schedule.

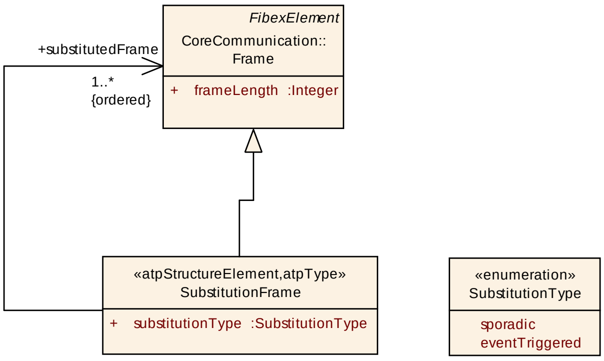

Figure 5.15: Substitution Frame (Fibex4Lin:SubstitutionFrame)

Class	**SubstitutionFrame**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	A LIN specific extension of the common FRAME to enable the usual frame handling of a placeholder frame that is substituted at runtime. Substitution frame must not declare signal instances nor multiplexers.			
Base	ARObject,FibexElement,Frame,Identifiable,PackageableElement			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
substituted Frame	Frame	1..*	ref	Collecting the frames that are substituted by the refering one. This reference is ordered. The order is used to describe the priority (Configuration parameter LinIfFramePriority). The first listed Substitution Frame has the highest priority.
substitutio nType	SubstitutionTyp e	1	attr	The type of substitution. Substitution frames can either be used for event triggered or for sporadic frames.

Table 5.54: SubstitutionFrame

Enumeration	**SubstitutionType**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication
Note	The type of substitution. Substitution frames can either be used for event triggered or for sporadic frames.
Literal	**Description**
eventTrig-gered	Eventtriggered Frame
sporadic	Sporadic Frame

Table 5.55: SubstitutionType

5.9 Can specific description

This chapter describes additions to the CAN definition of Frames.

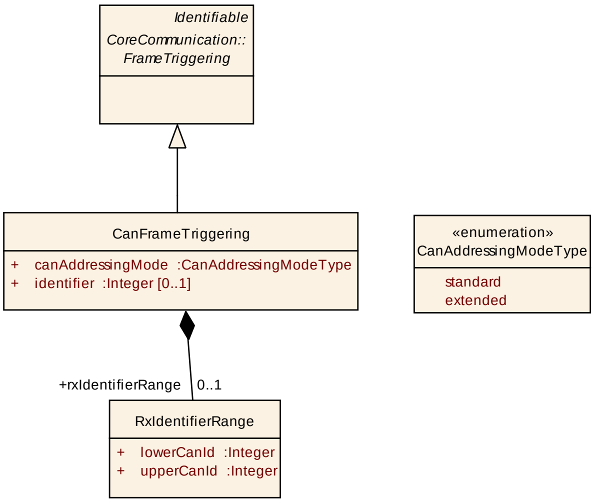

Figure 5.16: CanFrameTriggering (Fibex4Can:CanCommunication)

Class	**CanFrameTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanCommunication			
Note	CAN specific attributes to the FrameTriggering			
Base	ARObject,FrameTriggering,Identifiable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
canAddressingMode	CanAddressingModeType	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.
identifier	Integer	0..1	attr	To describe a frames identifier on the communication system, usualy 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.
rxIdentifierRange	RxIdentifierRange	0..1	aggr	Optional definition of a CanId range.

Table 5.56: CanFrameTriggering

Enumeration	**CanAddressingModeType**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanCommunication
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 5.57: 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 attributes "nmLowerCanId" and "nmUpperCanId" of "CanCluster". For backward compatibility reasons this redundancy shall be preserved and both shall be defined.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
lowerCanId	Integer	1	attr	This attribute can be used together with the upperCanId attribute to define a range of CanIds.
upperCanId	Integer	1	attr	This attribute can be used together with the lowerCanId attribute to define a range of CanIds.

Table 5.58: RxIdentifierRange

5.10 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` to decide which of the two modes is selected. It is possible to switch between the transmission modes during runtime.

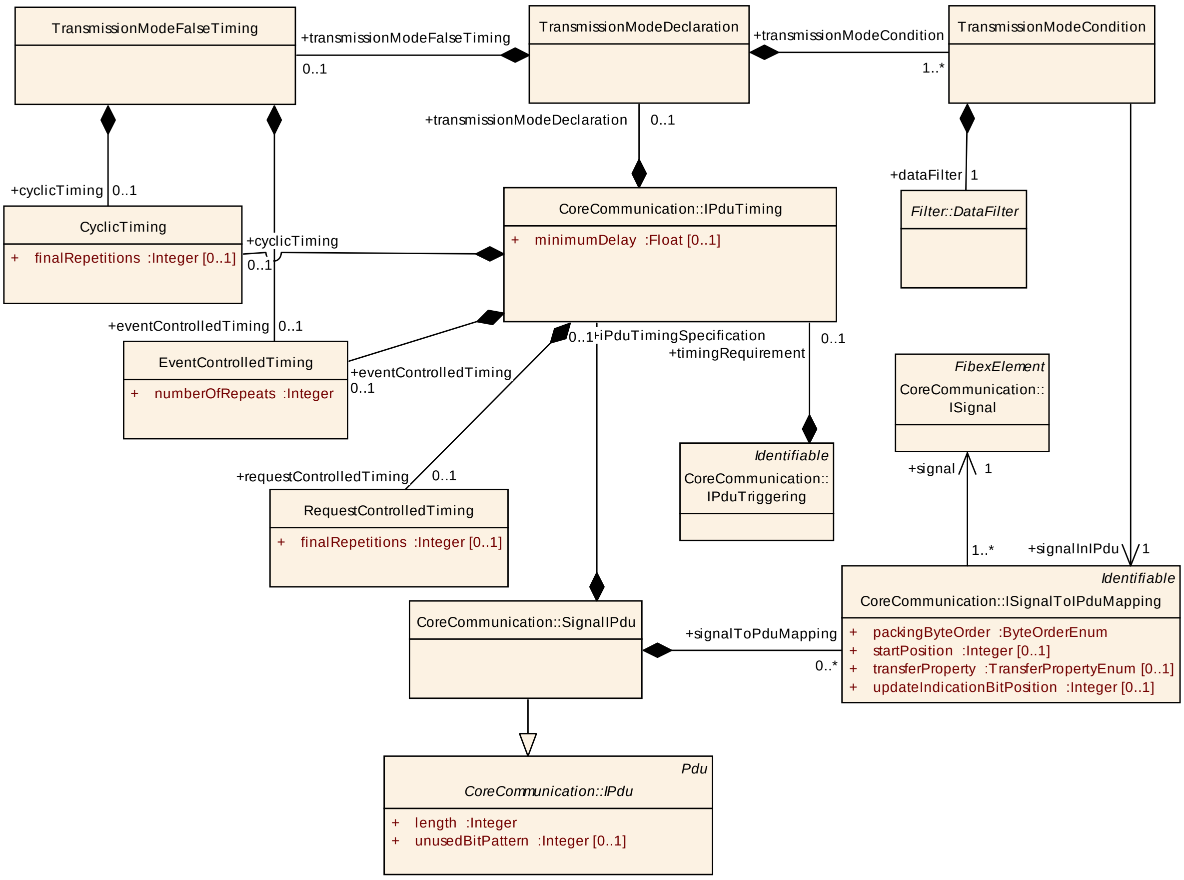

Figure 5.17: IPdu Timing

The System Template provides the possibility to attach a condition to each signal within an I-PDU. Each `TransmissionModeCondition` contains a reference to a signal and an assigned filter. The filter condition is used for the selection of the transmission mode. 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. More details can be found in the COM Specification [18].

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 5.59: COM Transmission Modes

If the transmission Mode is "False" the timing is described by the `transmissionModeFalseTiming` class. If the COM Transmission Mode is "True" the timing is directly aggregated by the IPduTriggering element. The available COM Transmission Mode Timings can be described by the `CyclicTiming` and `EventControlledTiming` elements (see Table 5.59).

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. In case only the TRUE transmission mode is used there is no need for the "TransmissionModeDeclaration" and its sub-structure.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
transmissionModeCondition	TransmissionModeCondition	1..*	aggr	The Transmission Mode Selector evaluates the conditions for a subset of signals and decides which transmission mode should be used.
transmissionModeFalseTiming	TransmissionModeFalseTiming	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.

Table 5.60: 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
signalInIPdu	ISignalToIPduMapping	1	ref	Reference to a signal to which a condition is attached.

Table 5.61: TransmissionModeCondition

Class	**TransmissionModeFalseTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Note	If the COM Transmission Mode is false the timing is aggregated by the TransmissionModeIFalseTiming element. If the COM Transmission Mode is true the timing is aggregated by the SignalIPdu/IPduTiming element. 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	EventControlled Timing	0..1	aggr	Direct Transmission Mode.

Table 5.62: TransmissionModeFalseTiming

Class	**DataFilter (abstract)**			
Package	M2::AUTOSARTemplates::CommonStructure::Filter			
Note	Base class for data filters.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Table 5.63: DataFilter

The following Data Filters are supported by AUTOSAR:

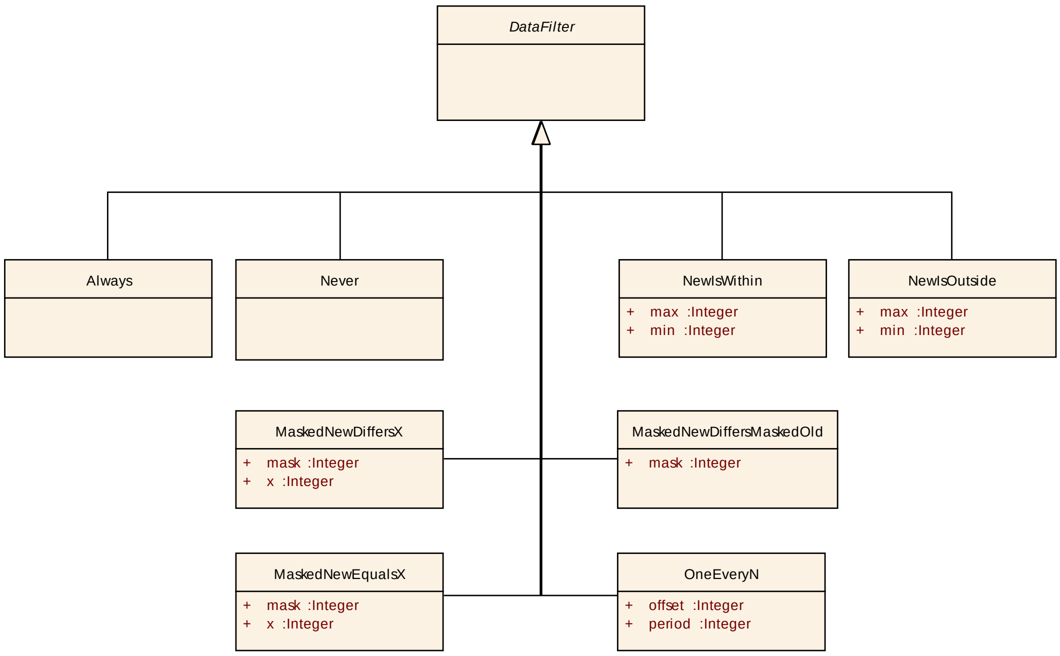

Figure 5.18: Data Filter

Class	**Always**			
Package	M2::AUTOSARTemplates::CommonStructure::Filter			
Note	No filtering is performed so that the message always passes.			
Base	ARObject,DataFilter			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Table 5.64: Always

Class	**Never**			
Package	M2::AUTOSARTemplates::CommonStructure::Filter			
Note	The filter removes all messages.			
Base	ARObject,DataFilter			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Table 5.65: Never

Class	**NewIsWithin**			
Package	M2::AUTOSARTemplates::CommonStructure::Filter			
Note	Pass a message if its value is within a predefined boundary. min <= new_value <= max			
Base	ARObject,DataFilter			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
max	Integer	1	attr	Value to specify the upper boundary
min	Integer	1	attr	Value to specify the lower boundary

Table 5.66: NewIsWithin

Class	**MaskedNewDiffersX**			
Package	M2::AUTOSARTemplates::CommonStructure::Filter			
Note	Pass messages whose masked value is not equal to a specific value x (new_value&mask) != x new_value: current value of the message			
Base	ARObject,DataFilter			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
mask	Integer	1	attr	mask for the new Value
x	Integer	1	attr	Value to compare with

Table 5.67: MaskedNewDiffersX

Class	**MaskedNewEqualsX**			
Package	M2::AUTOSARTemplates::CommonStructure::Filter			
Note	Pass messages whose masked value is equal to a specific value x (new_value&mask) == x new_value: current value of the message			
Base	ARObject,DataFilter			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
mask	Integer	1	attr	mask for the new Value
x	Integer	1	attr	Value to compare with

Table 5.68: MaskedNewEqualsX

Class	**MaskedNewDiffersMaskedOld**			
Package	M2::AUTOSARTemplates::CommonStructure::Filter			
Note	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 (initialised with the initial value of the message, updated with new_value if the new message value is not filtered out)			
Base	ARObject,DataFilter			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
mask	Integer	1	attr	mask for old and new value

Table 5.69: MaskedNewDiffersMaskedOld

Class	**OneEveryN**			
Package	M2::AUTOSARTemplates::CommonStructure::Filter			
Note	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).			
Base	ARObject,DataFilter			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
offset	Integer	1	attr	specifies the initial number of messages to occur before the first message is passed
period	Integer	1	attr	specifies number of messages to occur before the message is passed again

Table 5.70: OneEveryN

The IPduTriggering can be used for the specification of timing requirements for FlexRay and Lin. This timing requirements needs to be fulfilled by the timing specification on the Frame. The timing requirements (`CyclicTiming`, `EventControlledTiming`, `RequestControlledTiming`) are directly aggregated by the IPduTriggering element.

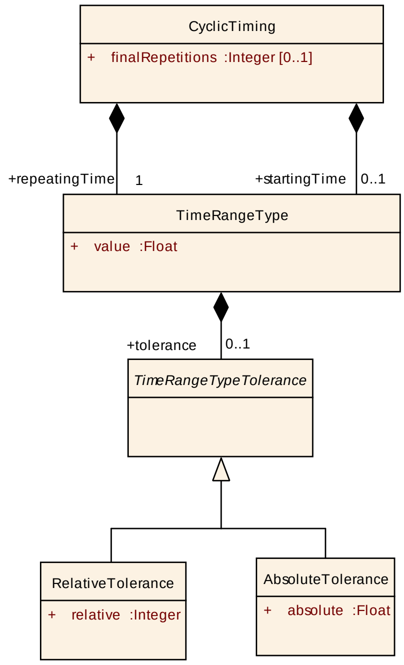

Figure 5.19: Cyclic Timing

Class	**CyclicTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Note	Specification of a cyclic sending behavior.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
finalRepetitions	Integer	0..1	attr	Number of repetitions the pdu is sent from the moment the stop condition has been met
repeatingTime	TimeRangeType	1	aggr	Specification of the repeating cycle.
startingTime	TimeRangeType	0..1	aggr	Specification of the time that is needed before the pdu can be sent the first time.

Table 5.71: CyclicTiming

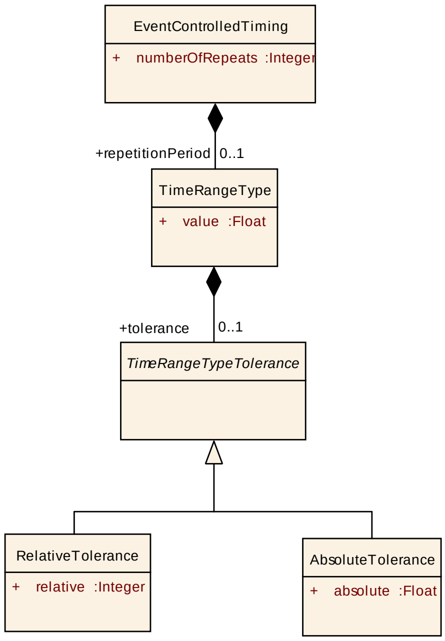

Figure 5.20: EventControlled Timing

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***
numberOfRepeats	Integer	1	attr	Defines the number of repetitions for the Direct/N-Times transmission mode and the event driven part of Mixed transmission mode.
repetitionPeriod	TimeRangeType	0..1	aggr	If the EventControlledTiming is aggregated by the IPduTiming the repetitionPeriod specifies the time in seconds that elapses before the pdu can be sent the next time (Minimum repeat gap between two pdus). If the EventControlledTiming is aggregated by the SignalTriggering the repetitionPeriod specifies the time in seconds that elapses before the signal can be sent the next time (Minimum repeat gap between two signals).

Table 5.72: EventControlledTiming

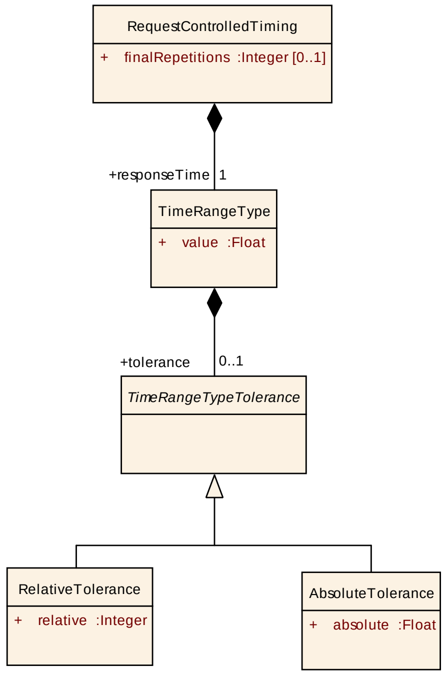

Figure 5.21: RequestControlled Timing

Class	**RequestControlledTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Note	Specification of a request driven sending behavior. Semantics of this communication mechanism is that basic software stores values but does not send it out until a frame requesting the information is received.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
finalRepetitions	Integer	0..1	attr	Number of repetitions the frame is sent for a single request
responseTime	TimeRangeType	1	aggr	Specification of the time that is needed before the frame can be sent after the requests arrival

Table 5.73: RequestControlledTiming

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	TimeRangeTypeTolerance	0..1	aggr	
value	Float	1	attr	Average value of a date (in seconds)

Table 5.74: 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 5.75: RelativeTolerance

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

Table 5.76: AbsoluteTolerance

5.11 Signal Timing

On the signal level only timing requirements can be specified. The final timing scheduling must be specified in the `IPduTiming` or `FrameTriggering`.

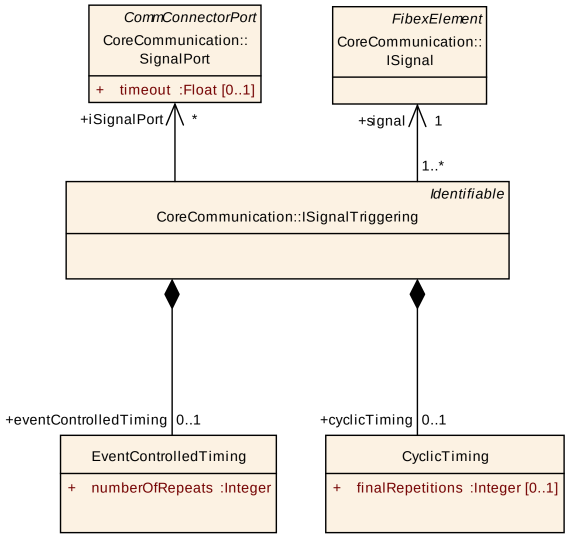

Figure 5.22: Signal Triggering

Class	**ISignalTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	A ISignalTriggering defines the manner of triggering of a ISignal on the channel, on which it is sent. ISignalTriggering should only be used for defining timing constraints. Com does not know of signals related to a specific cluster or channel.			
Base	ARObject,Identifiable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
cyclicTiming	CyclicTiming	0..1	aggr	Specification of a cyclic sending behavior.
eventControlledTiming	EventControlledTiming	0..1	aggr	Specification of a event driven sending behavior.
iSignalPort	SignalPort	*	ref	This relationship allows to specify explicitly which ISignals are received/sent by the connected ECU on the connected channel. This reference shall be provided to every SignalPort on every ECU in the System which sends and/or receives the Signal.
signal	ISignal	1	ref	Reference to the ISignal for which the ISignalTriggering is defined.

Table 5.77: ISignalTriggering

5.12 Transport Layer

In AUTOSAR, the Transport Layer has two main purposes: The segmentation and re-assembly 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. As of AUTOSAR Release 3.0 the usage of the Transport Layer is restricted to the Diagnostic Communication Manager. The usage for mapping long COM PDUs onto the Transport Layer is planned for Release 4.0.

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 and LIN. However, all models are embedded into the communication model: They use specialized classes of `TpChannel` as a root element into the TP configuration. All Transport Layers will take `IPdus` as input elements, which will be transferred in the form of one or more `NPdus`.

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	**TpChannel (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	A Transport Protocol channel.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Table 5.78: TpChannel

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 I-PDUs. 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.			
Base	ARObject,FibexElement,IPdu,Identifiable,PackageableElement,Pdu			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*

Table 5.79: NPdu

Class	**TpNode (abstract)**			
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			
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 5.80: TpNode

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

Table 5.81: 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
```

5.12.1 FlexRay Transport Layer

This section describes a Non-ISO FlexRay TP protocol that is supported by AUTOSAR in Rel. 3.x. Additionally the FlexRay ISO 10681-2 Transport Protocol is supported by AUTOSAR (see section 5.12.2).

The Non-ISO FlexRay Transport Layer supports multiple sessions, i.e. multiple segmented transfers can be handled at the same time.

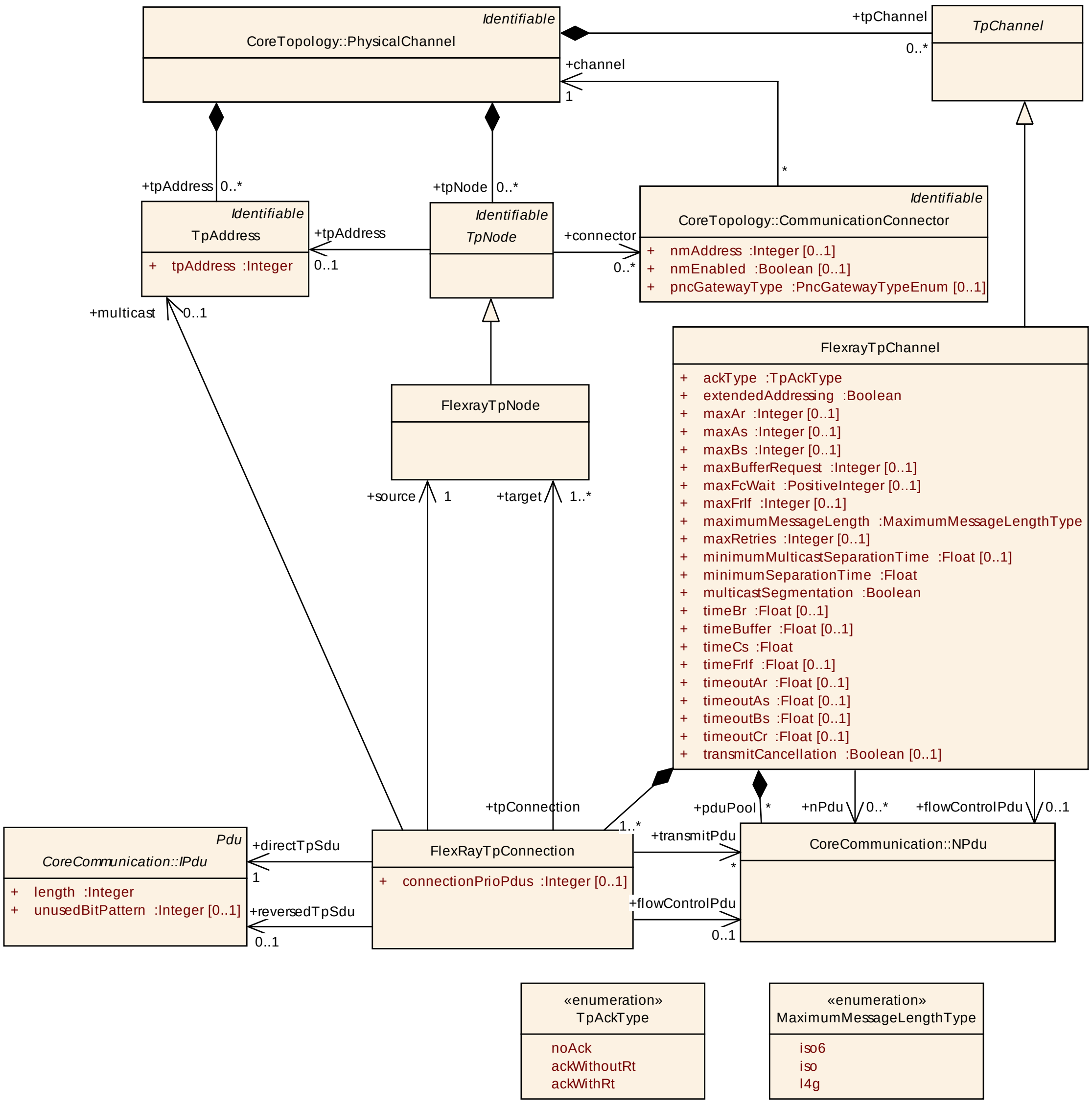

Figure 5.23: FlexRay Transport Layer Configuration (TransportProtocols: FlexRayTransportProtocol)

A `FlexrayTpChannel` provides a Tx and an Rx pool of `NPdu`s which are used by the channel's `FlexRayTpConnection`s.

`FlexrayTpConnection`s are used for communication between one source and one or more target device(s). These communication partners are specified by the source and target associations to `FlexrayTpNode`s, providing the diagnostic TpAddresses and the connection to the topology description. The actual payload to be transported by the `FlexrayTpConnection` is identified by the references `directTpSdu` and `reversedTpSdu` to `IPdu`s. When one of the two SDUs is omitted, the connection shall be used unidirectional.

Class	**FlexrayTpChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	A channel is a group of connections sharing several properties. The FlexRay Transport 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,TpChannel			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
ackType	TpAckType	1	attr	Type of Acknowledgement.
extendedAddressing	Boolean	1	attr	Adressing Type of this connection: true: Two Bytes false: One Byte
flowControlPdu	NPdu	0..1	ref	Reference to the Flow Control NPdu. Please note that this reference is deprecated and will be removed in future. **Tags:** atp.Status=obsolete
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).
maxBs	Integer	0..1	attr	This attribute defines the number of consecutive CFs between two FCs (block size). Valid values are 1 .. 16 when retry is activated, and 0 .. 255 otherwise.
maxBufferRequest	Integer	0..1	attr	Please note that this attribute is deprecated and will be removed in future. maxFcWait will be used instead to configure the maximum number of wait frames on receiver side. On the sender side, timeCs defines the maximum time for retries. **Tags:** atp.Status=obsolete
maxFcWait	PositiveInteger	0..1	attr	This attribute defines the maximal number of wait frames to be sent for a pending connection. Range is 0..255.
maxFrIf	Integer	0..1	attr	Please note that this attribute is deprecated and will be removed in future. Old description: This parameter defines the maximum number of trying to send a frame when the FrIf returns an error. **Tags:** atp.Status=obsolete

Attribute	Datatype	Mul.	Kind	Note
maxRetries	Integer	0..1	attr	This parameter 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.
minimumMulticastSeparationTime	Float	0..1	attr	This attribute defines the minimum amount of time between two succeeding CFs of a 1:n segmented transmission in seconds. Valid values are 0, 100μs, 200μs ... 900μs, 1ms, 2ms .. 127ms. The value can be changed at runtime using the FrArTp_ChangeParameter interface. minimumMulticastSeparationTime must be an integer multiple of the cycle length multiplied with the multiplexing factor, i.e. minimumMulticastSeparationTime = n * cycle * m, where n is an integer >= 0, cycle is FlexrayCluster.cycle, and m is the cycle multiplexor of those cycles where PDUs of the PDU pool are scheduled. Please note: Due to the scheduling strategies of FrTp, minimumMulticastSeparationTime can only be kept to a degree defined by the maximum temporal distance of the PDUs of a PDU pool within one FlexRay cycle. Range: 0 .. 0.127
minimumSeparationTime	Float	1	attr	This attribute defines the minimum amount of time between two succeeding CFs of a 1:1 segmented transmission in seconds. Valid values are 0, 100μs, 200μs .. 900μs, 1ms, 2ms .. 127ms. The value can be changed at runtime using the FrArTp_ChangeParameter interface. The minimumSeparationTime must be an integer multiple of the cycle length multiplied with the multiplexing factor, i.e. minimumSeparationTime = n * cycle * m, where n is an integer >=0, cycle is FlexrayCluster.cycle, and m is the cycle multiplexor of those cycles where PDUs of the PDU pool are scheduled. Please note: Due to the scheduling strategies of FrTp, minimumSeparationTime can only be kept to a degree defined by the maximum temporal distance of the PDUs of a PDU pool within one FlexRay cycle.
multicastSegmentation	Boolean	1	attr	This attribute defines whether segmentation within a 1:n connection is allowed or not.

Attribute	Datatype	Mul.	Kind	Note
nPdu	NPdu	*	ref	A FlexRayTpChannel references a set of NPdus. These NPdus are logically assembled into a pool of Rx NPdus and another pool of Tx NPdus. It must be ensured that a second channel either references all NPdus of such a pool, or none.
pduPool	NPdu	*	aggr	Please note that this aggregation is deprecated and will be removed in future. The nPdu reference will be used instead. **Tags:** atp.Status=obsolete
timeBr	Float	0..1	attr	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.
timeBuffer	Float	0..1	attr	Please note that this attribute is deprecated and will be removed in future. timeBr will be used instead to configure the delay between two wait frames (and thus two buffer requests) on receiver side. On sender side, the main task cycle will be used. **Tags:** atp.Status=obsolete
timeCs	Float	1	attr	This parameter 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	Float	0..1	attr	Please note that this attribute is deprecated and will be removed in future. Old description: This parameter defines the time in seconds of waiting for the next try (if retry is activated) to send via FrIf_Transmit. Specified in seconds. **Tags:** atp.Status=obsolete
timeoutAr	Float	0..1	attr	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).
timeoutAs	Float	0..1	attr	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).

Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
timeoutBs	Float	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	Float	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.
tpConnection	FlexRayTpConnection	1..*	aggr	Group of connections that can be used in this channel.
transmitCancellation	Boolean	0..1	attr	With this switch Tx and Rx Cancellation can be turned on or off.

Table 5.82: FlexrayTpChannel

Enumeration	**TpAckType**
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols
Note	Type of Acknowledgement.
Literal	*Description*
ackWithRt	Acknowledgement with retry.
ackWithoutRt	Acknowledgement without retry.
noAck	No acknowledgement.

Table 5.83: TpAckType

Enumeration	**MaximumMessageLengthType**
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols
Note	This specifies the maximum message length for the particular channel.
Literal	*Description*
iso	Up to (2**12)-1 Byte message length (No FF-Ex or SF-E or AF shall be used and recognized).
iso6	As ISO, but the maximum payload length is limited to 6 byte (SF-I, FF-I, CF). This is necessary to route TP on CAN when using Extended Addressing or Mixed Addressing on CAN.
l4g	SF-E allowed (SF of arbitrary length depending on FrTpPduLength), up to (2**32)-1 byte message length (all FF-x allowed).

Table 5.84: MaximumMessageLengthType

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,TpNode			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*

Table 5.85: FlexrayTpNode

Class	**FlexRayTpConnection**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	A connection within a channel identifies the sender and the receiver of this particular communication. The FlexRayTp module routes a Pdu through this connection.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
connection PrioPdus	Integer	0..1	attr	This parameter defines the number of PDUs that shall be reserved for this connection when it is active. The range is 1-255.
directTpSd u	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 must not reference a NPdu with this tpSdu reference.
flowControl Pdu	NPdu	0..1	ref	Please note that this reference is deprecated and will be removed in future. The PDU pool referenced by the FlexRayTpChannel as nPdu will be used instead. **Tags:** atp.Status=obsolete
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 must not reference a NPdu with this tpSdu reference.
source	FlexrayTpNode	1	ref	The source of the TP connection.
target	FlexrayTpNode	1..*	ref	The target of the TP connection.
transmitPdu	NPdu	*	ref	Please note that this reference is deprecated and will be removed in future. The PDU pool referenced by the FlexRayTpChannel as nPdu will be used instead. **Tags:** atp.Status=obsolete

Table 5.86: FlexRayTpConnection

5.12.2 FlexRay ISO 10681-2 Transport Layer

This section describes the FlexRay ISO 10681-2 Transport Protocol.

The FlexRay ISO Transport Layer supports multiple sessions, i.e. multiple segmented transfers can be handled at the same time. Thus, multiple FlexRayIsoTpConnections can be defined on the same ECU. Each FlexRayIsoTpConnection is controlled by configuration parameters defined in FlexRayIsoTpConnectionControl. The same FlexRayIsoTpConnectionControl can be reused for an arbitrary number of FlexRayIsoTpConnections.

A FlexRayIsoTpConnection defines the way of communication between a sender and a receiver and uses a FlexRayIsoTpPduPool of NPdus to transmit data to the FlexRay Interface. Each FlexRayIsoTpConnection 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.

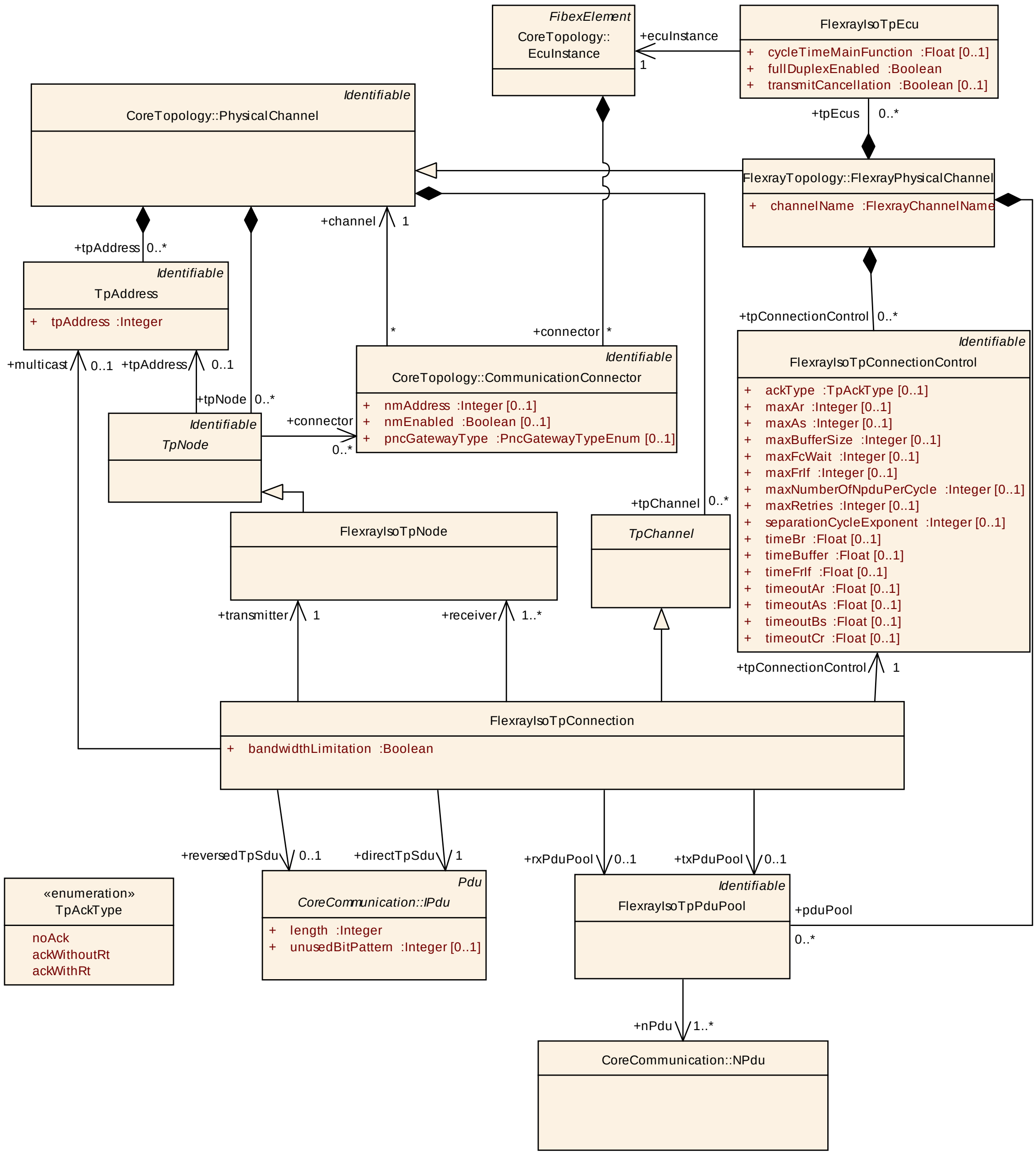

Figure 5.24: FlexRay ISO 10681-2 Transport Layer Configuration (TransportProtocols: FlexRayIsoTransportProtocol)

`FlexRayIsoTpConnections` 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 `FlexrayIsoTpNodes`, 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 `FlexRayIsoTpConnection` 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	**FlexrayIsoTpConnection**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	A connection identifies the sender and the receiver of this particular communication. The FlexRayIsoTp 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,TpChannel			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
bandwidthLimitation	Boolean	1	attr	Specifies whether the connection requires a bandwidth limitation or not.
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 FlexrayIsoTpConnection must not reference a NPdu with this tpSdu reference.
multicast	TpAddress	0..1	ref	TP address for 1:n connections.
receiver	FlexrayIsoTpNode	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 FlexrayIsoTpConnection must not reference a NPdu with this tpSdu reference.
rxPduPool	FlexrayIsoTpPduPool	0..1	ref	A connection has a reference to a set of NPdus (FrTpRxPduPool) which are defined for receiving data via this particular connection.
tpConnectionControl	FlexrayIsoTpConnectionControl	1	ref	Reference to the connection control.
transmitter	FlexrayIsoTpNode	1	ref	The source of the TP connection.
txPduPool	FlexrayIsoTpPduPool	0..1	ref	A connection has a reference to a set of NPdus (FrTpTxPduPool) which are defined for sending data via this particular connection.

Table 5.87: FlexrayIsoTpConnection

The FlexrayIsoTpConnection refers to the FlexrayIsoTpPduPool in two roles: **rxPduPool** and **txPduPool**.

[TPS_SYST_01064] Transmit/Receive Semantics of Pdu Pools ⌈ The transmit/receive semantics of Pdu Pools depends on the role of the regarded ECU:

- If the ECU is the transmitter then the txPduPool holds the sent NPdus and the rxPduPool holds the received NPdus.
- If the ECU is the receiver then the the txPduPool holds the received NPdus and the rxPduPool holds the sent 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 -> sent Pdus
RxPool = PDU_2 -> received Pdus

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

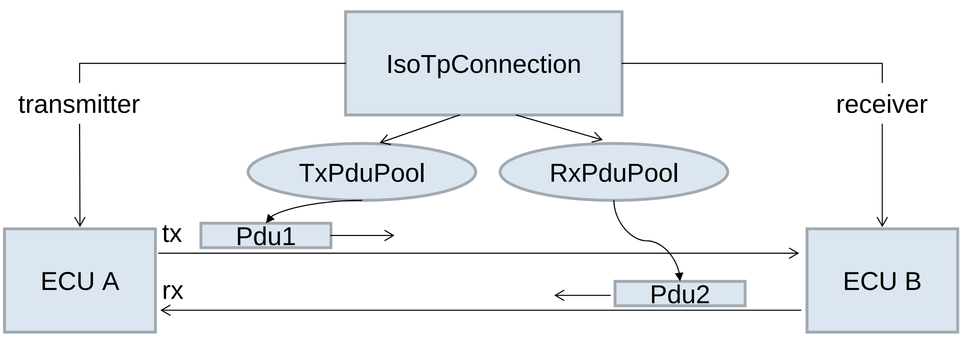

Figure 5.25: IsoTp Example

Class	FlexrayIsoTpConnectionControl			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	Configuration parameters to control a FlexRay TP connection.			
Base	ARObject,Identifiable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
ackType	TpAckType	0..1	attr	This parameter defines the type of acknowledgement which is used for the specific channel. For FlexrayIsoTp only noAck and ackWithRt shall be used.
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)
maxBufferSize	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
maxNumberOfNpduPerCycle	Integer	0..1	attr	This parameter limits the number of N-Pdus the sender is allowed to transmit within a FlexRay cycle.
maxRetries	Integer	0..1	attr	This parameter defines the maximum number of retries (if retry is configured for the particular channel).
separationCycleExponent	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	Float	0..1	attr	Time (in seconds) until transmission of the next FlowControl N-PDU.
timeBuffer	Float	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	Float	0..1	attr	This parameter defines the time of waiting for the next try to send. Specified in seconds.
timeoutAr	Float	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.

Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
timeoutAs	Float	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	Float	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	Float	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 5.88: FlexrayIsoTpConnectionControl

Class	**FlexrayIsoTpNode**			
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,TpNode			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*

Table 5.89: FlexrayIsoTpNode

Class	**FlexrayIsoTpPduPool**			
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			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
nPdu	NPdu	1..*	ref	Reference to NPdus that are part of the PduPool.

Table 5.90: FlexrayIsoTpPduPool

Class	**FlexrayIsoTpEcu**			
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	Float	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.
fullDuplex Enabled	Boolean	1	attr	The full duplex mechanisms is enabled if this attribute is set to true. Otherwise half duplex is enabled.
transmitCa ncellation	Boolean	0..1	attr	With this switch Tx and Rx Cancellation can be turned on or off.

Table 5.91: FlexrayIsoTpEcu

5.12.3 CAN Transport Layer

Similarly to the FlexRay TP, the CAN Transport Layer supports multiple sessions by means of so called CanTpConnectionChannels: Each CAN TP CanTpConnectionChannel uses its own resources, such as internal buffer, timer, state machine and thus can operate independently and simultaneously to other CanTpConnectionChannels.

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

A CanTpConnectionChannel is specifically used for communication between one source and one target device. These communication partners are specified using the source and target associations to CanTpNode, providing the diagnostic tpAddress and the connection to the topology description.

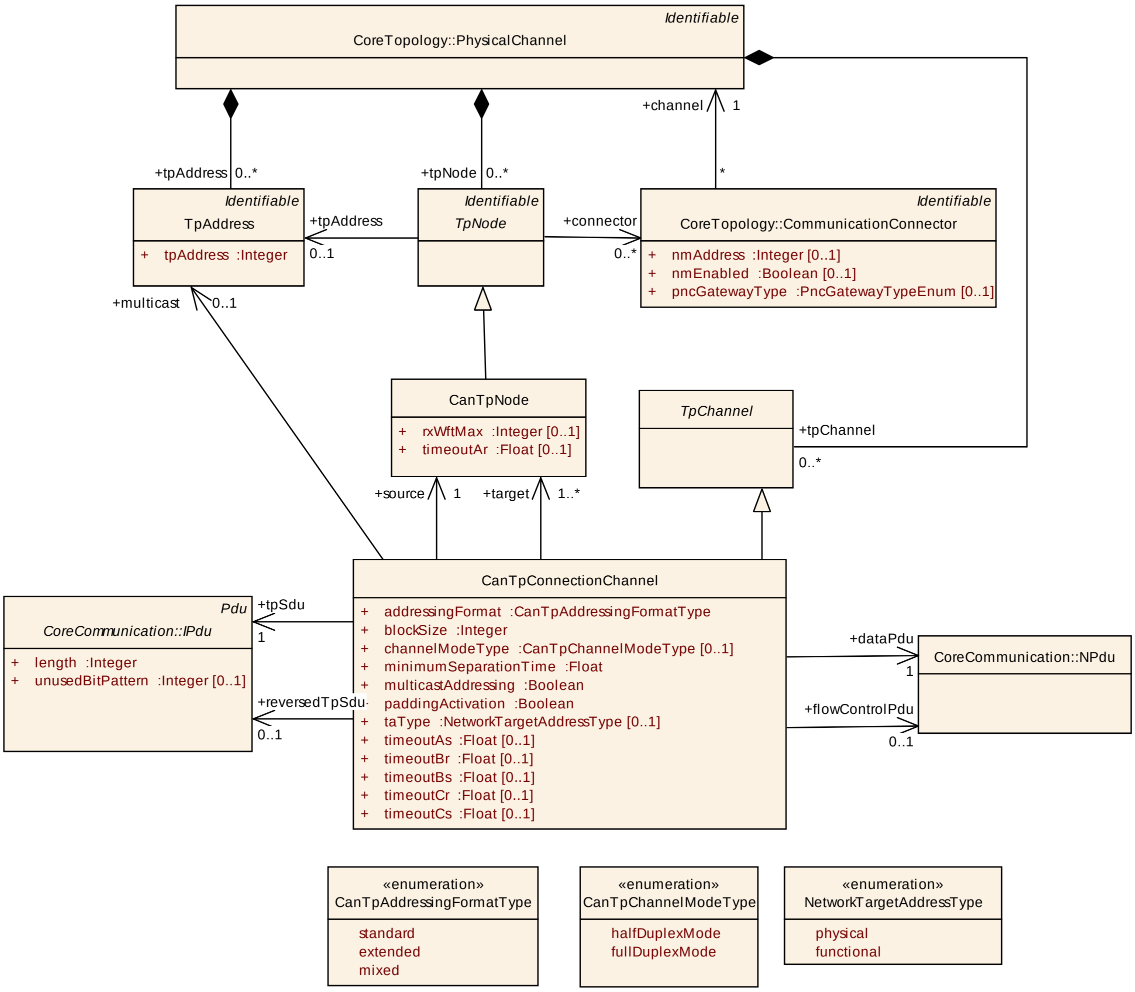

Figure 5.26: CAN Transport Layer Configuration (TransportProtocols: CanTransportProtocol)

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

Class	**CanTpConnectionChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	A connection channel 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 channel			
Base	ARObject,TpChannel			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
addressingFormat	CanTpAddressingFormatType	1	attr	Declares which communication addressing mode is supported.
blockSize	Integer	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
channelModeType	CanTpChannelModeType	0..1	attr	The CAN Transport Layer supports half and full duplex channel modes.
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
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.
minimumSeparationTime	Float	1	attr	This attribute defines the minimum amount of time (separation Time) between two succeeding CFs. Specified in seconds.
multicast	TpAddress	0..1	ref	TP address for 1:n connections.
multicastAddressing	Boolean	1	attr	Specifies the communication type: true: 1:n communication (Functional) false: 1:1 communication (Physical)
paddingActivation	Boolean	1	attr	Defines if the receive frame uses padding or not. 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)
reversedTpSdu	IPdu	0..1	ref	Reference to the IPdu that is segmented by the Transport Protocol. The reversedTpSdu is used if channelModeType in CanTpConnectionChannel is configured to fullDuplexMode. In this case this association references the IPdu used for the additional second direction.
source	CanTpNode	1	ref	The source of the TP connection.
taType	NetworkTargetAddressType	0..1	attr	Network Target Address type.
target	CanTpNode	1..*	ref	The target of the TP connection.
timeoutAs	Float	0..1	attr	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.

Attribute	Datatype	Mul.	Kind	Note
timeoutBr	Float	0..1	attr	N_Br is the elapsed time (in seconds) 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	Float	0..1	attr	Value in seconds of the N_Bs timeout. N_Bs is the time of transmission until reception of the next Flow Control N_PDU.
timeoutCr	Float	0..1	attr	Value in seconds of the N_Cr timeout. N_Cr is the time until reception of the next Consecutive Frame N_PDU.
timeoutCs	Float	0..1	attr	N_Cs is 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 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 CanTpConnection must not reference a NPdu with this tpSdu reference.

Table 5.92: CanTpConnectionChannel

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 5.93: CanTpAddressingFormatType

Enumeration	**CanTpChannelModeType**
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols
Note	
Literal	***Description***
fullDuplex Mode	full duplex channel mode
halfDuplex Mode	half duplex channel mode

Table 5.94: CanTpChannelModeType

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,TpNode			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
rxWftMax	Integer	0..1	attr	This parameter indicates how many Flow Control wait N-PDUs can be consecutively transmitted by the receiver.
timeoutAr	Float	0..1	attr	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.

Table 5.95: CanTpNode

Enumeration	**NetworkTargetAddressType**
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols
Note	Network Target Address type (see ISO 15765-2).
Literal	***Description***
functional	Functional request type
physical	Physical request type

Table 5.96: NetworkTargetAddressType

5.12.4 LIN Transport Layer

LinTpConnectionChannel is used for modelling 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 one LinTpChannel can be defined per Physical Channel.

LinTpChannel 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 LinTpChannel is specifically used for communication between one source and one target device. These communication partners are specified using the source and target associations to LinTpNode, providing the diagnostic tpAddress and the connection to the topology description.

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

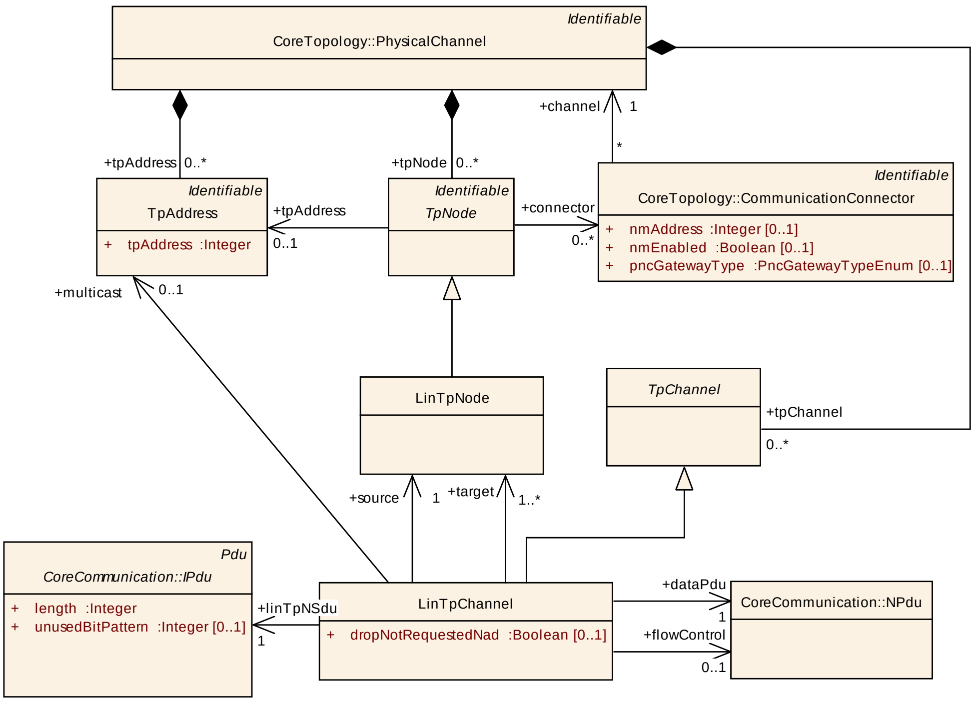

Figure 5.27: LIN Transport Layer Configuration (TransportProtocols: LinTransportProtocol)

Class	**LinTpChannel**			
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. The LinTp module routes a Pdu through the connection channel			
Base	ARObject,TpChannel			
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).
dropNotRequestedNad	Boolean	0..1	attr	Configures if TP Frames of not requested LIN-Slaves are dropped or not.

Attribute	Datatype	Mul.	Kind	Note
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 LinTpChannel must not reference a NPdu with this linTpNSdu reference.
multicast	TpAddress	0..1	ref	TP address for 1:n connections.
source	LinTpNode	1	ref	The source of the TP connection.
target	LinTpNode	1..*	ref	The target of the TP connection.

Table 5.97: LinTpChannel

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,TpNode			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**

Table 5.98: LinTpNode

5.12.5 Unicast TP Example

The example in Figure 5.28 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).

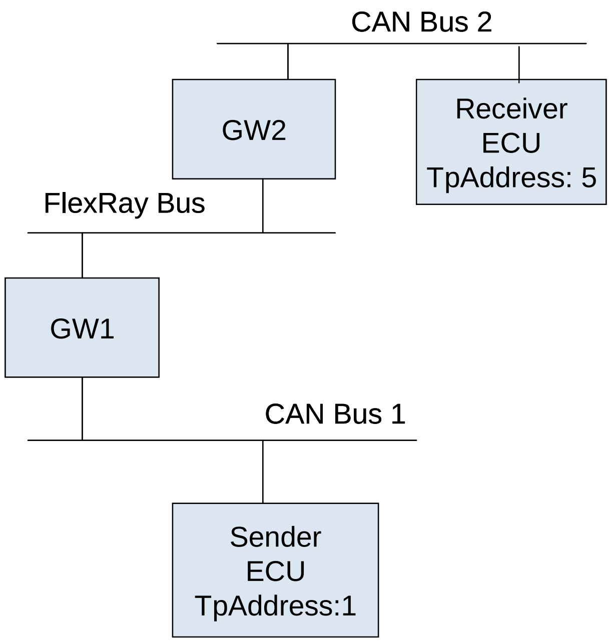

Figure 5.28: TP unicast Example

Modeling in the System Description:

```
CAN Physical Channel 1:
CanTpConnectionChannel
  source TpNode: Sender ECU, TpAddress: 1
  target TpNode: GW1, TpAddress: 5

FlexRay Physical Channel:
FlexRayTpConnection
  source TpNode: GW1, TpAddress: 1
  target TpNode: GW2, TpAddress: 5

CAN Physical Channel 2:
CanTpConnectionChannel
  source TpNode: GW2, TpAddress: 1
  target TpNode: Receiver ECU, TpAddress: 5
```

Please note that 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.

5.12.6 Multicast TP Example

A second example illustrates the usage of the multicast reference.

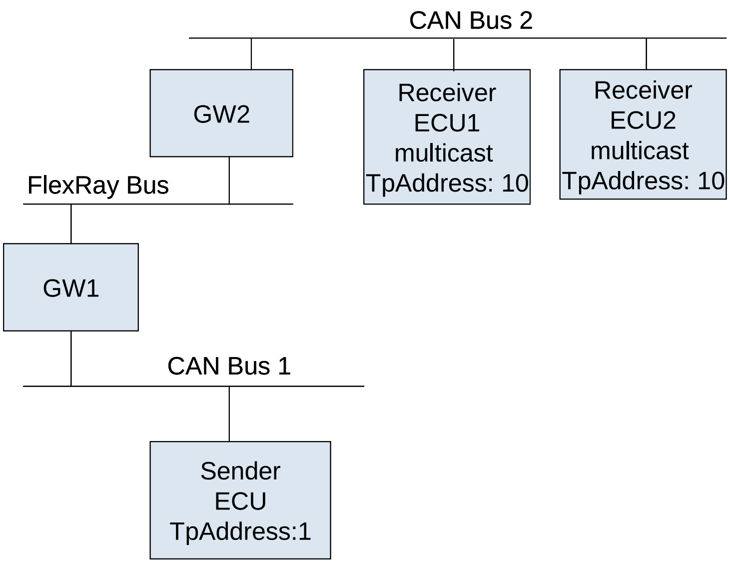

Figure 5.29: TP multicast Example

```
CAN Physical Channel 1:
CanTpConnectionChannel
  source TpNode: Sender ECU, TpAddress: 1
  target TpNode: GW1
  multicast TpAddress: 10

FlexRay Physical Channel:
FlexRayTpConnection
  source TpNode: GW1, TpAddress: 1
  target TpNode: GW2
  multicast TpAddress: 10

CAN Physical Channel 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.

5.13 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. And the FlexRay interface supports a fan-out where the same Pdu is mapped into more than one frame.

5.13.1 RTE fan-out

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

5.13.2 Pdu Router fan-out

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

5.13.3 Bus Interface fan-out

- The fan-out done in the FlexRay interface is described by the PduToFrameMapping element (The same PDU being mapped into more than one frame).
- 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.

5.13.4 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 fanout 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).

5.13.5 Semantic Rules

- IPduTriggering
  - 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.

5.14 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 5.3 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.

The `UserDefinedIPdu` can also be used to describe the communication of Pdus which shall be routed by the PduR in a gateway ECU.

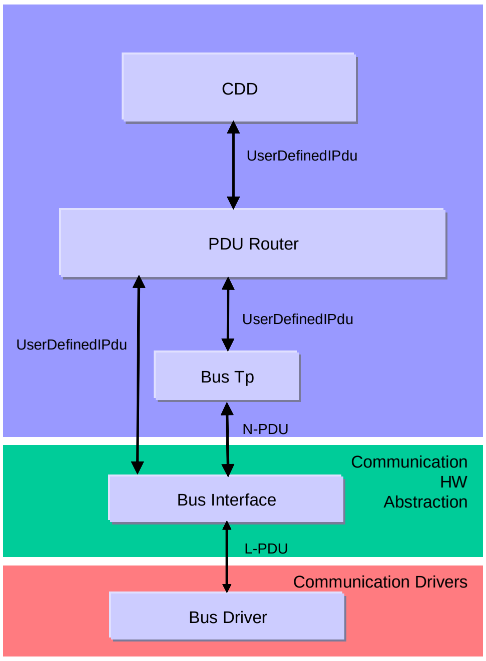

Figure 5.30: CDD over PduR

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 (without the need to be routed in a gateway ECU). 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 [19] for more details.

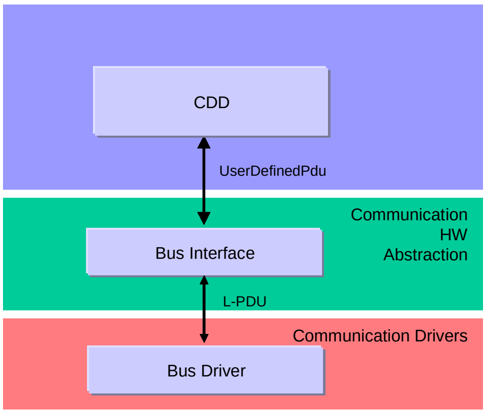

Figure 5.31: CDD over Bus Interface

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 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

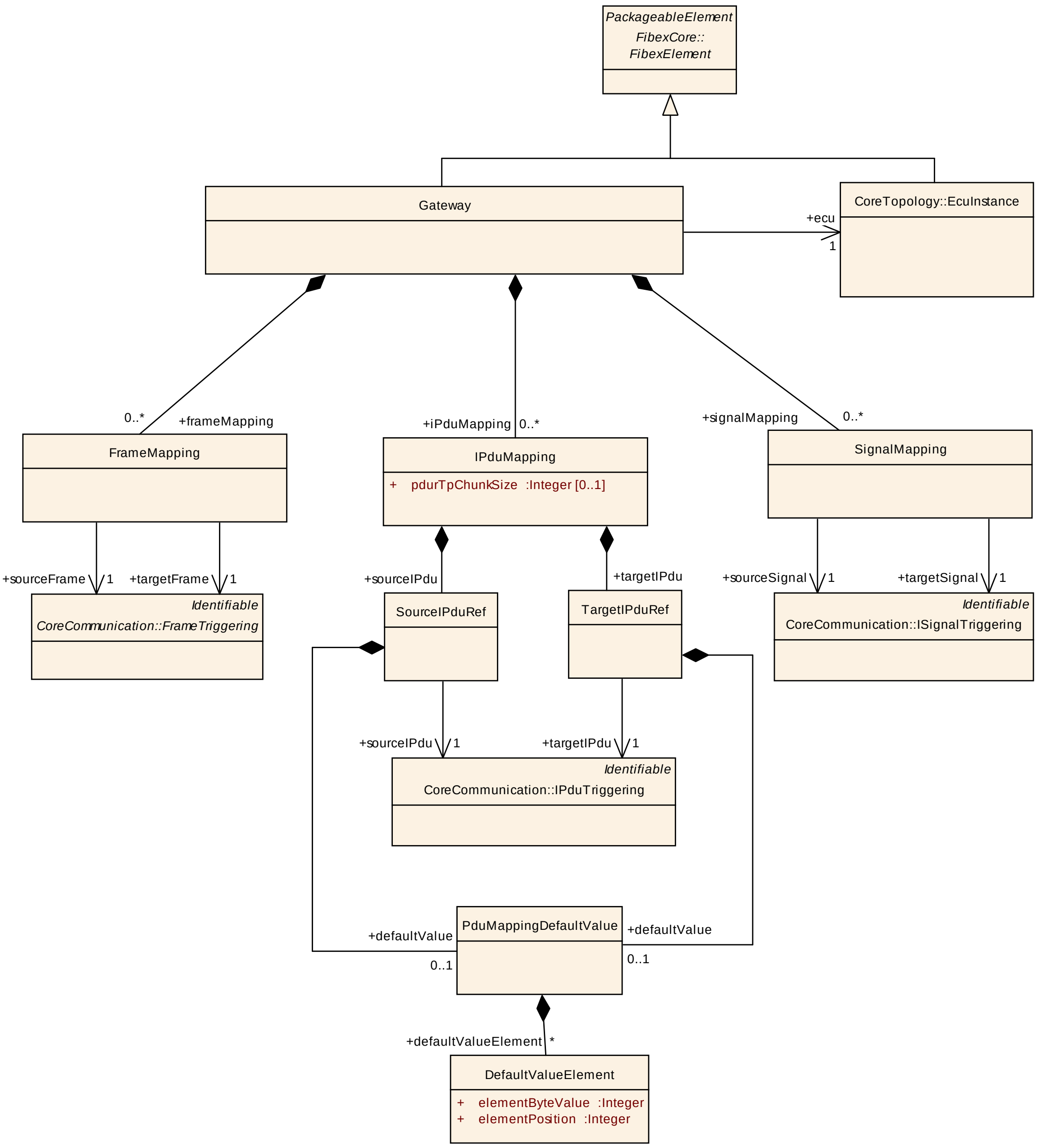

Figure 6.1: Communication Overview (Fibex4Multiplatform: Gateway)

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.

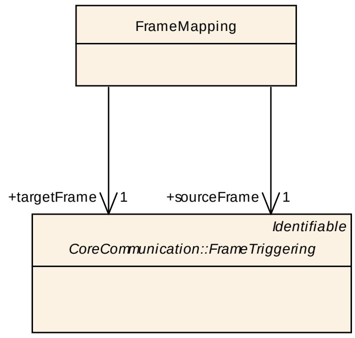

Figure 6.2: Frame Mapping (Fibex4Multiplatform: FrameMapping)

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***
sourceFrame	FrameTriggering	1	ref	Source destination of the referencing mapping.
targetFrame	FrameTriggering	1	ref	Target destination of the referencing mapping.

Table 6.1: 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 `IPduTriggering`.

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 IpduTriggerings must be described as gateway mappings.

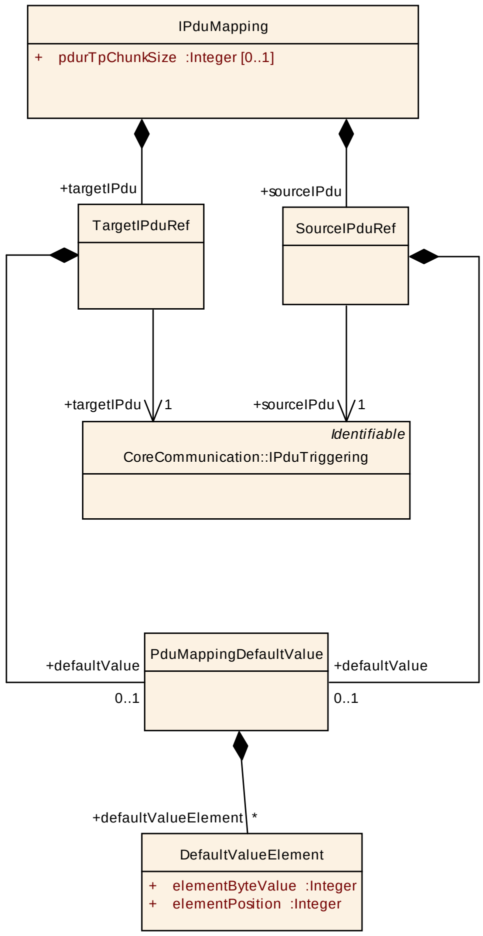

Figure 6.3: I-Pdu Mapping (Fibex4Multiplatform: IPduMapping)

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
pdurTpChunkSize	Integer	0..1	attr	Optionally defines the to be configured Pdu Router TpChunkSize for this routing relation.
sourceIPdu	SourceIPduRef	1	aggr	Source destination of the referencing mapping.
targetIPdu	TargetIPduRef	1	aggr	Target destination of the referencing mapping.

Table 6.2: 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	IPduTriggering	1	ref	IPdu Reference

Table 6.3: TargetIPduRef

Class	SourceIPduRef			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Note	Source 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.
sourceIPdu	IPduTriggering	1	ref	IPdu Reference

Table 6.4: SourceIPduRef

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	*	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 SignalMapping 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 an ISignal which is part of a SignalIPdu. The ISignal refers to the to be routed SystemSignal or SystemSignalGroup.

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

The 1:n multicast routing is supported with the definition of several SignalMappings. See also the COM Signal Gateway fan-out description in section 5.13.4.

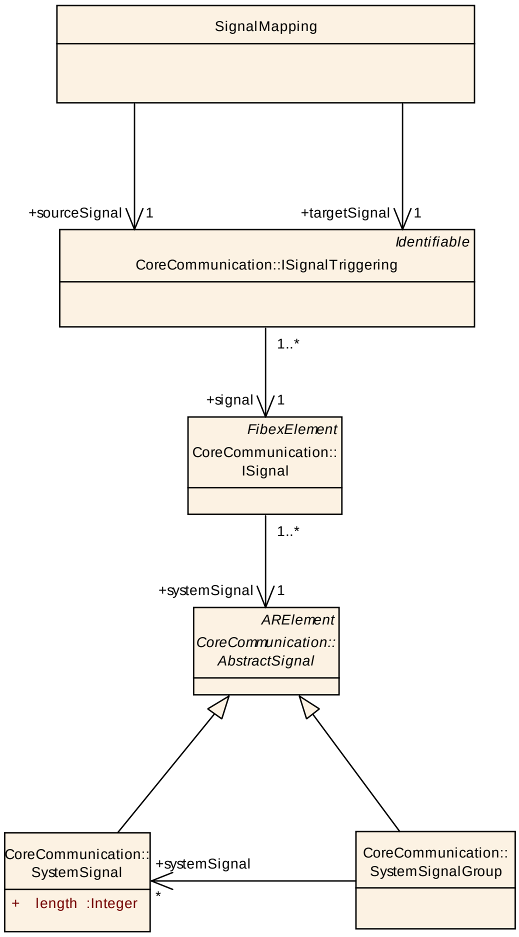

Figure 6.4: Signal Mapping (Fibex4Multiplatform: Signal Mapping)

Class	**SignalMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Note	Arranges those signals 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*
sourceSignal	ISignalTriggering	1	ref	Source destination of the referencing mapping.
targetSignal	ISignalTriggering	1	ref	Target destination of the referencing mapping.

Table 6.7: SignalMapping

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 ECU Extract of the System Configuration Description

As shown in Figure 1.2 in chapter 1.3 only parts of the System Configuration Description are used as input for the ECU generation step for an individual ECU. Only the relevant information that is needed to generate this individual ECU is included, all other information is removed from the configuration description. This extract is called *ECU extract of the System Configuration Description*.

In general, from a given System Configuration Description, it is straightforward to generate an extract for a specific ECU: Take the XML input and remove all elements that are not relevant for that ECU, such as SW components mapped to other ECUs, topology information elements that are not directly connected to the ECU, etc. There is one exemption to this simple "remove" rule: the communication mapping may need to be extended, which will be described in more detail in chapter 8.2.

8.1 Inclusion of elements

The following table shows the rules that define whether an element has to be included in the ECU extract or not. In the table, "included" means that the element as originally taken from the System Description is possibly being modified and/or reduced to contain the information relevant for the targeted ECU.

System top level	
System	Always included
Software Composition	Always included
System Mapping	Always included
FibexElement	Always included
Core Topology	
Communication Cluster	Included if ECU is connected to that cluster
Physical Channel	Included if ECU is connected to that physical channel
ECUInstance	Included if ECU under consideration
CommunicationConnector	Included if part of ECU under consideration
CommunicationController	Included if part of ECU under consideration
CommConnectorPort	Included if part of ECU under consideration
SignalPort	Included if part of ECU under consideration
IPduPort	Included if part of ECU under consideration

FramePort	Included if part of ECU under consideration
Core Communication	
FrameTriggering	Included if ECU sends or receives this frame (if included connector references to this FrameTriggering)
IPduTriggering	Included if ECU sends or receives this IPdu (if included connector references to this IPduTriggering)
IPduTiming	Included if ECU sends or receives this SignalIPdu
ISignalTriggering	Included if ECU sends or receives this frame (if included connector references to this ISignalTriggering)
Frame	Included if ECU sends or receives this frame (if included FrameTriggering references to this Frame)
ISignal	Included if ECU sends or receives this ISignal (if included ISignalTriggering references to this Frame)
IPdu	Included if ECU sends or receives this IPdu
NPdu	included if PDUToFrameMapping that refers to this NPdu is included
NmPdu	included if PDUToFrameMapping that refers to this NmPdu is included
ISignalToIPduMapping	Included if aggregating IPdu is included
PduToFrameMapping	Included if aggregating Frame is included
IPduGroup	Included if referenced IPdu is included
SystemSignal	Included if there is an ISignal included that references to this SystemSignal

SystemSignalGroup	Included if ECU sends this SystemSignalGroup. On the receiving Ecu the following cases exist: • only one SystemSignal out of the transmitted SystemSignalGroup is received: no SystemSignalGroup is required in the Ecu Extract of the receiving Ecu. • more than one but not all SystemSignals out of the transmitted SystemSignalGroup are received: new SystemSignalGroup shall be created in the Ecu Extract of the receiving Ecu containing the received SystemSignals. • all SystemSignals out of the transmitted SystemSignalGroup are received: the original SystemSignalGroup shall be the taken over to the Ecu Extract of the receiving Ecu.
SignalPdu	Included if ECU sends or receives this IPdu
MultiplexedPdu	Included if ECU sends or receives this IPdu
StaticPart	Included if aggregating MultiplexedPdu is included
DynamicPart	Included if aggregating MultiplexedPdu is included
IPduTriggering	
TransmissionModeDeclaration	Included if aggregating IPduTiming is included
TransmissionModeCondition	Included if aggregating IPduTiming is included
DataFilter	Included if aggregating TransmissionModeCondition is included
TransmissionModeFalseTiming	Included if aggregating IPduTiming is included
CyclicTiming	Included if aggregating TransmissionModeFalseTiming is included or if aggregating IPduTiming is included.

EventControlledTiming	Included if aggregating Transmission-ModeFalseTiming is included or if aggregating IPduTiming is included.
RelativelyScheduledTiming	Included if aggregating IPduTiming is included
FrameTriggering	
AbsolutelyScheduledTiming	Included if aggregating FrameTriggering is included
RelativelyScheduledTiming	Included if aggregating FrameTriggering is included
ISignalTriggering	
CyclicTiming	Included if aggregating ISignalTriggering is included
EventControlledTiming	Included if aggregating ISignalTriggering is included
Fibex4FlexRay	
CycleCounter	Included if aggregating AbsolutelyScheduledTiming is included
CycleRepetition	Included if aggregating AbsolutelyScheduledTiming is included
Fibex4Lin	
LinScheduleTable	Included if RelativelyScheduledTiming that refers to this LinSchedulingTable is included
AssingNadTiming	Included if aggregating LinFrameTriggering is included
AssignFrameIdTiming	Included if aggregating LinFrameTriggering is included
UnassignFrameIdTiming	Included if aggregating LinFrameTriggering is included
FreeFormat	Included if aggregating RelativelyScheduledTiming is included
Fibex4Multiplatform	
Gateway	Included if ECU under consideration is referenced
FrameMapping	Included if aggregating Gateway is included
IPduMapping	Included if aggregating Gateway is included
SignalMapping	Included if aggregating Gateway is included
SourceIPduRef	Included if aggregating IPduMapping is included
TargetIPduRef	Included if aggregating IPduMapping is included

PduMappingDefaultValue	Included if aggregating TargetPduRef or SourcePduRef is included
DataMapping	
DataMapping	Always included
SenderReceiverToSignalMapping	Added or included if signal is sent or received by the considered ECU. Added means that the mapping may need to be added if only a mapping of the sender existed and ECU is receiver. Then the corresponding receiving SW component's port needs to be mapped.
SenderReceiverToSignalGroupMapping	Added or included if a signal, which is part of a signal group, is sent or received by the considered ECU. Added means that the mapping may need to be added if only a mapping of the sender existed and ECU is receiver. Then the corresponding receiving SW component's port needs to be mapped.
SenderRecRecordTypeMapping	Included if aggregating SenderReceiverToSignalGroupMapping is included.
SenderRecArrayTypeMapping	Included if aggregating SenderReceiverToSignalGroupMapping is included.
SenderRecRecordElementMapping	Included if aggregating SenderReceiverToSignalGroupMapping is included.
SenderRecArrayElementMapping	Included if aggregating SenderReceiverToSignalGroupMapping is included.
ClientServerToSignalGroupMapping	Added or included if a signal, in which an argument of an operation is transported, is sent or received by the considered ECU. Added means that the mapping may need to be added if only a mapping of the sender existed and ECU is receiver. Then the corresponding receiving SW component's port needs to be mapped.
ClientServerPrimitiveTypeMapping	Included if aggregating ClientServerToSignalGroupMapping is included.
ClientServerArrayTypeMapping	Included if aggregating ClientServerToSignalGroupMapping is included.
ClientServerRecordTypeMapping	Included if aggregating ClientServerToSignalGroupMapping is included.
ClientServerArrayElementMapping	Included if aggregating ClientServerArrayTypeMapping is included.

ClientServerRecordElementMapping	Included if aggregating ClientServerArrayTypeMapping is included.
ClientIdMapping	Included if aggregating ClientServerToSignalGroupMapping is included.
SequenceCounterMapping	Included if aggregating ClientServerToSignalGroupMapping is included.
ApplicationErrorMapping	Included if aggregating ClientServerToSignalGroupMapping is included.
EmptySignalMapping	Included if aggregating ClientServerToSignalGroupMapping is included.
SW Mapping	
SwcToECUMapping	Included if considered ECU is referenced.
SwcToImplMapping	Included if SWC, which is mapped to the ECU, is referenced.
MappingConstraint	Not included (also all aggregated elements are not included)
ECUResourceEstimation	Included if considered ECU is referenced.
ResourceConsumption	Included if aggregating ECUResourceEstimation is included
StackUsage	Included if aggregating ResourceConsumption is included
HeapUsage	Included if aggregating ResourceConsumption is included
ExecutionTime	Included if aggregating ResourceConsumption is included
SignalPathConstraints	
SignalPathConstraint	Not included (also all aggregated elements are not included)
ECU Resource Mapping	
ECUMapping	Included if considered ECU is referenced.
CommunicationControllerMapping	Included if aggregating ECUMapping is included
HwPortMapping	Included if aggregating ECUMapping is included
From Software Component Template	
CompositionType	Included if it is the flattened top level composition of the system. Aggregated elements are included if they are mapped to this ECU, see below.

ComponentPrototype and the matching type	Included if mapped to this ECU, i.e. referenced by a SwCompToEcuMapping that references to ECU under consideration
Implementation and all aggregated elements	Included if mapped to this ECU, i.e. referenced by a SwCompToImplMapping that references to a SWC, which is mapped to ECU under consideration
Internal Behavior	Included if at least one ComponentPrototype of the referenced AtomicSoftwareComponentType is mapped to this ECU
From ECU Resource Template	
ECU and everything aggregated	included if referencing ECU instance is included (i.e. ECU is of this type)

Table 8.1: Inclusion of elements in the ECU Extract

8.2 SW component inclusion and data mapping

As mentioned before, there is a slight complication to above include/exclude rules. This can be shown best with an example. Assume a simple topology with two ECUs A and B and two frames X (sent from A to B) and Y (sent from B to A) as shown in Figure 8.1.

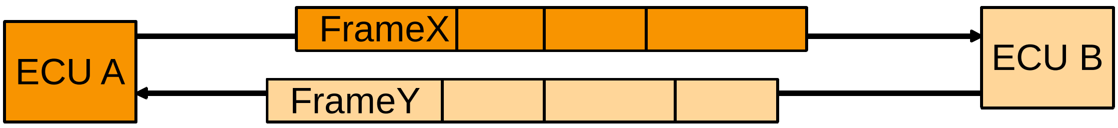

Figure 8.1: Example topology with two ECUs and two frames exchanged between them

Furthermore assume a SW composition as shown in Figure 8.2. It consists of five atomic SW components ’A1’ to ’A3’ (aggregated in composition ’SwCompA’) and ’B1’ / ’B2’ (aggregated in composition ’SWCompB’). The overall composition ’SWCompAplusB’ aggregates ’SwCompA’ and ’SWCompB’.

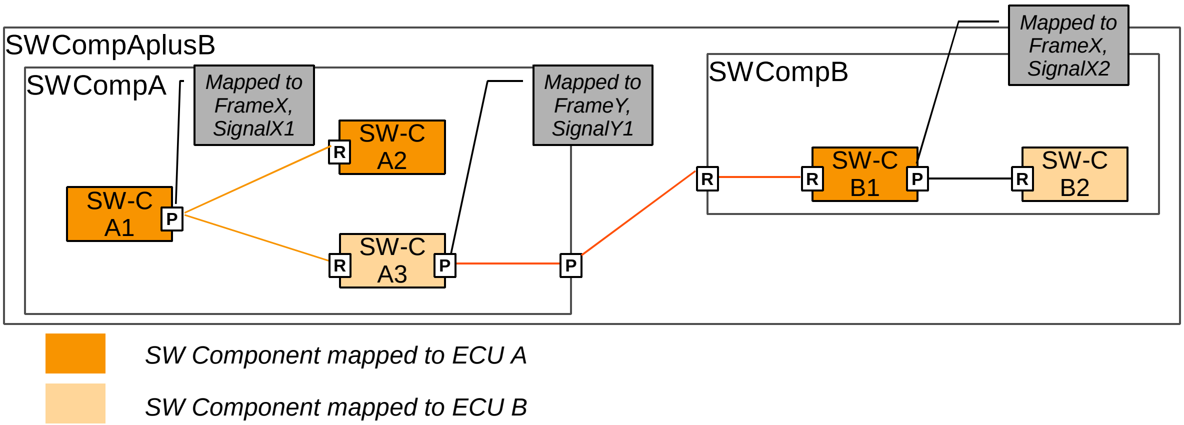

Figure 8.2: Example SW composition with mapping information

The atomic SW components 'A1', 'A2' and 'B1' are mapped to 'ECU A', the others to 'ECU B'. The data sent from

- 'A1' to 'A3' is mapped to 'FrameX', 'SignalX1',
- 'B1' to 'B2' is mapped to 'FrameX', 'SignalX2' and
- 'A3' to 'B1' is mapped to 'FrameY', 'SignalY1'.

As usual, the data mapping rules refer to the data element in the P-Port of the sending SW component.

Figure 8.3 shows how the ECU extract for ECU A of this SW composition would look like: Only those atomic SW components are included that are mapped to ECU A.

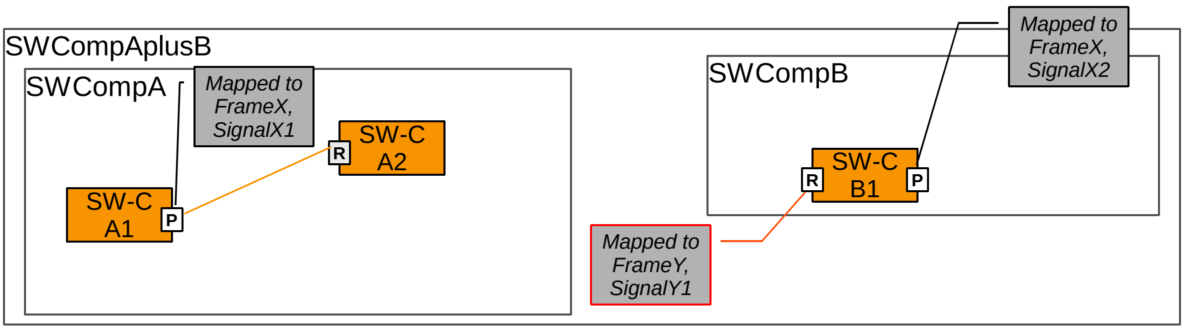

Figure 8.3: Example ECU extract for ECU A of above introduced composition

All compositions are included since they have aggregated atomic SW components which are included and cannot be left out for that reason.

Only those connectors are included that represent intra-ECU communication (in our example, only 'A1' to 'A2'), since this information is still needed for the RTE generation.

Connectors that were used to connect to SW components that are not included in the extract are not included either. Instead, the mapping to a signal in a frame 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 'FrameY', '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 ECU generator has the information that B1 receives the data via 'FrameY'.

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 ECU 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 8.4.

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 8.4 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.

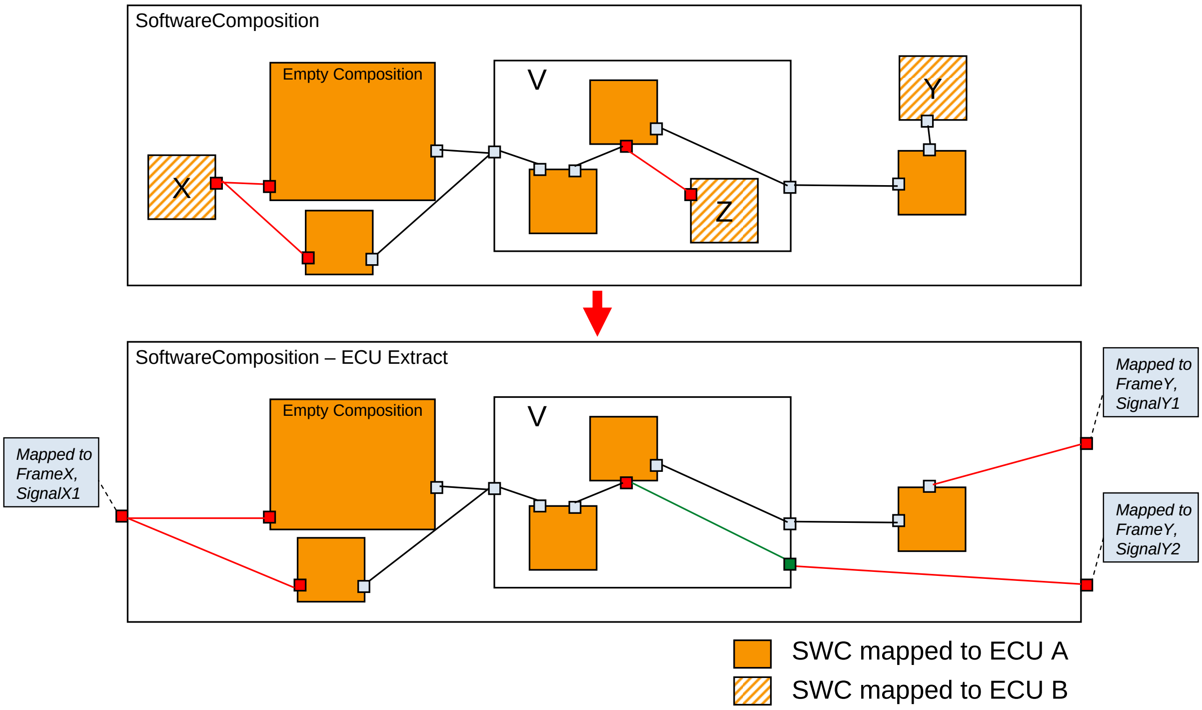

Figure 8.4: Example with software components mapped to two ECUs

9 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 must be described in order to answer typical questions like: How shall a supplier use the information in a System Template 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

9.1 Com Mapping

BSW Module	BSW Context		
Com	Com		
BSW Parameter		**BSW Type**	
ComConfig		ParamConfContainerDef	
BSW Description			
This container contains the configuration parameters and sub containers of the 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		IntegerParamDef	
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		ParamConfContainerDef	
BSW Description			
Each instance of this container defines one mapping of the integrated Signal Gateway.			
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. Each pair consists in a source and a target referencing to a ISignalTriggering.		
M2 Parameter			
Fibex::Fibex4Multiplatform::SignalMapping			
Mapping Rule			**Mapping Type**
Create Container for each SignalGateway Mapping that is defined on the ECU.			full

BSW Module	BSW Context	
Com	Com/ComConfig/ComGwMapping	
BSW Parameter		**BSW Type**
ComGwDestination		ChoiceContainerDef
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::SignalMapping.targetSignal		
Mapping Rule		**Mapping Type**
Create Container for each targetSignal reference that is defined in the Signal Mapping.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComGwMapping/ComGwDestination	
BSW Parameter		**BSW Type**
ComGwDestinationDescription		ParamConfContainerDef
BSW Description		
Description of a gateway destination. This container allows defining a gateway destination without the configuration of a complete COM signal. This allows adding / changing gateway relations post build without the configuration of new signals.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the ISignal for which the ISignalTriggering is defined.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalTriggering.signal		
Mapping Rule		**Mapping Type**
Informations can be derived from ISignalToIPduMapping element via the ISignal that is referenced by the destination ISignalTriggering within the SignalMapping,		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestination Description	
BSW Parameter		**BSW Type**
ComBitPosition		IntegerParamDef
BSW Description		
Starting position within the I-PDU. This parameter refers to the position in the I-PDU and not in the shadow buffer.		
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.startPosition		
Mapping Rule		**Mapping Type**
The ISignalToIPduMapping element contains the startPosition attribute. To describe the mapping of a Signal to the IPdu the ISignal element shall refer to a SystemSignal.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestination Description	
BSW Parameter		**BSW Type**
ComGwIPduRef		SymbolicNameReferenceParamDef
BSW Description		
Symbolic reference to an I-PDU of a Signal Gateway source or destination description.		

M2 Template	M2 Description	
SystemTemplate	Reference to the ISignal for which the ISignalTriggering is defined.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalTriggering.signal		
Mapping Rule		**Mapping Type**
Create reference for each existing ISignalToIPduMapping that is referenced from the regarded Signal Gateway.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestination Description	
BSW Parameter		**BSW Type**
ComSignalDataInvalidValue		IntegerParamDef
BSW Description		
COM391: On receiver side: When this value is received it is recognized as the invalid value and the appropriate invalid action (as specified by ComDataInvalidAction) is performed. COM501: On sender side: This configures the data invalid value that is used by a call to Com_InvalidateSignal.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Com	Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestination Description	
BSW Parameter		**BSW Type**
ComSignalEndianess		EnumerationParamDef
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 SignalIPdu. The byte ordering "Little Endian" (MostSignificantByteLast) and "Big Endian" (MostSignificantByteFirst) 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/ComGwMapping/ComGwDestination/ComGwDestination Description	
BSW Parameter		**BSW Type**
ComSignalInitValue		IntegerParamDef
BSW Description		

COM170: Initial value for this signal. The default value is 0. The lower n-bits of the configured Integer shall be used as init-value for an n-bit sized signal type. COM483: If the signal is of type UINT[n], the Integer's least significant byte shall be assigned to the byte array's last byte. The second-least significant byte shall be assigned to the byte array's last but one byte, and so on.		
M2 Template	**M2 Description**	
SystemTemplate	Optional reference to a SystemSignal's initValue in case the System Description doesn't use a complete Software Component Description. If a full DataMapping exist for the SystemSignal this information may be available from a configured ComSpec.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SystemSignal.initValue		
Mapping Rule		**Mapping Type**
This information may be available in the system description (referenced by the SystemSignal) or in the SWC description (Receiver or Sender COM Spec).		full

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestination Description		
BSW Parameter		**BSW Type**	
ComTransferProperty		EnumerationParamDef	
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. TRIGGERED: Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU. PENDING: A write access to this signal never triggers the transmission of the corresponding I-PDU. TRIGGERED_ON_CHANGE: Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU, but only in case the written value is different to the locally stored (last written or init) value. TRIGGERED_WITHOUT_REPETITION: Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU just once without a repetition. TRIGGERED_ON_CHANGE_WITHOUT_REPETITION: Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU just once without a repetition, but only in case the written value is different to the locally stored (last written or init) value.			
M2 Template	**M2 Description**		
SystemTemplate	The triggered or triggeredOnChange 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 IPdu.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.transferProperty			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestination Description
BSW Parameter	**BSW Type**
ComUpdateBitPosition	IntegerParamDef
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.	
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/ComGwMapping/ComGwDestination
BSW Parameter	**BSW Type**
ComGwSignal	ParamConfContainerDef
BSW Description	
This container allows specifying a gateway source or destination respectively with a reference to a ComSignal, a ComGroupSignal or a ComSignalGroup.	
M2 Template	**M2 Description**
SystemTemplate	Reference to the ISignal for which the ISignalTriggering is defined.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::ISignalTriggering.signal	
Mapping Rule	**Mapping Type**
Create Container if ISignal is referenced from Gateway::SignalMapping	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComGwMapping/ComGwDestination/ComGwSignal
BSW Parameter	**BSW Type**
ComGwSignalRef	ChoiceReferenceParamDef
BSW Description	
Reference to an object of a gateway relation. Either to a ComSignal, ComGroupSignal or to a SignalGroup.	
M2 Template	**M2 Description**
SystemTemplate	An ISignalToIPduMapping describes the mapping of ISignals to ISignalIPdus and defines the position of the ISignal within an ISignalIPdu.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping	
Mapping Rule	**Mapping Type**
System Template element ISingalToIPduMapping represents the ComSignal (SystemSignal that is referenced by an ISignal), ComSignalGroup (SystemSignal Group that is referenced by an ISignal) or ComGroupSignal (SystemSignal that is part of a SystemSignalGroup)	full

BSW Module	BSW Context	
Com	Com/ComConfig/ComGwMapping	
BSW Parameter		**BSW Type**
ComGwSource		ChoiceContainerDef
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::SignalMapping.sourceSignal		
Mapping Rule		**Mapping Type**
Create Container for each sourceSignal reference that is defined in the Signal Mapping.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComGwMapping/ComGwSource	
BSW Parameter		**BSW Type**
ComGwSignal		ParamConfContainerDef
BSW Description		
This container allows specifying a gateway source or destination respectively with a reference to a ComSignal, a ComGroupSignal or a ComSignalGroup.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the ISignal for which the ISignalTriggering is defined.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalTriggering.signal		
Mapping Rule		**Mapping Type**
Create Container if ISignal is referenced from Gateway::SignalMapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComGwMapping/ComGwSource/ComGwSignal	
BSW Parameter		**BSW Type**
ComGwSignalRef		ChoiceReferenceParamDef
BSW Description		
Reference to an object of a gateway relation. Either to a ComSignal, ComGroupSignal or to a SignalGroup.		
M2 Template	**M2 Description**	
SystemTemplate	An ISignalToIPduMapping describes the mapping of ISignals to ISignalIPdus and defines the position of the ISignal within an ISignalIPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping		
Mapping Rule		**Mapping Type**
System Template element ISingalToIPduMapping represents the ComSignal (SystemSignal that is referenced by an ISignal), ComSignalGroup (SystemSignal Group that is referenced by an ISignal) or ComGroupSignal (SystemSignal that is part of a SystemSignalGroup)		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComGwMapping/ComGwSource	
BSW Parameter		**BSW Type**

ComGwSourceDescription		ParamConfContainerDef
BSW Description		
Description of a gateway source. This container allows defining a gateway source without the configuration of a complete COM signal. This allows adding / changing gateway relations post build without the configuration of new signals.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the ISignal for which the ISignalTriggering is defined.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalTriggering.signal		
Mapping Rule		**Mapping Type**
Informations can be derived from ISignalToIPduMapping element via the ISignal that is referenced by the source ISignalTriggering within the SignalMapping,		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComGwMapping/ComGwSource/ComGwSourceDescription	
BSW Parameter		**BSW Type**
ComBitPosition		IntegerParamDef
BSW Description		
Starting position within the I-PDU. This parameter refers to the position in the I-PDU and not in the shadow buffer.		
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.startPosition		
Mapping Rule		**Mapping Type**
The ISignalToIPduMapping element contains the startPosition attribute. To describe the mapping of a Signal to the IPdu the ISignal element shall refer to a SystemSignal.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComGwMapping/ComGwSource/ComGwSourceDescription	
BSW Parameter		**BSW Type**
ComBitSize		IntegerParamDef
BSW Description		
Size in bits.		
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/ComGwMapping/ComGwSource/ComGwSourceDescription	
BSW Parameter		**BSW Type**
ComGwIPduRef		SymbolicNameReferenceParamDef
BSW Description		
Symbolic reference to an I-PDU of a Signal Gateway source or destination description.		
M2 Template	**M2 Description**	

SystemTemplate	Reference to the ISignal for which the ISignalTriggering is defined.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalTriggering.signal		
Mapping Rule		**Mapping Type**
Create reference for each existing ISignalToIPduMapping that is referenced from the regarded Signal Gateway.		full

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComGwMapping/ComGwSource/ComGwSourceDescription		
BSW Parameter		**BSW Type**	
ComSignalEndianess		EnumerationParamDef	
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 SignalIPdu. The byte ordering "Little Endian" (MostSignificantByteLast) and "Big Endian" (MostSignificantByteFirst) 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/ComGwMapping/ComGwSource/ComGwSourceDescription		
BSW Parameter		**BSW Type**	
ComSignalType		EnumerationParamDef	
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**		
SWComponentTemplate	Abstract base class for user defined (and AUTOSAR predefined) datatypes.		
M2 Parameter			
Datatype::Datatypes::Datatype			
Mapping Rule			**Mapping Type**
Mapping of AUTOSAR data types (defined in the software component description) to COM Signal Types. Mapping rules are described in RTE Specification.			full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComGwMapping/ComGwSource/ComGwSourceDescription	
BSW Parameter		**BSW Type**
ComUpdateBitPosition		IntegerParamDef
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.		
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**
ComIPdu		ParamConfContainerDef
BSW Description		
Contains the configuration parameters of Com I-Pdus. COM174: The shortName is used as the symbolic name (ComIpduName) of this I-Pdu when communicating with the PduR. This parameter is only stored in the XML file, and must not be used within the implementation.		
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		FunctionNameDef
BSW Description		
If there is a 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**
ComIPduGroupRef		ReferenceParamDef
BSW Description		
Reference to all I-PDU groups including this I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to a set of SignalIPdus, which are contained in the I-Pdu Group.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::IPduGroup.iPdu		
Mapping Rule		**Mapping Type**
Find IPduGroup that points to this SignalIPdu and create the reference.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComIPduRxHandleId		IntegerParamDef
BSW Description		
The numerical value used as the ID of this I-PDU. The Com_IPduRxHandleId is required by the API calls to receive I-PDUs from the PduR. It is only present for I-PDU is received from the PduR, because Com is the starting module for Tx I-PDUs and there is no need to define IDs for Tx I-PDUs in the Com module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComIPduSignalGroupRef		ReferenceParamDef
BSW Description		
References to all signal groups contained in this I-Pdu		
M2 Template	**M2 Description**	
SystemTemplate	Reference to a ISignal that is mapped into the SignalIPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.signal		
Mapping Rule		**Mapping Type**
If ISignal that is contained in the SignalIPdu contains a reference to a System SignalGroup than this reference shall be created.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComIPduSignalProcessing		EnumerationParamDef
BSW Description		
For the definition of the two modes Immediate and Defered, see COM298.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComIPduSignalRef		ReferenceParamDef
BSW Description		
References to all signals contained in this I-PDU.		
M2 Template	**M2 Description**	

SystemTemplate	Reference to a ISignal that is mapped into the SignalIPdu.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.signal	
Mapping Rule	**Mapping Type**
If ISignal that is contained in the SignalIPdu contains a reference to a System Signal than this reference shall be created.	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComIPdu
BSW Parameter	**BSW Type**
ComIPduSize	IntegerParamDef
BSW Description	
The size of the I-PDU in bytes. The maximum size is limited by the underlying communication interface. 0-8 for CAN and LIN 0-254 for FlexRay	
M2 Template	**M2 Description**
SystemTemplate	The size of the IPDU in bits.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::IPdu.length	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComIPdu
BSW Parameter	**BSW Type**
ComIPduTriggerTransmitCallout	FunctionNameDef
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**
ComIpduDirection	EnumerationParamDef
BSW Description	
The direction defines if this I-PDU, and therefore the contributing signals and signal groups, shall be send or received.	
M2 Template	**M2 Description**
SystemTemplate	communication Direction of the Connector Port (input or output Port).
M2 Parameter	
Fibex::FibexCore::CoreTopology::CommConnectorPort.communicationDirection	

Mapping Rule	Mapping Type
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.	full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComTxIPdu		ParamConfContainerDef
BSW Description		
This container contains additional transmission related configuration parameters of COM 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 if a SignalIPdu is transmitted by the regarded ECU.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu	
BSW Parameter		**BSW Type**
ComTxIPduClearUpdateBit		EnumerationParamDef
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**
ComTxIPduMinimumDelayTimeFactor		IntegerParamDef
BSW Description		
COM181: Minimum delay between successive transmissions of this I-PDU, independent of the transmission mode. There is only one minimum delay time parameter for the I-PDU. This minimum delay time does not change with mode changes. Neither is the timer reset. This means that mode changes are not allowed to violate the minimum delay time. It is not possible to monitor the minumum delay time for I-PDUs that are requested using the Com_TriggerTransmit API. Depending on the implementation, this timeout may be implemented as a 32-bit or a 16-bit counter. COM471: No minimum delay time monitoring shall take place, if ComTxIPduMinimumDelayTimeFactor is omitted or configured to 0.		
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		IntegerParamDef
BSW Description		
AUTOSAR COM fills not used areas of an I-PDU with this bit-pattern. This attribute is mandatory to avoid undefined behaviour. This byte-pattern will be repeated throughout 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::IPdu.unusedBitPattern		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu	
BSW Parameter		**BSW Type**
ComTxModeFalse		ParamConfContainerDef
BSW Description		
This container contains the configuration parameters of COM transmission modes in the case the ComFilter evaluates to false.		
M2 Template	**M2 Description**	
SystemTemplate	Timing Specification if the COM Transmission Mode is false. The Transmission Mode Selector is defined to be false, if all Conditions evaluate to false.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::Timing::TransmissionModeDeclaration.transmissionModeFalseTiming		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse	
BSW Parameter		**BSW Type**
ComTxMode		ParamConfContainerDef
BSW Description		
This container contains the configuration parameters of COM transmission modes.		
M2 Template	**M2 Description**	
SystemTemplate	AUTOSAR COM provides the possibility to define two different TRANSMISSION MODES (True and False) for each I-PDU. In case only the TRUE transmission mode is used there is no need for the "TransmissionModeDeclaration" and its sub-structure.	
M2 Parameter		

Fibex::FibexCore::CoreCommunication::Timing::TransmissionModeDeclaration	
Mapping Rule	**Mapping Type**
Create Container for each TxMode that is defined for the IPdu.	full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeMode		EnumerationParamDef
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	AUTOSAR COM provides the possibility to define two different TRANSMISSION MODES for each I-PDU.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::IPduTiming		
Mapping Rule		**Mapping Type**
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/ComTxModeFalse/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeNumberOfRepetitions		IntegerParamDef
BSW Description		
Defines the number of repetitions for the Direct/N-Times transmission mode and the event driven part of Mixed transmission mode.		
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::Timing.EventControlledTiming.numberOfRepeats		
Mapping Rule		**Mapping Type**
If "EventControlledTiming.numberOfRepetitions" = 0 then ComTxModeNumber OfRepetitions = 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**
ComTxModeRepetitionPeriodFactor		IntegerParamDef
BSW Description		
Period of the repetition of the n transmission for the Direct/NTimes transmission mode and the event driven part of the Mixed transmission mode. Depending on the implementation, this timeout may be implemented as a 32-bit or a 16-bit counter.		

M2 Template	M2 Description	
SystemTemplate	If the EventControlledTiming is aggregated by the IPduTiming the repetitionPeriod specifies 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::Timing::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**
ComTxModeTimeOffsetFactor		IntegerParamDef
BSW Description		
Time until first transmission of this I-PDU. ComTxModeTimeOffsetFactor defines the time between Com_IpduGroupStart and the first transmission of the cyclic part of this transmission request for this I-PDU. Depending on the implementation, this timeout may be implemented as a 32-bit or a 16-bit counter.		
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::Timing::CyclicTiming.startingTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeTimePeriodFactor		IntegerParamDef
BSW Description		
Period of the repetition of cyclic transmissions. Depending on the implementation, this timeout may be implemented as a 32-bit or a 16-bit counter.		
M2 Template	**M2 Description**	
SystemTemplate	Specification of the repeating cycle.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::Timing::CyclicTiming.repeatingTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu	
BSW Parameter		**BSW Type**
ComTxModeTrue		ParamConfContainerDef
BSW Description		
This container contains the configuration parameters of COM 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 directly by the IPduTiming.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::IPduTiming.cyclicTiming AND/OR Fibex::FibexCore::CoreCommunication::IPduTiming.eventControlledTiming		
Mapping Rule		**Mapping Type**
Create container if CyclicTiming or EventControlledTiming is aggregated by the IPduTiming element.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue	
BSW Parameter		**BSW Type**
ComTxMode		ParamConfContainerDef
BSW Description		
This container contains the configuration parameters of COM transmission modes.		
M2 Template	**M2 Description**	
SystemTemplate	AUTOSAR COM provides the possibility to define two different TRANSMISSION MODES (True and False) for each I-PDU. In case only the TRUE transmission mode is used there is no need for the "TransmissionModeDeclaration" and its sub-structure.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::Timing::TransmissionModeDeclaration		
Mapping Rule		**Mapping Type**
Create Container for each TxMode that is defined for the IPdu.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeMode		EnumerationParamDef
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	AUTOSAR COM provides the possibility to define two different TRANSMISSION MODES for each I-PDU.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::IPduTiming		
Mapping Rule		**Mapping Type**
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		IntegerParamDef
BSW Description		
Defines the number of repetitions for the Direct/N-Times transmission mode and the event driven part of Mixed transmission mode.		

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::Timing.EventControlledTiming.numberOfRepeats		
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**
ComTxModeRepetitionPeriodFactor		IntegerParamDef
BSW Description		
Period of the repetition of the n transmission for the Direct/NTimes transmission mode and the event driven part of the Mixed transmission mode. Depending on the implementation, this timeout may be implemented as a 32-bit or a 16-bit counter.		
M2 Template	**M2 Description**	
SystemTemplate	If the EventControlledTiming is aggregated by the IPduTiming the repetitionPeriod specifies 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::Timing::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**
ComTxModeTimeOffsetFactor		IntegerParamDef
BSW Description		
Time until first transmission of this I-PDU. ComTxModeTimeOffsetFactor defines the time between Com_IpduGroupStart and the first transmission of the cyclic part of this transmission request for this I-PDU. Depending on the implementation, this timeout may be implemented as a 32-bit or a 16-bit counter.		
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::Timing::CyclicTiming.startingTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode

BSW Parameter	BSW Type
ComTxModeTimePeriodFactor	IntegerParamDef
BSW Description	
Period of the repetition of cyclic transmissions. Depending on the implementation, this timeout may be implemented as a 32-bit or a 16-bit counter.	
M2 Template	**M2 Description**
SystemTemplate	Specification of the repeating cycle.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::Timing::CyclicTiming.repeatingTime	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
Com	Com/ComConfig/ComIPdu
BSW Parameter	**BSW Type**
PduIdRef	ReferenceParamDef
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**

BSW Module	BSW Context
Com	Com/ComConfig
BSW Parameter	**BSW Type**
ComIPduGroup	ParamConfContainerDef
BSW Description	
Contains the configuration parameters of Com I-Pdu groups. COM126: The shortName is used as the symbolic name of the I-Pdu group (ComIpduGroup-Name). This parameter is only stored in the XML file, and must not be used within the implementation.	
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::IPduGroup	
Mapping Rule	**Mapping Type**
Create container for each CoreCommunication::ISignalIPduGroup that is referenced by the regarded ECU.	full

BSW Module	BSW Context
Com	Com/ComConfig/ComIPduGroup
BSW Parameter	**BSW Type**
ComIPduGroupGroupRef	ReferenceParamDef
BSW Description	

References to all I-PDU groups including this I-PDU group. If this reference is omitted, this I-PDU group is not included in another I-PDU group.	
M2 Template	**M2 Description**
SystemTemplate	An I-PDU group can be included in other I-Pdu groups.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::IPduGroup.containedIPduGroups	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPduGroup	
BSW Parameter		**BSW Type**
ComIPduGroupHandleId		IntegerParamDef
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. For the rational for the range see COM187.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Com	Com/ComConfig	
BSW Parameter		**BSW Type**
ComSignal		ParamConfContainerDef
BSW Description		
Contains the configuration parameters of Com signals. COM163: The shortName is used as the symbolic name of the signal (ComSignalName). This name is also used as the handle name for the signal. This parameter is only stored in the XML file, and must not be used within the implementation.		
M2 Template	**M2 Description**	
SystemTemplate	An ISignalToIPduMapping describes the mapping of ISignals to ISignalIPdus and defines the position of the ISignal within an ISignalIPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping		
Mapping Rule		**Mapping Type**
A ComSignal must be defined for each ISignalToPduMapping that contains a reference to a SystemSignal that is not part of a SystemSignalGroup and is transmitted or received by the regarded ECU.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComBitPosition		IntegerParamDef
BSW Description		

Starting position within the I-PDU. This parameter refers to the position in the I-PDU and not in the shadow buffer.		
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.startPosition		
Mapping Rule		**Mapping Type**
The ISignalToIPduMapping element contains the startPosition attribute. To describe the mapping of a Signal to the IPdu the ISignal element shall refer to a SystemSignal.		full

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComSignal		
BSW Parameter		**BSW Type**	
ComBitSize		IntegerParamDef	
BSW Description			
Size in bits.			
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		EnumerationParamDef	
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 strategy of handling the reception of invalidValue.		
M2 Parameter			
Communication::UnqueuedReceiverComSpec.handleInvalid			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComErrorNotification		FunctionNameDef
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		ParamConfContainerDef
BSW Description		
This container contains the configuration parameters of COM Filters. Note: On sender side the container is used to specify the transmission mode conditions.		
M2 Template	**M2 Description**	
CommonStructure	Base class for data filters.	
M2 Parameter		
Filter::DataFilter		
Mapping Rule		**Mapping Type**
Data Filters are used in the system description to describe the Transmission ModeConditions. In the SWC Template the DataFilters are aggregated by the ReceiverComSpec.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterAlgorithm		EnumerationParamDef
BSW Description		
The range of values is specified in the [17] specification, chapter 2.2.2, Reception Filtering.		
M2 Template	**M2 Description**	
CommonStructure	Base class for data filters.	
M2 Parameter		
Filter::DataFilter		
Mapping Rule		**Mapping Type**
The algorithm shall be derived from the subclasses of DataFilter.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterMask		IntegerParamDef
BSW Description		
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.		
M2 Template	**M2 Description**	
CommonStructure	mask for old and new value	
M2 Parameter		
Filter::MaskedNewDiffersMaskedOld.mask AND Filter::MaskedNewDiffersX.mask AND Filter::MaskedNewEqualsX.mask		
Mapping Rule		**Mapping Type**
The mask attribute is used by three different algorithms.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterMax		IntegerParamDef
BSW Description		
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.		
M2 Template	**M2 Description**	
CommonStructure	Value to specify the upper boundary	
M2 Parameter		
Filter::NewIsWithin.max AND Filter::NewIsOutside.max		
Mapping Rule		**Mapping Type**
ComFilterMax is used by NewIsWithin and NewIsOutside algorithms.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterMin		IntegerParamDef
BSW Description		
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.		
M2 Template	**M2 Description**	
CommonStructure	Value to specify the lower boundary	
M2 Parameter		
Filter::NewIsWithin.min AND Filter::NewIsOutside.min		
Mapping Rule		**Mapping Type**
ComFilterMin is used by NewIsWithin and NewIsOutside algorithms.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterOffset		IntegerParamDef
BSW Description		
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant. Range = 0..(ComFilterPeriodFactor-1)		
M2 Template	**M2 Description**	
CommonStructure	specifies the initial number of messages to occur before the first message is passed	
M2 Parameter		
Filter::OneEveryN.offset		
Mapping Rule		**Mapping Type**
Offset is used by OneEveryN algorithm.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterPeriodFactor		IntegerParamDef
BSW Description		

The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 are significant.	
M2 Template	**M2 Description**
CommonStructure	specifies number of messages to occur before the message is passed again
M2 Parameter	
Filter::OneEveryN.period	
Mapping Rule	**Mapping Type**
Period is used by OneEveryN algorithm,	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignal/ComFilter
BSW Parameter	**BSW Type**
ComFilterX	IntegerParamDef
BSW Description	
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.	
M2 Template	**M2 Description**
CommonStructure	Value to compare with
M2 Parameter	
Filter::MaskedNewDiffersX.x AND Filter::MaskedNewEqualsX.x	
Mapping Rule	**Mapping Type**
x is used by MaskedNewEqualsX and MaskedNewDiffersX	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignal
BSW Parameter	**BSW Type**
ComFirstTimeoutFactor	IntegerParamDef
BSW Description	
Defines the first timeout period for the deadline monitoring. Details can be found in [17]. Note: See also COM263 for the configuration of the remaining timeout periods. Depending on the implementation, this timeout may be implemented as a 32-bit or a 16-bit counter.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignal
BSW Parameter	**BSW Type**
ComHandleId	IntegerParamDef
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/ComSignal		
BSW Parameter		**BSW Type**	
ComInitialValueOnly		BooleanParamDef	
BSW Description			
If set to true this parameter defines that the respective signal's initial value shall be put into the respective Pdu but there will not be any update of the value through the RTE. Thus the Com implementation does not need to expect any API calls for this signal.			
M2 Template	**M2 Description**		
SystemTemplate	Whether an ECU actually participates in the communication (Tx or Rx) is defined via the association of the TriggeringElements to Port Elements of the CommunicationConnector.		
M2 Parameter			
CoreCommunication::ISignalTriggering.iSignalPort			
Mapping Rule			**Mapping Type**
Tx: If an ISignal has no ISignalPort assigned a ComSignal shall always be created in the transmitting ECUs in order to send the init value. Rx: If an ISignal has no ISignalPort assigned there is no need for the existence of a ComSignal in the rec. Ecu			full

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComSignal		
BSW Parameter		**BSW Type**	
ComInvalidNotification		FunctionNameDef	
BSW Description			
Only valid on receiver side: Name of Com_CbkRxInv 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 ComSignalDataInvalidAction is configured to Notify.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**

Com	Com/ComConfig/ComSignal		
BSW Parameter		**BSW Type**	
ComNotification		FunctionNameDef	
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/ComSignal		
BSW Parameter		**BSW Type**	
ComRxDataTimeoutAction		EnumerationParamDef	
BSW Description			
COM412: This parameter defines the action performed upon a reception timeout violation. COM500: If this parameter is omitted or configured to None no replacement shall take place. COM470: Relating to signals: When this parameter is set to Replace, the replacement value used shall be the ComInitValue. COM513: Relating to signal groups: When this parameter is set to Replace, all included signals shall be set to their ComInitValue.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComSignal		
BSW Parameter		**BSW Type**	
ComSignalDataInvalidValue		IntegerParamDef	
BSW Description			
COM391: On receiver side: When this value is received it is recognized as the invalid value and the appropriate invalid action (as specified by ComDataInvalidAction) is performed. COM501: On sender side: This configures the data invalid value that is used by a call to Com_InvalidateSignal.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

	local

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComSignalEndianess		EnumerationParamDef
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 SignalIPdu. The byte ordering "Little Endian" (MostSignificantByteLast) and "Big Endian" (MostSignificantByteFirst) 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		IntegerParamDef
BSW Description		
COM170: Initial value for this signal. The default value is 0. The lower n-bits of the configured Integer shall be used as init-value for an n-bit sized signal type. COM483: If the signal is of type UINT[n], the Integer's least significant byte shall be assigned to the byte array's last byte. The second-least significant byte shall be assigned to the byte array's last but one byte, and so on.		
M2 Template	**M2 Description**	
SystemTemplate	Optional reference to a SystemSignal's initValue in case the System Description doesn't use a complete Software Component Description. If a full DataMapping exist for the SystemSignal this information may be available from a configured ComSpec.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SystemSignal.initValue		
Mapping Rule		**Mapping Type**
This information may be available in the system description (referenced by the SystemSignal) or in the SWC description (Receiver or Sender COM Spec).		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComSignalLength		IntegerParamDef
BSW Description		
The ComSignalLength specifies the n (in Bytes: 1..8) of the type UINT8[n]. For other types it will be ignored.		
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		EnumerationParamDef
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**	
SWComponentTemplate	Abstract base class for user defined (and AUTOSAR predefined) datatypes.	
M2 Parameter		
Datatype::Datatypes::Datatype		
Mapping Rule		**Mapping Type**
Mapping of AUTOSAR data types (defined in the software component description) to COM Signal Types. Mapping rules are described in RTE Specification.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComTimeoutFactor		IntegerParamDef
BSW Description		
COM263: Defines the timeout period for the deadline monitoring. Details can be found in [17]. Note: The period for the ComFirstTimeoutFactor could differ from the ComTimeoutFactor. Depending on the implementation, this timeout may be implemented as a 32-bit or a 16-bit counter. COM264: If deadline monitoring is used on a signal with an update-bit this defines the time-out for deadline monitoring. COM333: If the timeout is omitted or configured to 0 than no timeout monitoring shall take place. In this case ComFirstTimeoutFactor shall be ignored.		
M2 Template	**M2 Description**	
SystemTemplate	Optional timeout value in seconds for the reception of the ISignal. If a full DataMapping exist for the SystemSignal this information may be available from a configured ReceiverComSpec.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SignalPort.timeout		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComTimeoutNotification		FunctionNameDef
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		EnumerationParamDef
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. TRIGGERED: Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU. PENDING: A write access to this signal never triggers the transmission of the corresponding I-PDU. TRIGGERED_ON_CHANGE: Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU, but only in case the written value is different to the locally stored (last written or init) value. TRIGGERED_WITHOUT_REPETITION: Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU just once without a repetition. TRIGGERED_ON_CHANGE_WITHOUT_REPETITION: Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU just once without a repetition, but only in case the written value is different to the locally stored (last written or init) value.		
M2 Template	**M2 Description**	
SystemTemplate	The triggered or triggeredOnChange 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 IPdu.	
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		IntegerParamDef
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.		
M2 Template	**M2 Description**	

<table>
<tr><td>SystemTemplate</td><td colspan="2">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.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.updateIndicationBitPosition</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>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">SystemTemplateSystemSignalRef</td><td>ForeignReferenceParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Reference to the ISignalToIPduMapping that contains a reference to the ISignal (System Template) which this ComSignal (or ComGroupSignal) represents.</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>Com</td><td colspan="2">Com/ComConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComSignalGroup</td><td>ParamConfContainerDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Contains the configuration parameters of Com signal groups.

COM044: The shortName is used as the symbolic name of the signal group (ComSignalGroupName). This name is also used as the handle name for the signal group. This parameter is only stored in the XML file, and must not be used within the implementation.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">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.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreCommunication::SystemSignalGroup</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">Container shall be created for each SystemSignalGroup that exists in the ECU Extract.</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/ComSignalGroup</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComDataInvalidAction</td><td>EnumerationParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.</td></tr>
</table>

M2 Template	**M2 Description**	
SWComponentTemplate	Specifies strategy of handling the reception of invalidValue.	
M2 Parameter		
Communication::UnqueuedReceiverComSpec.handleInvalid		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup	
BSW Parameter		**BSW Type**
ComErrorNotification		FunctionNameDef
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**
ComFirstTimeoutFactor		IntegerParamDef
BSW Description		
Defines the first timeout period for the deadline monitoring. Details can be found in [17]. Note: See also COM263 for the configuration of the remaining timeout periods. Depending on the implementation, this timeout may be implemented as a 32-bit or a 16-bit counter.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup	
BSW Parameter		**BSW Type**
ComGroupSignal		ParamConfContainerDef
BSW Description		
COM520: This container contains the configuration parameters of group signals. I.e. signals that are included within a signal group. COM521: The shortName is used as the symbolic name of the signal (ComSignalName). This name is also used as the handle name for the signal. This parameter is only stored in the XML file, and must not be used within the implementation.		
M2 Template	**M2 Description**	

SystemTemplate	Reference to the System Signal that is supposed to be transmitted in the ISignal.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::ISignal.systemSignal	
Mapping Rule	**Mapping Type**
Create container for each SystemSignal that is part of a SystemSignalGroup and is mapped into a Pdu.	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal
BSW Parameter	**BSW Type**
ComBitPosition	IntegerParamDef
BSW Description	
Starting position within the I-PDU. This parameter refers to the position in the I-PDU and not in the shadow buffer.	
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.startPosition	
Mapping Rule	**Mapping Type**
The ISignalToIPduMapping element contains the startPosition attribute. To describe the mapping of a Signal to the IPdu the ISignal element shall refer to a SystemSignal.	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal
BSW Parameter	**BSW Type**
ComBitSize	IntegerParamDef
BSW Description	
Size in bits.	
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/ComSignalGroup/ComGroupSignal
BSW Parameter	**BSW Type**
ComFilter	ParamConfContainerDef
BSW Description	
This container contains the configuration parameters of COM Filters. Note: On sender side the container is used to specify the transmission mode conditions.	
M2 Template	**M2 Description**
CommonStructure	Base class for data filters.
M2 Parameter	
Filter::DataFilter	
Mapping Rule	**Mapping Type**

Data Filters are used in the system description to describe the Transmission ModeConditions. In the SWC Template the DataFilters are aggregated by the ReceiverComSpec.	full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterAlgorithm		EnumerationParamDef
BSW Description		
The range of values is specified in the [17] specification, chapter 2.2.2, Reception Filtering.		
M2 Template	**M2 Description**	
CommonStructure	Base class for data filters.	
M2 Parameter		
Filter::DataFilter		
Mapping Rule		**Mapping Type**
The algorithm shall be derived from the subclasses of DataFilter.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterMask		IntegerParamDef
BSW Description		
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.		
M2 Template	**M2 Description**	
CommonStructure	mask for old and new value	
M2 Parameter		
Filter::MaskedNewDiffersMaskedOld.mask AND Filter::MaskedNewDiffersX.mask AND Filter::MaskedNewEqualsX.mask		
Mapping Rule		**Mapping Type**
The mask attribute is used by three different algorithms.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterMax		IntegerParamDef
BSW Description		
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.		
M2 Template	**M2 Description**	
CommonStructure	Value to specify the upper boundary	
M2 Parameter		
Filter::NewIsWithin.max AND Filter::NewIsOutside.max		
Mapping Rule		**Mapping Type**
ComFilterMax is used by NewIsWithin and NewIsOutside algorithms.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter	
BSW Parameter		**BSW Type**

ComFilterMin	IntegerParamDef
BSW Description	
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.	
M2 Template	**M2 Description**
CommonStructure	Value to specify the lower boundary
M2 Parameter	
Filter::NewIsWithin.min AND Filter::NewIsOutside.min	
Mapping Rule	**Mapping Type**
ComFilterMin is used by NewIsWithin and NewIsOutside algorithms.	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter
BSW Parameter	**BSW Type**
ComFilterOffset	IntegerParamDef
BSW Description	
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant. Range = 0..(ComFilterPeriodFactor-1)	
M2 Template	**M2 Description**
CommonStructure	specifies the initial number of messages to occur before the first message is passed
M2 Parameter	
Filter::OneEveryN.offset	
Mapping Rule	**Mapping Type**
Offset is used by OneEveryN algorithm.	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter
BSW Parameter	**BSW Type**
ComFilterPeriodFactor	IntegerParamDef
BSW Description	
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 are significant.	
M2 Template	**M2 Description**
CommonStructure	specifies number of messages to occur before the message is passed again
M2 Parameter	
Filter::OneEveryN.period	
Mapping Rule	**Mapping Type**
Period is used by OneEveryN algorithm,	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter
BSW Parameter	**BSW Type**
ComFilterX	IntegerParamDef
BSW Description	
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.	
M2 Template	**M2 Description**

CommonStructure	Value to compare with	
M2 Parameter		
Filter::MaskedNewDiffersX.x AND Filter::MaskedNewEqualsX.x		
Mapping Rule		**Mapping Type**
x is used by MaskedNewEqualsX and MaskedNewDiffersX		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal	
BSW Parameter		**BSW Type**
ComHandleId		IntegerParamDef
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		IntegerParamDef
BSW Description		
COM391: On receiver side: When this value is received it is recognized as the invalid value and the appropriate invalid action (as specified by ComDataInvalidAction) is performed. COM501: On sender side: This configures the data invalid value that is used by a call to Com_InvalidateSignal.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal	
BSW Parameter		**BSW Type**
ComSignalEndianess		EnumerationParamDef
BSW Description		
Defines the endianness of the signal's network representation.		
M2 Template	**M2 Description**	

<table>
<tr><td>SystemTemplate</td><td colspan="2">This parameter defines the order of the bytes of the signal and the packing into the SignalIPdu. The byte ordering "Little Endian" (MostSignificantByteLast) and "Big Endian" (MostSignificantByteFirst) can be selected.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.packingByteOrder</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>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">ComSignalInitValue</td><td colspan="2">IntegerParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">COM170: Initial value for this signal. The default value is 0.
The lower n-bits of the configured Integer shall be used as init-value for an n-bit sized signal type.

COM483: If the signal is of type UINT[n], the Integer's least significant byte shall be assigned to the byte array's last byte. The second-least significant byte shall be assigned to the byte array's last but one byte, and so on.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">Optional reference to a SystemSignal's initValue in case the System Description doesn't use a complete Software Component Description. If a full DataMapping exist for the SystemSignal this information may be available from a configured ComSpec.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">Fibex::FibexCore::CoreCommunication::SystemSignal.initValue</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">This information may be available in the system description (referenced by the SystemSignal) or in the SWC description (Receiver or Sender COM Spec).</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</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComSignalLength</td><td colspan="2">IntegerParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The ComSignalLength specifies the n (in Bytes: 1..8) of the type UINT8[n]. For other types it will be ignored.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SWComponentTemplate</td><td colspan="3">The number of bits that are used to make up the opaque type.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">Datatype::Datatypes::OpaqueType.numberOfBits</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Opaque data shall always be of uint8[n] and shall always be mapped to an n-bytes sized signal.</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/ComSignalGroup/ComGroupSignal</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComSignalType</td><td>EnumerationParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
</table>

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**	
SWComponentTemplate	Abstract base class for user defined (and AUTOSAR predefined) datatypes.	
M2 Parameter		
Datatype::Datatypes::Datatype		
Mapping Rule		**Mapping Type**
Mapping of AUTOSAR data types (defined in the software component description) to COM Signal Types. Mapping rules are described in RTE Specification.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal	
BSW Parameter		**BSW Type**
ComTransferProperty		EnumerationParamDef
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. PENDING: a change of the value of this group signal shall not be considered in the evaluation of the signal groups ComTransferProperty. TRIGGERED_ON_CHANGE: a change of the value of this group signal shall be considered in the in the evaluation of the signal groups ComTransferProperty.		
M2 Template	**M2 Description**	
SystemTemplate	The triggered or triggeredOnChange 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 IPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.transferProperty		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal	
BSW Parameter		**BSW Type**
SystemTemplateSystemSignalRef		ForeignReferenceParamDef
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		
BSW Parameter		**BSW Type**	
ComHandleId		IntegerParamDef	
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**	
ComInitialValueOnly		BooleanParamDef	
BSW Description			
If set to true this parameter defines that the initial values of all group signals of the ComSignalGroup shall be put into the respective Pdu but there will not be any update of the values through the RTE. Thus the Com implementation does not need to expect any API calls for this signal group.			
M2 Template	**M2 Description**		
SystemTemplate	Whether an ECU actually participates in the communication (Tx or Rx) is defined via the association of the TriggeringElements to Port Elements of the CommunicationConnector.		
M2 Parameter			
CoreCommunication::ISignalTriggering.iSignalPort			
Mapping Rule			**Mapping Type**
Tx: If an ISignal has no ISignalPort assigned a ComSignal shall always be created in the transmitting ECUs in order to send the init value. Rx: If an ISignal has no ISignalPort assigned there is no need for the existence of a ComSignal in the rec. Ecu			full

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComSignalGroup		
BSW Parameter		**BSW Type**	
ComInvalidNotification		FunctionNameDef	
BSW Description			
Only valid on receiver side: Name of Com_CbkRxInv 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 ComSignalDataInvalidAction 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	FunctionNameDef
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	EnumerationParamDef
BSW Description	
COM412: This parameter defines the action performed upon a reception timeout violation. COM500: If this parameter is omitted or configured to None no replacement shall take place. COM470: Relating to signals: When this parameter is set to Replace, the replacement value used shall be the ComInitValue. COM513: Relating to signal groups: When this parameter is set to Replace, all included signals shall be set to their ComInitValue.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
Com	Com/ComConfig/ComSignalGroup
BSW Parameter	**BSW Type**
ComTimeoutFactor	IntegerParamDef
BSW Description	

COM263: Defines the timeout period for the deadline monitoring. Details can be found in [17]. Note: The period for the ComFirstTimeoutFactor could differ from the ComTimeoutFactor. Depending on the implementation, this timeout may be implemented as a 32-bit or a 16-bit counter. COM264: If deadline monitoring is used on a signal with an update-bit this defines the time-out for deadline monitoring. COM333: If the timeout is omitted or configured to 0 than no timeout monitoring shall take place. In this case ComFirstTimeoutFactor shall be ignored.		
M2 Template	**M2 Description**	
SystemTemplate	Optional timeout value in seconds for the reception of the ISignal. If a full DataMapping exist for the SystemSignal this information may be available from a configured ReceiverComSpec.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SignalPort.timeout		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup	
BSW Parameter		**BSW Type**
ComTimeoutNotification		FunctionNameDef
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		EnumerationParamDef
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. TRIGGERED: Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU. PENDING: A write access to this signal never triggers the transmission of the corresponding I-PDU. TRIGGERED_ON_CHANGE: Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU, but only in case the written value is different to the locally stored (last written or init) value. TRIGGERED_WITHOUT_REPETITION: Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU just once without a repetition. TRIGGERED_ON_CHANGE_WITHOUT_REPETITION: Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU just once without a repetition, but only in case the written value is different to the locally stored (last written or init) value.		
M2 Template	**M2 Description**	
SystemTemplate	The triggered or triggeredOnChange 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 IPdu.	
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		IntegerParamDef
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.		
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/ComSignalGroup	
BSW Parameter		**BSW Type**

SystemTemplateSignalGroupRef		ForeignReferenceParamDef	
BSW Description			
Reference to the ISignalToIPduMapping that contains a reference to the ISignal (SystemTemplate) which this ComSignalGroup represents.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	**BSW Context**		
Com	Com		
BSW Parameter		**BSW Type**	
ComGeneral		ParamConfContainerDef	
BSW Description			
Contains the general configuration parameters of the Com module.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Com	Com/ComGeneral		
BSW Parameter		**BSW Type**	
ComConfigurationTimeBase		FloatParamDef	
BSW Description			
If no "ComTimeBase" container is specified this value defines the period between successive calls to the Main Functions (Rx, Tx, Routing) of AUTOSAR COM in seconds. In this case this "ComConfigurationTimeBase" value is used by the COM generator to take the values of the Tx/Rx/Routing 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. If additionally a "ComTimeBase" container and its mandatory parameters ComTxTimeBase, ComRxTimeBase and ComGwTimeBase are specified this value defines the time base for all timing configuration parameters (which are specified as factors of this "ComConfigurationTimeBase"). In this case the time base parameters from "ComTimeBase" container are used by the COM generator to transform the values of the Tx/Rx/Routing related timing configuration parameters of the COM module to internal implementation specific counter or tick values to support the possibly different time bases for Tx/Rx/Routing. The COM module's internal timing handling is implementation specific.			
M2 Template	**M2 Description**		
SystemTemplate	The COM scheduling time is used in order to be able to calculate the worst case bus timing. The processing period shall be specified AUTOSAR conform in seconds.		
M2 Parameter			
Fibex::FibexCore::CoreTopology::EcuInstance.comProcessingPeriod			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Com	Com/ComGeneral		
BSW Parameter		**BSW Type**	
ComConfigurationUseDet		BooleanParamDef	
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**	
ComRetryFailedTransmitRequests		BooleanParamDef	
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		IntegerParamDef	
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**
ComTimeBase		ParamConfContainerDef
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/ComGeneral/ComTimeBase	
BSW Parameter		**BSW Type**
ComGwTimeBase		FloatParamDef
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	Optional signal Routing processing period of the COM scheduling in order to be able to calculate the worst case bus timing. Only applicable if a different processing period for Tx and Routing shall be respected.	
M2 Parameter		
Fibex::FibexCore::CoreTopology::EcuInstance.comProcessingPeriodGw		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComGeneral/ComTimeBase	
BSW Parameter		**BSW Type**
ComRxTimeBase		FloatParamDef
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	Optional Rx processing period of the COM scheduling in order to be able to calculate the worst case bus timing. Only applicable if a different processing period for Tx and Rx shall be respected. If not present the "comProcessingPeriod" attribute shall be	
M2 Parameter		
Fibex::FibexCore::CoreTopology::EcuInstance.comProcessingPeriodRx		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComGeneral/ComTimeBase	
BSW Parameter		**BSW Type**
ComTxTimeBase		FloatParamDef
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 COM scheduling time is used in order to be able to calculate the worst case bus timing. The processing period shall be specified AUTOSAR conform in seconds.	
M2 Parameter		
Fibex::FibexCore::CoreTopology::EcuInstance.comProcessingPeriod		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComGeneral	
BSW Parameter		**BSW Type**
ComUserCbkHeaderFile		StringParamDef
BSW Description		
Defines header files for callback functions which shall be included by the COM module. The minimum length for this StringParamDef is 1. The maximum length for this StringParamDef is 32.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Com	Com/ComGeneral	
BSW Parameter		**BSW Type**
ComVersionInfoApi		BooleanParamDef
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

9.2 Pdu Router Mapping

BSW Module	**BSW Context**		
PduR	PduR		
BSW Parameter		**BSW Type**	
PduRGeneral		ParamConfContainerDef	
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**	
PduRCanIfSupport		BooleanParamDef	
BSW Description			
Configuration parameter to enable or disable PDU Router support for CAN interface.			
M2 Template	**M2 Description**		
SystemTemplate	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			
Fibex::Fibex4Can::CanCommunication::CanFrameTriggering			
Mapping Rule			**Mapping Type**
If the regarded ECU is connected to a CANCluster and the ecu extract contains CanFrameTriggerings than set this parameter to true.			local

BSW Module	**BSW Context**		
PduR	PduR/PduRGeneral		
BSW Parameter		**BSW Type**	
PduRCanTpCancelReceive		BooleanParamDef	
BSW Description			
Specifies if the Can 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/PduRGeneral		
BSW Parameter		**BSW Type**	
PduRCanTpChangeParameterRequestApi		BooleanParamDef	

BSW Description	
This parameter, if set to true, enables the PduR_CanTpChangeParameterRequest Api for CanTp.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRCanTpSupport		BooleanParamDef
BSW Description		
Configuration parameter to enable or disable PDU Router support for CAN TP.		
M2 Template	**M2 Description**	
SystemTemplate	This is a PDU of the Transport Layer. The main purpose of the TP Layer is to segment and reassemble I-PDUs.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::NPdu		
Mapping Rule		**Mapping Type**
If there exists a CAN frame (transmitted or received from ECU under consideration) in which an N-PDU is transmitted then set PduRCanTpSupport = ON		full

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRComSupport		BooleanParamDef
BSW Description		
Configuration parameter to enable or disable PDU Router support for COM.		
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**
If a ISignalIPdu exists in the Ecu Extract than set this parameter to true (transmitted or received from ECU under consideration).		full

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRDcmSupport		BooleanParamDef
BSW Description		
Configuration parameter to enable or disable PDU Router support for DCM.		
M2 Template	**M2 Description**	
SystemTemplate	Represents the I-PDU's handled by Dcm.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::DcmIPdu		

Mapping Rule	Mapping Type
If a Dcm I-Pdu exists in the ecu extract than set this parameter to true (transmitted or received from ECU under consideration).	full

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRDevErrorDetect		BooleanParamDef
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	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRFifoTxBufferSupport		BooleanParamDef
BSW Description		
Configuration parameter to enable or disable PDU Router support for FIFOs as PDU transmit buffers; if PDUR_GATEWAY_OPERATION is disabled, this parameter has to be disabled.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRFrIfSupport		BooleanParamDef
BSW Description		
Configuration parameter to enable or disable PDU Router support for FlexRay interface.		
M2 Template	**M2 Description**	
SystemTemplate	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		
Fibex::Fibex4Flexray::FlexrayCommunication::FlexrayFrameTriggering		
Mapping Rule		**Mapping Type**
If the regarded ECU is connected to a FlexRayCluster and the ecu extract contains FlexrayFrameTriggerings than set this parameter to true.		full

BSW Module	BSW Context
PduR	PduR/PduRGeneral

<table>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">PduRFrTpCancelReceive</td><td colspan="2">BooleanParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Specifies if the Flaxray Transport protocol module supports the CancelReceive API or not. Value true the API is supported.</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>PduR</td><td colspan="3">PduR/PduRGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">PduRFrTpChangeParameterRequestApi</td><td colspan="2">BooleanParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This parameter, if set to true, enables the PduR_FrTpChangeParameterRequest Api for FrTp.</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>PduR</td><td colspan="3">PduR/PduRGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">PduRFrTpSupport</td><td colspan="2">BooleanParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Configuration parameter to enable or disable PDU Router support for FlexRay TP.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">This is a PDU of the Transport Layer.
The main purpose of the TP Layer is to segment and reassemble I-PDUs.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">Fibex::FibexCore::CoreCommunication::NPdu</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">If there exists a FlexRay frame (transmitted or received from ECU under consideration)
in which an N-PDU is transmitted then set PduRFrTpSupport = ON</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>PduR</td><td colspan="3">PduR/PduRGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">PduRGatewayOperation</td><td colspan="2">BooleanParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Configuration parameter to enable or disable PDU Router gateway operation; if PDUR_ZERO_COST_OPERATION is enabled, this parameter has to be disabled.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
</table>

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRIPduMSupport		BooleanParamDef
BSW Description		
Configuration parameter to enable or disable PDU Router support for IPDUM; if PDUR_ZERO_COST_OPERATION is enabled, this parameter has to be disabled.		
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**
If an IPdu (transmitted or received from ECU under consideration) contains a mulitplexer than set this parameter to true		full

BSW Module	**BSW Context**	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRLinIfSupport		BooleanParamDef
BSW Description		
Configuration parameter to enable or disable PDU Router support for LIN interface.		
M2 Template	**M2 Description**	
SystemTemplate	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		
Fibex::Fibex4Lin::LinCommunication::LinFrameTriggering		
Mapping Rule		**Mapping Type**
If the regarded ECU is connected to a LinCluster and the ecu extract contains LinFrameTriggerings than set this parameter to true.		full

BSW Module	**BSW Context**	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRLinTpChangeParameterRequestApi		BooleanParamDef
BSW Description		
This parameter, if set to true, enables the PduR_LinTpChangeParameterRequest Api for LinTp.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRLinTpSupport		BooleanParamDef
BSW Description		
Configuration parameter to enable or disable PDU Router support for LIN TP.		
M2 Template	**M2 Description**	
SystemTemplate	This is a PDU of the Transport Layer. The main purpose of the TP Layer is to segment and reassemble I-PDUs.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::NPdu		
Mapping Rule		**Mapping Type**
If there exists a LIN frame (transmitted or received from ECU under consideration) in which an N-PDU is transmitted then set PduRLinTpSupport = ON		full

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRMemorySize		IntegerParamDef
BSW Description		
Memory size reserved for PDU Router buffers. Only required for gateway operation.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRMinimumRoutingLoModule		EnumerationParamDef
BSW Description		
Please note that this parameter is deprecated and will be removed in a future release. Minimum routing has been removed from PduR. Old description: Lower layer module to be used for minimum routing; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**

PduRMinimumRoutingLoRxPduId		IntegerParamDef	
BSW Description			
Please note that this parameter is deprecated and will be removed in a future release. Minimum routing has been removed from PduR. Old description: Receive PDU identifier of the lower layer module which shall be used at the PDU Router interface to the lower layer module specified by PDUR_MINIMUM_ROUTING_LO_MODULE for minimum routing; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
PduR	PduR/PduRGeneral		
BSW Parameter		**BSW Type**	
PduRMinimumRoutingLoTxPduId		IntegerParamDef	
BSW Description			
Please note that this parameter is deprecated and will be removed in a future release. Minimum routing has been removed from PduR. Old description: Transmit PDU identifier of the lower layer module which shall be used at the PDU Router interface to the lower layer module specified by PDUR_MINIMUM_ROUTING_LO_MODULE for minimum routing; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRMinimumRoutingUpModule		EnumerationParamDef
BSW Description		
Please note that this parameter is deprecated and will be removed in a future release. Minimum routing has been removed from PduR. Old description: Upper layer module to be used for minimum routing; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRGeneral
BSW Parameter	**BSW Type**
PduRMinimumRoutingUpRxPduId	IntegerParamDef
BSW Description	
Please note that this parameter is deprecated and will be removed in a future release. Minimum routing has been removed from PduR. Old description: Receive PDU identifier of the upper layer module which shall be used at the PDU Router interface to the upper layer module specified by PDUR_MINIMUM_ROUTING_UP_MODULE for minimum routing; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRGeneral
BSW Parameter	**BSW Type**
PduRMinimumRoutingUpTxPduId	IntegerParamDef
BSW Description	
Please note that this parameter is deprecated and will be removed in a future release. Minimum routing has been removed from PduR. Old description: Transmit PDU identifier of the upper layer module which shall be used at the PDU Router interface to the upper layer module specified by PDUR_MINIMUM_ROUTING_UP_MODULE for minimum routing; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRGeneral
BSW Parameter	**BSW Type**

PduRMulticastFromIfSupport			BooleanParamDef	
BSW Description				
Configuration parameter to enable or disable PDU Router support for multicasts from an interface module to upper layer modules or lower layer interface modules; if PDUR_ZERO_COST_OPERATION is enabled, this parameter has to be disabled.				
M2 Template	**M2 Description**			
M2 Parameter				
Mapping Rule				**Mapping Type**
Description of Fan-In is not supported by the System Template.				local

BSW Module	**BSW Context**		
PduR	PduR/PduRGeneral		
BSW Parameter		**BSW Type**	
PduRMulticastFromTpSupport		BooleanParamDef	
BSW Description			
Configuration parameter to enable or disable PDU Router support for multicasts from a TP module to upper layer modules or lower layer TP modules; if PDUR_ZERO_COST_OPERATION is enabled, this parameter has to be disabled.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
Description of Fan-In is not supported by the System Template.			local

BSW Module	**BSW Context**		
PduR	PduR/PduRGeneral		
BSW Parameter		**BSW Type**	
PduRMulticastToIfSupport		BooleanParamDef	
BSW Description			
Configuration parameter to enable or disable PDU Router support for multicasts from an upper layer module to interface modules; if PDUR_ZERO_COST_OPERATION is enabled, this parameter has to be disabled.			
M2 Template	**M2 Description**		
SystemTemplate	The IPduTriggering describes on which channel the IPdu is transmitted. 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.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::IPduTriggering			
Mapping Rule			**Mapping Type**
The Pdu Router fan-out is described by the PduTriggering. If several PduTriggerings exist for the same IPdu, than set this parameter to true.			full

BSW Module	**BSW Context**	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRMulticastToTpSupport		BooleanParamDef
BSW Description		

Configuration parameter to enable or disable PDU Router support for multicasts from an upper layer module to TP modules; if PDUR_ZERO_COST_OPERATION is enabled, this parameter has to be disabled.		
M2 Template	**M2 Description**	
SystemTemplate	The IPduTriggering describes on which channel the IPdu is transmitted. 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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::IPduTriggering		
Mapping Rule		**Mapping Type**
The Pdu Router fan-out is described by the PduTriggering. If several PduTriggerings exist for the same IPdu, than set this parameter to true.		full

BSW Module	**BSW Context**		
PduR	PduR/PduRGeneral		
BSW Parameter		**BSW Type**	
PduRSbTxBufferSupport		BooleanParamDef	
BSW Description			
Configuration parameter to enable or disable PDU Router support for single buffers as PDU transmit buffers; if PDUR_GATEWAY_OPERATION is disabled, this parameter has to be disabled.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
PduR	PduR/PduRGeneral		
BSW Parameter		**BSW Type**	
PduRSingleIf		EnumerationParamDef	
BSW Description			
Single interface module in case zero cost operation is enabled (PDUR_ZERO_COST_OPERATION).			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRSingleTp		EnumerationParamDef
BSW Description		
Single transport protocol module in case zero cost operation is enabled (PDUR_ZERO_COST_OPERATION).		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRVersionInfoApi		BooleanParamDef
BSW Description		
Activates/Deactivates the Version Info API.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRZeroCostOperation		BooleanParamDef
BSW Description		
All routing paths are implicitly defined and the communication modules directly above or below the PDU Router shall directly call each other without using PDU Router functions (zero cost operation). The configuration parameters PDUR_SINGLE_IF and PDUR_SINGLE_TP are used to specify the related lower layer module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR	
BSW Parameter		**BSW Type**
PduRGlobalConfig		ParamConfContainerDef
BSW Description		
This container contains the global configuration parameter of the PduR. It 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	
PduR	PduR/PduRGlobalConfig	
BSW Parameter		**BSW Type**
PduRConfigurationId		IntegerParamDef
BSW Description		
unique configuration identifier of post-build time configuration; this parameter shall be used if PDUR_ZERO_COST_OPERATION is disabled; otherwise it shall not be used.		
M2 Template	**M2 Description**	
SystemTemplate	unique PDURconfiguration identifier	
M2 Parameter		
Fibex::FibexCore::CoreTopology::EcuInstance.pduRConfigurationId		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
PduR	PduR/PduRGlobalConfig	
BSW Parameter		**BSW Type**
PduRRoutingPathGroup		ParamConfContainerDef
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 IPdus during runtime according to the AUTOSAR PduR specification.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::PdurIPduGroup		
Mapping Rule		**Mapping Type**
shall be derived from PdurIPduGroup element.		full

BSW Module	BSW Context	
PduR	PduR/PduRGlobalConfig/PduRRoutingPathGroup	
BSW Parameter		**BSW Type**
PduRDestPduRef		ReferenceParamDef
BSW Description		
This reference selects one destination of the routing path.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to a set of IPdus, which are contained in the PduR I-Pdu Group. To enable/disable a specific destination the PdurIPduGroup refers to the PduTriggering.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::PdurIPduGroup.iPdu		
Mapping Rule		**Mapping Type**
The Pdus of a PduRRoutingPathGroup shall be derived from the PdurIPdu Group-PduTriggering reference.		full

BSW Module	BSW Context
PduR	PduR/PduRGlobalConfig/PduRRoutingPathGroup

<table>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">PduRIsEnabledAtInit</td><td colspan="2">BooleanParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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).</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>PduR</td><td colspan="3">PduR/PduRGlobalConfig/PduRRoutingPathGroup</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">PduRRoutingPathGroupId</td><td colspan="2">IntegerParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Identification of the routing group.

The identification will be used by the disable/enable API in the PDU Router module API.</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>PduR</td><td colspan="3">PduR/PduRGlobalConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">PduRRoutingTable</td><td colspan="2">ParamConfContainerDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">PDU Router routing table is a subcontainer ofPduR. This container shall only be considered by the PDU Router Configuration Generator if PduRGeneral/PDUR_ZERO_COST_OPERATION is disabled.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">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.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">Fibex::FibexCore::CoreCommunication::IPdu</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">If at least one CoreCommunication::IPdu exist and if minimum routing is not used than create this container.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>PduR</td><td colspan="3">PduR/PduRGlobalConfig/PduRRoutingTable</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">PduRRoutingPath</td><td colspan="2">ParamConfContainerDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
</table>

This container is a subcontainer of PduRRoutingTable and specifies the routing path of a PDU.		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
Mapping Rule		**Mapping Type**
The information that is described in this container can be derived from the Communication description (Pdu Triggerings, PduGateway) in the System Template.		local

BSW Module	**BSW Context**	
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath	
BSW Parameter		**BSW Type**
PduRDefaultValue		ParamConfContainerDef
BSW Description		
This container is a subcontainer of PduRRoutingPath and 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 I-Pdu has been received since last sending.	
M2 Parameter		
Fibex::Fibex4Multiplatform::PduMappingDefaultValue		
Mapping Rule		**Mapping Type**
Container must be created if PduMappingDefaulValue is described in the Sys-T		full

BSW Module	**BSW Context**	
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRDefault Value	
BSW Parameter		**BSW Type**
PduRDefaultValueElement		ParamConfContainerDef
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**
1:1 mapping		full

BSW Module	**BSW Context**	
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRDefault Value/PduRDefaultValueElement	
BSW Parameter		**BSW Type**
DefaultValueElement		IntegerParamDef
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 ElementBytePosition 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/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRDefaultValue/PduRDefaultValueElement
BSW Parameter	**BSW Type**
ElementBytePosition	IntegerParamDef
BSW Description	
This parameter specifies the byte position of the element within the default value	
M2 Template	**M2 Description**
SystemTemplate	This attribute specifies the byte position of the element within the default value
M2 Parameter	
Fibex::Fibex4Multiplatform::DefaultValueElement.elementPosition	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath
BSW Parameter	**BSW Type**
PduRDestPdu	ParamConfContainerDef
BSW Description	
This container is a subcontainer of PduRRoutingPath and specifies one destination for the PDU to be routed.	
M2 Template	**M2 Description**
SystemTemplate	The IPduTriggering describes on which channel the IPdu is transmitted.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::IPduTriggering	
Mapping Rule	**Mapping Type**
The destination of a Pdu can be derived from the PduTriggering element. If an IPdu is routed by the PduR to different destinations (PduR fan-out) than an PduTriggering for each destination exists in the System Template.	full

BSW Module	**BSW Context**
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRDestPdu
BSW Parameter	**BSW Type**
DataProvision	EnumerationParamDef
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 gateway PDUs.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRDestPdu	
BSW Parameter		**BSW Type**
DestPduRef		ReferenceParamDef
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/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRDestPdu	
BSW Parameter		**BSW Type**
TxBufferRef		ReferenceParamDef
BSW Description		
Reference to a buffer that is allocated in the PduRTxBufferTable. This buffer is required for communication interface gatewaying, and for transport protocol gatewaying for single frame routing.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath	
BSW Parameter		**BSW Type**
PduRSrcPdu		ParamConfContainerDef
BSW Description		
This container is a subcontainer of PduRRoutingPath and specifies the source of the PDU to be routed.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::IPdu		
Mapping Rule		**Mapping Type**
This information can be derived from the communication description (IPdus and PduTriggerings).		full

BSW Module	**BSW Context**	
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRSrcPdu	
BSW Parameter		**BSW Type**
HandleId		IntegerParamDef
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/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRSrcPdu		
BSW Parameter		**BSW Type**	
SrcPduRef		ReferenceParamDef	
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/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath		
BSW Parameter		**BSW Type**	
PduRTpRxBlockSize		IntegerParamDef	
BSW Description			
This parameter defines the block size to be used by the receiving TP module.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath		
BSW Parameter		**BSW Type**	
SduLength		IntegerParamDef	
BSW Description			
Length of PDU data (SDU). Only required if a TX buffer is configured.			
M2 Template	**M2 Description**		
SystemTemplate	The size of the IPDU in bits.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::IPdu.length			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**

PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath
BSW Parameter	**BSW Type**
TpChunkSize	IntegerParamDef
BSW Description	
Chunk size for routing on the fly. Defines the number of bytes which shall be received before transmission on the destination bus may start. Only required for TP gateway PDUs. The TpChunkSize 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 this routing relation.
M2 Parameter	
Fibex::Fibex4Multiplatform::IPduMapping.pdurTpChunkSize	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
PduR	PduR/PduRGlobalConfig
BSW Parameter	**BSW Type**
PduRTpBufferTable	ParamConfContainerDef
BSW Description	
This container is a subcontainer of PduR and contains the definition of all TP buffers (only required for PDU Router gateway operation). This container shall only be considered by the PDU Router Configuration Generator if PduRGeneral/PDUR_GATEWAY_OPERATION is enabled.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRGlobalConfig/PduRTpBufferTable
BSW Parameter	**BSW Type**
PduRMaxTpBufferNumber	IntegerParamDef
BSW Description	
maximum number of TP buffers.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRGlobalConfig/PduRTpBufferTable
BSW Parameter	**BSW Type**
PduRTpBuffer	ParamConfContainerDef
BSW Description	
This container is a subcontainer of PduRTpBufferTable and specifies a TP buffer.	
M2 Template	**M2 Description**

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRGlobalConfig/PduRTpBufferTable/PduRTpBuffer
BSW Parameter	**BSW Type**
Length	IntegerParamDef
BSW Description	
Length of the buffer.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRGlobalConfig
BSW Parameter	**BSW Type**
PduRTxBufferTable	ParamConfContainerDef
BSW Description	
This container contains the buffers used for gatewaying via communication interfaces and for single frames of transport protocols.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRGlobalConfig/PduRTxBufferTable
BSW Parameter	**BSW Type**
PduRMaxTxBufferNumber	IntegerParamDef
BSW Description	
maximum number of transmit buffers	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRGlobalConfig/PduRTxBufferTable
BSW Parameter	**BSW Type**

PduRTxBuffer		ParamConfContainerDef	
BSW Description			
This container is a subcontainer of PduRTxBufferTable and specifies a transmit buffer for a non-TP PDU.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
PduR	PduR/PduRGlobalConfig/PduRTxBufferTable/PduRTxBuffer		
BSW Parameter		**BSW Type**	
Depth		IntegerParamDef	
BSW Description			
Specifies the depth of the buffer. For TP single frames, the depth is always 1.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
PduR	PduR/PduRGlobalConfig/PduRTxBufferTable/PduRTxBuffer		
BSW Parameter		**BSW Type**	
Length		IntegerParamDef	
BSW Description			
Length of the buffer. When this buffer is used for TP routing path the Length has to be large enough to contain the largest possible single frame of the source network.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

9.3 IPdu Multiplexer Mapping

BSW Module	**BSW Context**	
IpduM	IpduM	
BSW Parameter		**BSW Type**
IpduMConfig		ParamConfContainerDef
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		ParamConfContainerDef
BSW Description		
Contains the configuration parameters received I-PDUs by the IpduM module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMRxPathway	
BSW Parameter		**BSW Type**
IpduMRxIndication		ParamConfContainerDef
BSW Description		
Contains the configuration for incoming RxIndication calls.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication	
BSW Parameter		**BSW Type**
IpduMBitField		ParamConfContainerDef

BSW Description		
This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.		
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**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMBitField	
BSW Parameter		**BSW Type**
IpduMEndBit		IntegerParamDef
BSW Description		
Position of the end bit in the I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMBitField	
BSW Parameter		**BSW Type**
IpduMStartBit		IntegerParamDef
BSW Description		
Position of the start bit in the I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication	
BSW Parameter		**BSW Type**
IpduMByteOrder		EnumerationParamDef
BSW Description		

<table>
<tr><td colspan="3">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.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">This attribute defines the order of the bytes of the selectorField and the packing into the MultiplexedIPdu. The byte ordering "Little Endian" (MostSignificantByte-Last) and "Big Endian" (MostSignificantByteFirst) can be selected.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreCommunication::SegmentPosition.segmentByteOrder AND Fibex::FibexCore::CoreCommunication::MultiplexedIPdu.selectorFieldByteOrder</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">segmentByteOrder and selectorFieldByteOrder shall have the same value. The IPduMByteOrder parameter shall be filled with this value.</td><td>full</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/IpduMRxPathway/IpduMRxIndication</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">IpduMRxDynamicPart</td><td>ParamConfContainerDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This container contains the configuration for the dynamic part of incoming RxIndication calls. When an incoming received I-PDU's selector field matches the IpduMRxSelectorValue, 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.

In case no dynamic part shall be extracted from this received I-PDU this container does not exist. This use-case can occur in case a MultiplexedIPdu is received by an ECU which is only interested in the static part of the MultiplexedIPdu.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">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.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreCommunication::DynamicPart</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>IpduM</td><td colspan="2">IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxDynamicPart</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">IpduMCopyBitField</td><td>ParamConfContainerDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Specifies the source bit fields and the destination bit position, so that the bits in the source can be copied to the bits in the destination.
Within one I-PDU multiple instances of this container are used to specify the bit fields in that I-PDU. Adjacent bit fields could be merged in order to reduce the number of instances of this container.</td></tr>
</table>

M2 Template	M2 Description	
SystemTemplate	Segments bit position relatively to the beginning of a multiplexed IPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	BSW Context		
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxDynamicPart/IpduMCopyBitField		
BSW Parameter		**BSW Type**	
IpduMBitField		ParamConfContainerDef	
BSW Description			
This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.			
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**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.			full

BSW Module	BSW Context		
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxDynamicPart/IpduMCopyBitField/IpduMBitField		
BSW Parameter		**BSW Type**	
IpduMEndBit		IntegerParamDef	
BSW Description			
Position of the end bit in the I-PDU.			
M2 Template	**M2 Description**		
SystemTemplate	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.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::SegmentPosition			
Mapping Rule			**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.			full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxDynamicPart/IpduMCopyBitField/IpduMBitField	
BSW Parameter		**BSW Type**
IpduMStartBit		IntegerParamDef
BSW Description		
Position of the start bit in the I-PDU.		

M2 Template	M2 Description	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxDynamicPart/IpduMCopyBitField	
BSW Parameter		**BSW Type**
IpduMDestinationBit		IntegerParamDef
BSW Description		
Bit position in an I-PDU of the start of the destination bit field for the copy. The resulting destination field must fit inside the I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	Segments bit position relatively to the beginning of a multiplexed IPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition.segmentPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxDynamicPart	
BSW Parameter		**BSW Type**
IpduMOutgoingDynamicPduRef		ReferenceParamDef
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**	
SystemTemplate	Reference to a Com IPdu which is routed to the IPduM module and is combined to a multiplexedPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::DynamicPartAlternative.iPdu		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxDynamicPart	
BSW Parameter		**BSW Type**
IpduMRxSelectorValue		IntegerParamDef
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 dynamic 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
BSW Parameter	**BSW Type**
IpduMRxHandleId	IntegerParamDef
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	ReferenceParamDef
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	ParamConfContainerDef
BSW Description	
This container contains the information on how to unpack the static part of an incoming I-PDU.	
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**
1:1 mapping	full

BSW Module	**BSW Context**

IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxStaticPart		
BSW Parameter		**BSW Type**	
IpduMCopyBitField		ParamConfContainerDef	
BSW Description			
Specifies the source bit fields and the destination bit position, so that the bits in the source can be copied to the bits in the destination. Within one I-PDU multiple instances of this container are used to specify the bit fields in that I-PDU. Adjacent bit fields could be merged in order to reduce the number of instances of this container.			
M2 Template	**M2 Description**		
SystemTemplate	Segments bit position relatively to the beginning of a multiplexed IPdu.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::SegmentPosition			
Mapping Rule			**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.			full

BSW Module	**BSW Context**		
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxStaticPart/IpduMCopyBitField		
BSW Parameter		**BSW Type**	
IpduMBitField		ParamConfContainerDef	
BSW Description			
This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.			
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**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.			full

BSW Module	**BSW Context**		
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxStaticPart/IpduMCopyBitField/IpduMBitField		
BSW Parameter		**BSW Type**	
IpduMEndBit		IntegerParamDef	
BSW Description			
Position of the end bit in the I-PDU.			
M2 Template	**M2 Description**		
SystemTemplate	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.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::SegmentPosition			
Mapping Rule			**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.			full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxStaticPart/IpduMCopyBitField/IpduMBitField	
BSW Parameter		**BSW Type**
IpduMStartBit		IntegerParamDef
BSW Description		
Position of the start bit in the I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxStaticPart/IpduMCopyBitField	
BSW Parameter		**BSW Type**
IpduMDestinationBit		IntegerParamDef
BSW Description		
Bit position in an I-PDU of the start of the destination bit field for the copy. The resulting destination field must fit inside the I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	Segments bit position relatively to the beginning of a multiplexed IPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition.segmentPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxStaticPart	
BSW Parameter		**BSW Type**
IpduMOutgoingStaticPduRef		ReferenceParamDef
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**	
SystemTemplate	Reference to a Com IPdu which is routed to the IPduM module and is combined to a multiplexedPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::StaticPart.iPdu		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context
IpduM	IpduM/IpduMConfig

BSW Parameter		BSW Type	
IpduMTxPathway		ParamConfContainerDef	
BSW Description			
Contains the configuration parameters transmitted I-PDUs by the IpduM module.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
IpduM	IpduM/IpduMConfig/IpduMTxPathway		
BSW Parameter		**BSW Type**	
IpduMTxConfirmation		ParamConfContainerDef	
BSW Description			
A transmit request can be confirmed by the lower layer. This container is used to generate the matching confirmations for the static and dynamic parts of a multiplexed I-PDU. When an I-PDU is transmitted by the IpduM, the selector field value in that PDU needs to be stored in the IpduM so that the confirmation for the correct dynamic part can be generated. This is state internal to the IpduM at run-time. For the purposes of this container and IpduMDynamicTxConfirmation this stored state is called Stored_Selector.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxConfirmation		
BSW Parameter		**BSW Type**	
IpduMDynamicTxConfirmation		ParamConfContainerDef	
BSW Description			
The dynamic part of an I-PDU can have more than one I-PDU IDs for confirmations. The correct I-PDU ID for the confirmation is found from the selector field value of a previously transmitted I-PDU. It is assumed that this selector field is stored in some internal value called Stored_Selector. When a transmit confirmation is received the Stored_Selector is used to select an instance of IpduMDynamicTxConfirmation by matching the Stored_Selector with the IpduMSelectorValue.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxConfirmation/IpduMDynamicTx Confirmation	
BSW Parameter		**BSW Type**
IpduMDynamicTxConfirmIPduRef		ReferenceParamDef

BSW Description	
This is the I-PDU ID to use in the outgoing confirmation (confirmation for the COM I-PDU) when an incoming confirmation (for an IpduM I-PDU) is received and matches the stored Stored_Selector.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxConfirmation/IpduMDynamicTxConfirmation
BSW Parameter	**BSW Type**
IpduMSelectorValue	IntegerParamDef
BSW Description	
When the selector field of the confirmed I-PDU matches the value in here then generate a TxConfirmation for the I-PDU referenced by IpduMDynamicTxConfirmIPduRef.	
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 dynamic 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/IpduMTxPathway/IpduMTxConfirmation
BSW Parameter	**BSW Type**
IpduMStaticTxConfirmationIPduRef	ReferenceParamDef
BSW Description	
This references the I-PDU to use in the TxConfirmation for the static part. This entity does not appear if there is no static part.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
IpduM	IpduM/IpduMConfig/IpduMTxPathway
BSW Parameter	**BSW Type**
IpduMTxRequest	ParamConfContainerDef
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**

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest	
BSW Parameter		**BSW Type**
IpduMBitField		ParamConfContainerDef
BSW Description		
This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.		
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**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMBitField	
BSW Parameter		**BSW Type**
IpduMEndBit		IntegerParamDef
BSW Description		
Position of the end bit in the I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMBitField	
BSW Parameter		**BSW Type**
IpduMStartBit		IntegerParamDef
BSW Description		
Position of the start bit in the I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**

The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.	full

BSW Module	**BSW Context**
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest
BSW Parameter	**BSW Type**
IpduMByteOrder	EnumerationParamDef
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**
SystemTemplate	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.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::SegmentPosition.segmentByteOrder AND Fibex::FibexCore::CoreCommunication::MultiplexedIPdu.selectorFieldByteOrder	
Mapping Rule	**Mapping Type**
segmentByteOrder and selectorFieldByteOrder shall have the same value. The IPduMByteOrder parameter shall be filled with this value.	full

BSW Module	**BSW Context**
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest
BSW Parameter	**BSW Type**
IpduMIPduUnusedAreasDefault	IntegerParamDef
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::IPdu.unusedBitPattern	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest
BSW Parameter	**BSW Type**
IpduMInitialSelectorValue	IntegerParamDef
BSW Description	
This value is used by the initialization function to set the initial value of the selector field.	

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 dynamic part of the IPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::DynamicPartAlternative.selectorFieldCode		
Mapping Rule		**Mapping Type**
If the initialDynamicPart attribute in the DynamicPartAlternative is set to true than the selectorFieldCode of the same DynamicPartAlternative shall be used.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest	
BSW Parameter		**BSW Type**
IpduMOutgoingPduRef		ReferenceParamDef
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**
IpduMSize		IntegerParamDef
BSW Description		
The size of the I-PDU in bytes. The maximum size is limited by the underlying communication interface. 0-8 for CAN and LIN 0-254 for FlexRay		
M2 Template	**M2 Description**	
SystemTemplate	The size of the IPDU in bits.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::IPdu.length		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest	
BSW Parameter		**BSW Type**
IpduMTxConfirmationTimeout		FloatParamDef
BSW Description		
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.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest	
BSW Parameter		**BSW Type**
IpduMTxDynamicPart		ParamConfContainerDef
BSW Description		
Configuration parameters for an instance of a TxRequest call into the IpduM. When a Tx Request with the IpduMTxDynamicHandleId is received by the IpduM, the bit fields in the incoming I-PDU are packed into the outgoing I-PDU buffer and then the send mode honored. 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.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::DynamicPart		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart	
BSW Parameter		**BSW Type**
IpduMCopyBitField		ParamConfContainerDef
BSW Description		
Specifies the source bit fields and the destination bit position, so that the bits in the source can be copied to the bits in the destination. Within one I-PDU multiple instances of this container are used to specify the bit fields in that I-PDU. Adjacent bit fields could be merged in order to reduce the number of instances of this container.		
M2 Template	**M2 Description**	
SystemTemplate	Segments bit position relatively to the beginning of a multiplexed IPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart/ IpduMCopyBitField	
BSW Parameter		**BSW Type**
IpduMBitField		ParamConfContainerDef
BSW Description		
This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.		
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**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.	full

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart/ IpduMCopyBitField/IpduMBitField	
BSW Parameter		**BSW Type**
IpduMEndBit		IntegerParamDef
BSW Description		
Position of the end bit in the I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart/ IpduMCopyBitField/IpduMBitField	
BSW Parameter		**BSW Type**
IpduMStartBit		IntegerParamDef
BSW Description		
Position of the start bit in the I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart/ IpduMCopyBitField	
BSW Parameter		**BSW Type**
IpduMDestinationBit		IntegerParamDef
BSW Description		
Bit position in an I-PDU of the start of the destination bit field for the copy. The resulting destination field must fit inside the I-PDU.		

M2 Template	M2 Description
SystemTemplate	Segments bit position relatively to the beginning of a multiplexed IPdu.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::SegmentPosition.segmentPosition	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart
BSW Parameter	**BSW Type**
IpduMJitUpdate	BooleanParamDef
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**
IpduMTxDynamicHandleId	IntegerParamDef
BSW Description	
This is an incoming handle id. When the handle of an incoming Tx Request matches this, the bits fields (see IpduM_CopyBitField) 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	ReferenceParamDef
BSW Description	
Reference to the Pdu representation in the ECU Configuration Description exchange file to be transmitted.	
M2 Template	**M2 Description**
SystemTemplate	Reference to a Com IPdu which is routed to the IPduM module and is combined to a multiplexedPdu.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::DynamicPartAlternative.iPdu	
Mapping Rule	**Mapping Type**
1:1 mapping	full

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>IpduM</td><td colspan="3">IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IpduMTxStaticPart</td><td colspan="2">ParamConfContainerDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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, the bit fields in the incoming I-PDU are packed into the outgoing I-PDU buffer and then the send mode honored.
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.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">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.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">Fibex::FibexCore::CoreCommunication::StaticPart</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>IpduM</td><td colspan="3">IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IpduMCopyBitField</td><td colspan="2">ParamConfContainerDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Specifies the source bit fields and the destination bit position, so that the bits in the source can be copied to the bits in the destination.
Within one I-PDU multiple instances of this container are used to specify the bit fields in that I-PDU. Adjacent bit fields could be merged in order to reduce the number of instances of this container.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">Segments bit position relatively to the beginning of a multiplexed IPdu.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">Fibex::FibexCore::CoreCommunication::SegmentPosition</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>IpduM</td><td colspan="3">IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart/
IpduMCopyBitField</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IpduMBitField</td><td colspan="2">ParamConfContainerDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">Fibex::FibexCore::CoreCommunication::SegmentPosition</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
</table>

The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.	full

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart/IpduMCopyBitField/IpduMBitField	
BSW Parameter		**BSW Type**
IpduMEndBit		IntegerParamDef
BSW Description		
Position of the end bit in the I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart/IpduMCopyBitField/IpduMBitField	
BSW Parameter		**BSW Type**
IpduMStartBit		IntegerParamDef
BSW Description		
Position of the start bit in the I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
The Static- and the DynamicPart are segmented into parts that are copied into the MultiplexedIPdu. The segment in the source I-PDU that is located at the SegmentPosition is copied to the same position in the destination I-PDU.		full

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart/IpduMCopyBitField	
BSW Parameter		**BSW Type**
IpduMDestinationBit		IntegerParamDef
BSW Description		
Bit position in an I-PDU of the start of the destination bit field for the copy. The resulting destination field must fit inside the I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	Segments bit position relatively to the beginning of a multiplexed IPdu.	

M2 Parameter	
Fibex::FibexCore::CoreCommunication::SegmentPosition.segmentPosition	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart	
BSW Parameter		**BSW Type**
IpduMJitUpdate		BooleanParamDef
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**
IpduMTxStaticHandleId		IntegerParamDef
BSW Description		
This is an incoming handle id. When the handle of an incoming Tx Request matches this, the bits fields (see IpduMCopyBitField) 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/IpduMTxStaticPart	
BSW Parameter		**BSW Type**
IpduMTxStaticPduRef		ReferenceParamDef
BSW Description		
Reference to the Pdu representation in the ECU Configuration Description exchange file to be transmitted.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to a Com IPdu which is routed to the IPduM module and is combined to a multiplexedPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::StaticPart.iPdu		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest

BSW Parameter		BSW Type
IpduMTxTriggerMode		EnumerationParamDef
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		ParamConfContainerDef
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		FloatParamDef
BSW Description		
The period between successive ticks of AUTOSAR COM 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		BooleanParamDef
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		BooleanParamDef
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

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMGeneral	
BSW Parameter		**BSW Type**
IpduMVersionInfoApi		BooleanParamDef
BSW Description		
Active/Deactivate the version information API. true: version information activated false: version information deactivated		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

9.4 FlexRay Interface Mapping

BSW Module	**BSW Context**		
FrIf	FrIf		
BSW Parameter		**BSW Type**	
FrIfConfig		ParamConfContainerDef	
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**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig		
BSW Parameter		**BSW Type**	
FrIfCluster		ParamConfContainerDef	
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**		
SystemTemplate	FlexRay specific attributes to the physicalCluster		
M2 Parameter			
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster			
Mapping Rule			**Mapping Type**
Create this container for every FlexrayCluster that is exisiting in the system description.			full

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
FrIfClstIdx		IntegerParamDef	
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**
FrIfController		ParamConfContainerDef
BSW Description		

This container contains the configuration of FlexRay CC.		
M2 Template	**M2 Description**	
SystemTemplate	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::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController	
BSW Parameter		**BSW Type**
FrIfAbsTimer		ParamConfContainerDef
BSW Description		
This container contains the configuration of an absolute timer of a FlexRay CC.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfAbsTimer	
BSW Parameter		**BSW Type**
FrIfAbsTimerIdx		IntegerParamDef
BSW Description		
This parameter provides a zero-based consecutive index of the absolute timers. Upper layer BSW modules use this index to identify an absolute timer.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfAbsTimer	
BSW Parameter		**BSW Type**
FrIfFrAbsTimerRef		SymbolicNameReferenceParamDef
BSW Description		
Reference to an absolute timer, which is handled by a specific FlexRay Driver. This reference is unique for the ECU, therefore, an explicit reference to the FlexRay Driver is not necessary.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController	
BSW Parameter		**BSW Type**
FrIfCtrlIdx		IntegerParamDef
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		SymbolicNameReferenceParamDef
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		ParamConfContainerDef
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**	
SystemTemplate	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		
Fibex::Fibex4Flexray::FlexrayCommunication::FlexrayFrameTriggering		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfAllowDynamicLSduLength		BooleanParamDef
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfAlwaysTransmit		BooleanParamDef
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		IntegerParamDef
BSW Description		
This parameter contains the FlexRay Base Cycle used to transmit this FlexRay Frame.		
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		
Fibex::Fibex4Flexray::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		EnumerationParamDef
BSW Description		
This parameter contains the FlexRay Channel used to transmit this FlexRay Frame.		
M2 Template	**M2 Description**	
SystemTemplate	Name of the channel (Channel A or Channel B).	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayPhysicalChannel.channelName		
Mapping Rule		**Mapping Type**

The FrameTriggering is aggregated by the PhysicalChannel.Derive this parameter from the aggregation and the channelName attribute.	full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfCycleRepetition		IntegerParamDef
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**	
SystemTemplate	The number of communication cycles (after the first cycle) whenever the frame described by this timing is sent again.	
M2 Parameter		
Fibex::Fibex4Flexray::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		ReferenceParamDef
BSW Description		
Reference to the Construction Plan of the FlexRay Frame.		
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::FibexCore::CoreCommunication::FrameTriggering.frame		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfLSduLength		IntegerParamDef
BSW Description		
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].		
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**
In the system description the frameLength is specified in the Frame element that is referenced by the FrameTriggering.		full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering
BSW Parameter	**BSW Type**
FrIfPayloadPreamble	BooleanParamDef
BSW Description	
Switching the Payload Preamble bit.	
M2 Template	**M2 Description**
SystemTemplate	Switching the Payload Preamble bit.
M2 Parameter	
Fibex::Fibex4Flexray::FlexrayCommunication::FlexrayFrameTriggering.payloadPreambleIndicator	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering
BSW Parameter	**BSW Type**
FrIfSlotId	IntegerParamDef
BSW Description	
This parameter contains the FlexRay Slot ID used to transmit this FlexRay Frame.	
M2 Template	**M2 Description**
SystemTemplate	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.
M2 Parameter	
Fibex::Fibex4Flexray::FlexrayCommunication::AbsolutelyScheduledTiming.slotID	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController
BSW Parameter	**BSW Type**
FrIfLPdu	ParamConfContainerDef
BSW Description	
LPdu is an abstraction of all FlexRay Frames (L-PDUs) belonging to the same Frame Triggering FrIfFrameTriggering.	
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**
FrIfLPduIdx	IntegerParamDef
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**
FrIfVBTriggeringRef	ReferenceParamDef
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**
FrIfRelTimer	ParamConfContainerDef
BSW Description	
This container contains the configuration of a relative timer of a FlexRay CC.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfRelTimer
BSW Parameter	**BSW Type**
FrIfFrRelTimerRef	SymbolicNameReferenceParamDef
BSW Description	
Reference to a relative timer, which is handled by a specific FlexRay Driver. This reference is unique for the ECU, therefore, an explicit reference to the FlexRay Driver is not necessary.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfRelTimer
BSW Parameter	**BSW Type**

FrIfRelTimerIdx		IntegerParamDef	
BSW Description			
This parameter provides a zero-based consecutive index of the relative timers. Upper layer BSW modules use this index to identify a relative timer.			
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		ParamConfContainerDef	
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**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfTransceiver		
BSW Parameter		**BSW Type**	
FrIfClusterChannel		EnumerationParamDef	
BSW Description			
This parameter identifies to which one of the two Channels "A" or "B" of the Cluster the Transceiver is connected.			
M2 Template	**M2 Description**		
SystemTemplate	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.		
M2 Parameter			
Fibex::FibexCore::CoreTopology::CommunicationCluster.physicalChannel			
Mapping Rule			**Mapping Type**
The enumeration shall be derived from the aggregated physicalChannels.			full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfTransceiver	
BSW Parameter		**BSW Type**
FrIfFrTrcvChannelRef		SymbolicNameReferenceParamDef
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**
FrIfGAssumedPreciscion		IntegerParamDef
BSW Description		
This parameter is deprecated and will be removed in future. Old description: Assumed precision of the application network.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGChannels		EnumerationParamDef
BSW Description		
The channels that are used by the cluster. Implementation Type: Fr_ChannelType		
M2 Template	**M2 Description**	
SystemTemplate	Name of the channel (Channel A or Channel B).	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayPhysicalChannel.channelName		
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**
FrIfGClusterDriftDamping		IntegerParamDef
BSW Description		
The cluster drift damping factor, based on the longest microtick gdMaxMicrotick used in the cluster. Used to compute the local cluster drift damping factor pClusterDriftDamping [Micrroticks].		
M2 Template	**M2 Description**	
SystemTemplate	The cluster drift damping factor used in clock synchronization rate correction in microticks	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.clusterDriftDamping		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
FrIfGColdStartAttempts		IntegerParamDef	
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**		
SystemTemplate	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			
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.coldStartAttempts			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
FrIfGCycleCountMax		IntegerParamDef	
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**		
SystemTemplate	Maximum cycle counter value in a given cluster. Remark: Set to 63 for FlexRay Protocol 2.1 Rev. A compliance.		
M2 Parameter			
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.cycleCountMax			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
FrIfGListenNoise		IntegerParamDef	
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 cluster parameter pdListenTimeout.			
M2 Template	**M2 Description**		
SystemTemplate	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			
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.listenNoise			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGMacroPerCycle		IntegerParamDef
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**	
SystemTemplate	The number of macroticks in a communication cycle	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.macroPerCycle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
FrIfGMaxWithoutClockCorrectFatal		IntegerParamDef	
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**		
SystemTemplate	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			
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.maxWithoutClockCorrectionFatal			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
FrIfGMaxWithoutClockCorrectPassive		IntegerParamDef	
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**		
SystemTemplate	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			
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.maxWithoutClockCorrectionPassive			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGNetworkManagementVectLength		IntegerParamDef
BSW Description		

Length of the Network Management vector in a cluster [bytes]		
M2 Template	**M2 Description**	
SystemTemplate	Length of the Network Management vector on a cluster. Unit: Bytes	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.networkManagementVectorLength		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGNumberOfMinislots		IntegerParamDef
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**	
SystemTemplate	Number of minislots in the dynamic segment	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.numberOfMinislots		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGNumberOfStaticSlots		IntegerParamDef
BSW Description		
Number of static slots in the static segment		
M2 Template	**M2 Description**	
SystemTemplate	The number of static slots in the static segment.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.numberOfStaticSlots		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGOffsetCorrectionMax		FloatParamDef
BSW Description		
describes the maximum value which the offset correction should assume in seconds.		
M2 Template	**M2 Description**	
SystemTemplate	Cluster global magnitude of the maximum permissible offset correction value Unit: seconds (gOffsetCorrectionMax)	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.offsetCorrectionMax		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGPayloadLengthStatic		IntegerParamDef
BSW Description		
Payload length of a static frame [16 bit words]		
M2 Template	**M2 Description**	
SystemTemplate	Globally configured payload length of a static frame. Unit: 16-bit WORDS.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.payloadLengthStatic		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGSyncFrameIDCountMax		IntegerParamDef
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**	
SystemTemplate	Maximum number of distinct syncframe identifiers present in a given cluster.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.syncFrameIDCountMax		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdActionPointOffset		IntegerParamDef
BSW Description		
Number of Macroticks the action point is offset from the beginning of a Static Slots or symbol window.		
M2 Template	**M2 Description**	
SystemTemplate	The offset of the action point in networks	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.actionPointOffset		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdBit		EnumerationParamDef
BSW Description		
Nominal bit time in seconds		
M2 Template	**M2 Description**	
SystemTemplate	Nominal bit time (= 1 / fx:SPEED). gdBit = cSamplesPerBit * gdSampleClockPeriod. Unit: seconds (gdBit)	
M2 Parameter		

Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.bit	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdBitMax		FloatParamDef
BSW Description		
Maximum bit time taking into account the allowable clock deviation of each node (in seconds).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdBitMin		FloatParamDef
BSW Description		
Minimum bit time taking into account the allowable clock deviation of each node (in seconds).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdCasRxLowMax		IntegerParamDef
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**	
SystemTemplate	Upper limit of the Collision Avoidance Symbol (CAS) acceptance window.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.casRxLowMax		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdCycle		FloatParamDef

BSW Description	
Length of the cycle, expressed in [s] Remark: Lower limit 0.000024 for FlexRay Protocol 3.0 compliance.	
M2 Template	**M2 Description**
SystemTemplate	Length of the cycle.
M2 Parameter	
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.cycle	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdDynamicSlotIdlePhase		IntegerParamDef
BSW Description		
Duration of the idle phase within a dynamic slot [Minislots].		
M2 Template	**M2 Description**	
SystemTemplate	The duration of the dynamic slot idle phase in minislots.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.dynamicSlotIdlePhase		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdIgnoreAfterTx		IntegerParamDef
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**	
SystemTemplate	Duration for which the bitstrobing is paused after transmission [gdBit].	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.ignoreAfterTx		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdMacrotick		FloatParamDef
BSW Description		
Duration of the cluster wide nominal macrotick, expressed in s		
M2 Template	**M2 Description**	
SystemTemplate	Duration of the cluster wide nominal macrotick, expressed in seconds	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.macrotickDuration		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdMaxInitializationError		FloatParamDef
BSW Description		
Maximum error that a node may have following integration in seconds.		
M2 Template	**M2 Description**	
SystemTemplate	The maximum error that a node may have after initialization. Unit: seconds	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.maxInitialisationError		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdMaxMicrotick		EnumerationParamDef
BSW Description		
Maximum Microtick length of all Microticks configured within a Cluster.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdMaxPropagationDelay		FloatParamDef
BSW Description		
Maximum propagation delay of a Cluster (in seconds).		
M2 Template	**M2 Description**	
SystemTemplate	Maximum propagation delay of a Cluster (in seconds).	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.maxPropagationDelay		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdMinPropagationDelay		FloatParamDef
BSW Description		
Minimum propagation delay of a Cluster (in seconds).		
M2 Template	**M2 Description**	
SystemTemplate	Minimum propagation delay of a Cluster (in seconds).	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.minPropagationDelay		
Mapping Rule		**Mapping Type**

1:1 mapping	full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdMiniSlotActionPointOffset		IntegerParamDef
BSW Description		
Number of Macroticks the Minislot action point is offset from the beginning of a Minislot [Macroticks].		
M2 Template	**M2 Description**	
SystemTemplate	The Offset of the action point within a minislot. Unit: macroticks	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.minislotActionPointOffset		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdMinislot		IntegerParamDef
BSW Description		
Duration of a minislot [Macroticks]		
M2 Template	**M2 Description**	
SystemTemplate	The duration of a minislot (dynamic segment). Unit: macroticks.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.minislotDuration		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdNit		IntegerParamDef
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**	
SystemTemplate	The duration of the network idle time in macroticks	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.networkIdleTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdSampleClockPeriod		EnumerationParamDef
BSW Description		
Sample clock period		

M2 Template	**M2 Description**
SystemTemplate	Sample clock period. Unit: seconds
M2 Parameter	
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.sampleClockPeriod	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster
BSW Parameter	**BSW Type**
FrIfGdStaticSlot	IntegerParamDef
BSW Description	
Duration of a Static Slot [Macroticks]. Remark: Range 4 to 661 for FlexRay Protocol 2.1 Rev. A compliance	
M2 Template	**M2 Description**
SystemTemplate	The duration of a slot in the static segment. Unit: macroticks
M2 Parameter	
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.staticSlotDuration	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster
BSW Parameter	**BSW Type**
FrIfGdSymbolWindow	IntegerParamDef
BSW Description	
Duration of the symbol window [Macroticks]. Remark: Range 0-142 for FlexRay Protocol 2.1 Rev. A compliance.	
M2 Template	**M2 Description**
SystemTemplate	The duration of the symbol window. Unit: macroticks
M2 Parameter	
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.symbolWindow	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster
BSW Parameter	**BSW Type**
FrIfGdSymbolWindowActionPointOffset	IntegerParamDef
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**
SystemTemplate	Number of macroticks the action point offset is from the beginning of the symbol window [Macroticks].
M2 Parameter	
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.symbolWindowActionPointOffset	
Mapping Rule	**Mapping Type**
1:1 mapping	full

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="2">FrIf/FrIfConfig/FrIfCluster</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIfGdTssTransmitter</td><td>IntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Number of bits in the Transmission Start Sequence [gdBits].
Remark: Lower limit 3 for FlexRay Protocol 2.1 Rev. A compliance.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">Number of bits in the Transmission Start Sequence [gdBits].</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.transmissionStartSequenceDuration</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</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIfGdWakeupRxIdle</td><td>IntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">Number of bits used by the node to test the duration of the 'idle' or HIGH phase of a received wakeup. Unit:bitDuration</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.wakeupRxIdle</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</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIfGdWakeupRxLow</td><td>IntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">Number of bits used by the node to test the duration of the LOW phase of a received wakeup. Unit:bitDuration</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.wakeupRxLow</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</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIfGdWakeupRxWindow</td><td>IntegerParamDef</td></tr>
</table>

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**	
SystemTemplate	The size of the window used to detect wakeups [gdBit].	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.wakeupRxWindow		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdWakeupTxActive		IntegerParamDef
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**	
SystemTemplate	Number of bits used by the node to transmit the LOW phase of awakeup symbol and the HIGH and LOW phases of a WUDOP.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.wakeupTxActive		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdWakeupTxIdle		IntegerParamDef
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**	
SystemTemplate	Number of bits used by the node to transmit the 'idle' part of a wakeup symbol. Unit: gDbit.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.wakeupTxIdle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfJobList		ParamConfContainerDef
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		SymbolicNameReferenceParamDef
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		ParamConfContainerDef
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		ParamConfContainerDef
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

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList/FrIfJob/FrIfCommunicationOperation	
BSW Parameter		**BSW Type**
FrIfCommunicationAction		EnumerationParamDef
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		IntegerParamDef
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		ReferenceParamDef
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	
BSW Parameter		**BSW Type**
FrIfCycle		IntegerParamDef
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		IntegerParamDef
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/FrIfJobList/FrIfJob	
BSW Parameter		**BSW Type**
FrIfMaxIsrDelay		IntegerParamDef
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**
FrIfMainFunctionPeriod		FloatParamDef
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	
BSW Parameter		**BSW Type**

FrIfFrameStructure		ParamConfContainerDef
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**	
SystemTemplate	Data frame which is sent over a communication medium. This element describes the pure Layout of a frame sent on a channel.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::Frame		
Mapping Rule		**Mapping Type**
Create container for each Frame that is transmitted or received by the regarded ECU via FlexRay.		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfFrameStructure	
BSW Parameter		**BSW Type**
FrIfByteOrder		EnumerationParamDef
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**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::PduToFrameMapping.packingByteOrder		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfFrameStructure	
BSW Parameter		**BSW Type**
FrIfPdusInFrame		ParamConfContainerDef
BSW Description		
This container holds all the information about a PDU in a FlexRay Frame.		
M2 Template	**M2 Description**	
SystemTemplate	A PduToFrameMapping defines the composition of Pdus in each frame.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::PduToFrameMapping		
Mapping Rule		**Mapping Type**
Container shall be created for each IPduToFrameMapping element that is aggregated by the frame.		full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfFrameStructure/FrIfPdusInFrame

BSW Parameter			BSW Type
FrFrIfPduRef			ReferenceParamDef
BSW Description			
This is the reference to the local definition of a PDU.			
M2 Template	**M2 Description**		
SystemTemplate	A PduToFrameMapping defines the composition of Pdus in each frame.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::PduToFrameMapping.pdu			
Mapping Rule		**Mapping Type**	
This reference shall be derived from the PduToFrameMapping element. The PduToFrameMapping element contains a reference to the pdu that is mapped to the Frame.		full	

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfFrameStructure/FrIfPdusInFrame		
BSW Parameter			**BSW Type**
FrIfPduOffset			IntegerParamDef
BSW Description			
The value specifies the offset of the PDU within the Frame [bytes].			
M2 Template	**M2 Description**		
SystemTemplate	This attribute describes the bitposition of a Pdu within a Frame. The Pdus are byte aligned in a Frame and only the values 0, 8, 16, 24,... (for little endian) and 7, 15, 23, ... (for big endian) are allowed.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::PduToFrameMapping.startPosition			
Mapping Rule		**Mapping Type**	
Please note that the startPosition attribute is defined in bits and the FrIfPdu Offset parameter is defined in bytes.		full	

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfFrameStructure/FrIfPdusInFrame		
BSW Parameter			**BSW Type**
FrIfPduUpdateBitOffset			IntegerParamDef
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**		
SystemTemplate	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			
Fibex::FibexCore::CoreCommunication::PduToFrameMapping.updateIndicationBitPosition			
Mapping Rule		**Mapping Type**	
1:1 mapping		full	

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig	
BSW Parameter		**BSW Type**
FrIfPdu		ParamConfContainerDef
BSW Description		

Contains PDU information. A PDU may be either a transmission PDU or a reception PDU.		
M2 Template	**M2 Description**	
	Reference to a I-Pdu, N-Pdu or NmPdu that is transmitted in the Frame.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::PduToFrameMapping.pdu		
Mapping Rule		**Mapping Type**
The container must be created for each Pdu that is contained in a FlexRay Frame and is transmitted or received by the regarded ECU.		

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfPdu	
BSW Parameter		**BSW Type**
FrIfPduDirection		ChoiceContainerDef
BSW Description		
A PDU is either transmit or receive		
M2 Template	**M2 Description**	
SystemTemplate	Communication Direction of the Connector Port (input or output Port).	
M2 Parameter		
Fibex::FibexCore::CoreTopology::CommConnectorPort.communicationDirection		
Mapping Rule		**Mapping Type**
The direction of the Pdu can be derived from the IPduTriggering that contains a reference to the Pdu and to the IPduPort.		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection	
BSW Parameter		**BSW Type**
FrIfRxPdu		ParamConfContainerDef
BSW Description		
Receive PDU		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfRxPdu	
BSW Parameter		**BSW Type**
FrIfPduRef		ReferenceParamDef
BSW Description		
Reference to the external PDU definition.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfRxPdu	
BSW Parameter		**BSW Type**
FrIfRxIndicationName		FunctionNameDef
BSW Description		
This parameter defines the name of the <User_RxIndication>. This parameter depends on the parameter FRIF_USERRXINDICATION_UL. If FRIF_USERRXINDICATION_UL equals FR_TP, FR_NM, or PDUR the name of the <User_RxIndication> is fixed. If FRIF_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	
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfRxPdu	
BSW Parameter		**BSW Type**
FrIfUserRxIndicationUL		EnumerationParamDef
BSW Description		
This parameter defines the upper layer (UL) module to which the indication of the successfully received FRIFRXPDU has to be routed via <User_RxIndication>. This <User_RxIndication> has to be invoked when the indication of the configured FRIFRXPDU will be received by a Rx indication event from the FR Driver module. If no upper layer (UL) module is configured, no <User_RxIndication> has to be called in case of a Rx indication event of the FRIFRXPDU from the FR Driver module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection	
BSW Parameter		**BSW Type**
FrIfTxPdu		ParamConfContainerDef
BSW Description		
This container specifies transmission PDUs.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfTxPdu	
BSW Parameter		**BSW Type**

FrIfConfirm	BooleanParamDef
BSW Description	
Defines whether the transmission of a PDU should be checked and confirmed to the PDU owning BSW module.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfTxPdu
BSW Parameter	**BSW Type**
FrIfCounterLimit	IntegerParamDef
BSW Description	
This value states the maximum number of indication of ready PDU data to the FrIf (i.e. maximum number of invocations of FrIf_Transmit) without an intermediate transmission of the PDU.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfTxPdu
BSW Parameter	**BSW Type**
FrIfImmediate	BooleanParamDef
BSW Description	
Defines whether the the PDU is transmitted immediate or decoupled.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfTxPdu
BSW Parameter	**BSW Type**
FrIfNoneMode	BooleanParamDef
BSW Description	
Using the "None-Mode" which means that there is no API FrIf_Transmit call of the upper layer for this PDU.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

	local

BSW Module	BSW Context
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfTxPdu
BSW Parameter	**BSW Type**
FrIfPduRef	ReferenceParamDef
BSW Description	
Reference to the external PDU definition.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfTxPdu
BSW Parameter	**BSW Type**
FrIfTxConfirmationName	FunctionNameDef
BSW Description	
This parameter defines the name of the <User_TxConfirmation>. This parameter depends on the parameter FrIfUserTxUL. If FrIfUserTxUL equals FR_TP, FR_NM, PDUR or XCP, the name of the <User_TxConfirmation> is fixed. If FrIfUserTxUL 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
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfTxPdu
BSW Parameter	**BSW Type**
FrIfTxPduId	IntegerParamDef
BSW Description	
The global PDU identifier, which has to be used by the upper layer BSW module. The identifier has to be zero based and consecutive.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfTxPdu
BSW Parameter	**BSW Type**
FrIfUserTriggerTransmitName	FunctionNameDef

BSW Description		
This parameter defines the name of the <User_TriggerTransmit>. This parameter depends on the parameter FrIfUserTxUL. If FrIfUserTxUL equals FR_TP, FR_NM, PDUR or XCP, the name of the <User_TriggerTransmit> is fixed. If FrIfUserTxUL equals CDD, the name of the <User_TriggerTransmit> is selectable.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfTxPdu	
BSW Parameter		**BSW Type**
FrIfUserTxUL		EnumerationParamDef
BSW Description		
This parameter defines the upper layer (UL) module to which the trigger of the to be transmitted FRIFTXPDUID (via the <User_TriggerTransmit>) or the confirmation of the successfully transmitted FRIFTXPDUID has to be routed (via the <User_TxConfirmation>).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf	
BSW Parameter		**BSW Type**
FrIfGeneral		ParamConfContainerDef
BSW Description		
This container contains the general configuration parameters of the FlexRay Interface.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfCCReadbackApiSupport		BooleanParamDef
BSW Description		
Enables/disbales the "FrIf_ReadCCConfig" API (FrIf05313). This should not be mixed up with CCReadbackSupport functionality of FlexRay Driver during FrControllerInit (FR647).		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfDevErrorDetect		BooleanParamDef
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**
FrIfFreeOpAApiName		StringParamDef
BSW Description		
API name that is called when FREE_OP_A is selected as communication operation. See also chapter 8.8.3 Configurable Interfaces.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfFreeOpBApiName		StringParamDef
BSW Description		
API name that is called when FREE_OP_B is selected as communication operation. See also chapter 8.8.3 Configurable Interfaces.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**

FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfFreeOpsHeader		StringParamDef
BSW Description		
Defines header file for configurable FREE_OP_A / FREE_OP_B functions.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfGetClockCorrectionSupport		BooleanParamDef
BSW Description		
Enables/disables the "FrIf_GetClockCorrection" API (FrIf05071).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfGetGetChannelStatusSupport		BooleanParamDef
BSW Description		
Enables/disables the "FrIf_GetChannelStatus" API (FrIf05030).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfGetNmVectorSupport		BooleanParamDef
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**
FrIfNumClstSupported		IntegerParamDef
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		IntegerParamDef
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		StringParamDef
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**
FrIfUnusedBitValue		IntegerParamDef
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		BooleanParamDef
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

9.5 FlexRay Driver Mapping

BSW Module	BSW Context	
Fr	Fr	
BSW Parameter		**BSW Type**
FrGeneral		ParamConfContainerDef
BSW Description		
General configuration (parameters) of the FlexRay Driver module.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreTopology::EcuInstance		
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**
FrCCReadbackSupport		BooleanParamDef
BSW Description		
Enables/disbales the "read back and compare" functionality (FR649, FR647, FR598) during the execution of Fr_ControllerInit.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Fr	Fr/FrGeneral	
BSW Parameter		**BSW Type**
FrDevErrorDetect		BooleanParamDef
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		IntegerParamDef	
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		IntegerParamDef	
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**	
FrRelativeTimerEnable		BooleanParamDef	
BSW Description			
Enables or disables the usage of relative timers. Pre-compile time switch FR_RELATIVE_TIMER_ENABLE is derived from this configuration parameter.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
Fr	Fr/FrGeneral	
BSW Parameter		**BSW Type**
FrVersionInfoApi		BooleanParamDef
BSW Description		
Enables/disables the existence of the Fr_GetVersionInfo API. Pre-compile time switch FR_VERSION_INFO_API is derived from this configuration parameter.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
Fr	Fr	
BSW Parameter		**BSW Type**
FrMultipleConfiguration		ParamConfContainerDef
BSW Description		
Configuration of the individual controllers.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration	
BSW Parameter		**BSW Type**
FrController		ParamConfContainerDef
BSW Description		
Configuration of the individual controller.		
M2 Template	**M2 Description**	
SystemTemplate	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::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController		
Mapping Rule		**Mapping Type**
Container must be created if the ECU in the system description contains a Flex Ray communication controller that is connected to the regarded communication cluster.		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrAbsoluteTimer		ParamConfContainerDef
BSW Description		
Specifies the absolute timer configuration parameters of the Fr.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrAbsoluteTimer	
BSW Parameter		**BSW Type**

FrAbsTimerIdx		IntegerParamDef
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**
FrCtrlClock		IntegerParamDef
BSW Description		
Determines clock connected to the CC [Hz].		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrCtrlIdx		IntegerParamDef
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		ParamConfContainerDef
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::FlexrayTopology::FlexrayFifoConfiguration		
Mapping Rule		**Mapping Type**

This container shall be created for each FlexrayFifoConfiguration that is described in the system description in the regarded FlexrayCommunicationController.	full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo	
BSW Parameter		**BSW Type**
FrAdmitWithoutMessageId		BooleanParamDef
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::FlexrayTopology::FlexrayFifoConfiguration.admitWithoutMessageId		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo	
BSW Parameter		**BSW Type**
FrBaseCycle		IntegerParamDef
BSW Description		
FIFO cycle counter acceptance criteria.		
M2 Template	**M2 Description**	
SystemTemplate	FIFO cycle counter acceptance criteria.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayFifoConfiguration.baseCycle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo	
BSW Parameter		**BSW Type**
FrChannels		EnumerationParamDef
BSW Description		
FIFO channel admittance criteria.		
M2 Template	**M2 Description**	
SystemTemplate	Fifo channel admittance criteria.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayFifoConfiguration.channel		
Mapping Rule		**Mapping Type**
If channel A is referenced by the FlexrayFifoConfiguration class use FR_CHANNEL_A literal. If channel B is referenced use FR_CHANNEL_B literal. If channel A and channelB are referenced use FR_CHANNEL_AB literal.		full

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo

BSW Parameter		BSW Type
FrCycleRepetition		IntegerParamDef
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::FlexrayTopology::FlexrayFifoConfiguration.cycleRepetition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo	
BSW Parameter		**BSW Type**
FrFifoDepth		IntegerParamDef
BSW Description		
Fifo Depth.		
M2 Template	**M2 Description**	
SystemTemplate	Fifo Depth.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayFifoConfiguration.fifoDepth		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo	
BSW Parameter		**BSW Type**
FrMsgIdMask		IntegerParamDef
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::FlexrayTopology::FlexrayFifoConfiguration.msgIdMask		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo	
BSW Parameter		**BSW Type**
FrMsgIdMatch		IntegerParamDef
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::FlexrayTopology::FlexrayFifoConfiguration.msgIdMatch		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo	
BSW Parameter		**BSW Type**
FrRange		ParamConfContainerDef
BSW Description		
FIFO Frame Id range acceptance criteria.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo/FrRange	
BSW Parameter		**BSW Type**
FrRangeMax		IntegerParamDef
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::FlexrayTopology::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		IntegerParamDef
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::FlexrayTopology::FlexrayFifoRange.rangeMin		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPAllowHaltDueToClock		BooleanParamDef
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**	

SystemTemplate	Boolean flag that controls the transition to the POC:halt state due to a clock synchronization errors.	
M2 Parameter		
Fibex::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		IntegerParamDef
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**	
SystemTemplate	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		
Fibex::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		EnumerationParamDef
BSW Description		
Channels to which the node is connected Implementation Type: Fr_ChannelType		
M2 Template	**M2 Description**	
SystemTemplate	The CommunicationConnector creates the connection between the ECU and a PhysicalChannel.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexRayCommunicationConnector		
Mapping Rule		**Mapping Type**

If the Controller is referenced by a connector that contain references to:
channelA => use Channel_A literal;
channelB => use Channel_B literal;
		full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPClusterDriftDamping		IntegerParamDef
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**	

SystemTemplate	The cluster drift damping factor used in clock synchronization rate correction in microticks	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.clusterDriftDamping		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPDecodingCorrection		IntegerParamDef
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**	
SystemTemplate	Value used by the receiver to calculate the difference between primary time reference point and secondary time reference point. Unit: Microticks (pDecodingCorrection)	
M2 Parameter		
Fibex::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		IntegerParamDef
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 0.0125 μs to 0.05 μs. In practice, the minimum of the propagation delays of all sync nodes should be applied [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**	
SystemTemplate	Value used by the receiver to calculate the difference between primary time reference point and secondary time reference point. Unit: Microticks (pDecodingCorrection)	
M2 Parameter		
Fibex::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		IntegerParamDef
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 0.0125 μs to 0.05 μs. In practice, the minimum of the propagation delays of all sync nodes should be applied [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**	
SystemTemplate	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.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.delayCompensationB		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPExternOffsetCorrection		IntegerParamDef
BSW Description		
Number of microticks added or subtracted to the NIT to carry out a host-requested external offset correction [Microticks]. Remark: Upper limit 7 for FlexRay Protocol 2.1 Rev A compliance.		
M2 Template	**M2 Description**	
SystemTemplate	Fixed amount added or subtracted to the calculated offset correction term to facilitate external offset correction, expressed in node-local microticks.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.externOffsetCorrection		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPExternRateCorrection		IntegerParamDef
BSW Description		
Number of microticks added or subtracted to the cycle to carry out a host-requested external rate correction [Microticks]. Remark: Upper limit 7 for FlexRay Protocol 2.1 Rev A compliance.		
M2 Template	**M2 Description**	
SystemTemplate	Fixed amount added or subtracted to the calculated rate correction term to facilitate external rate correction, expressed in node-local microticks.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.externRateCorrection		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPExternalSync		BooleanParamDef
BSW Description		

<table>
<tr><td colspan="3">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.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">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.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.externalSync</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>Fr</td><td colspan="2">Fr/FrMultipleConfiguration/FrController</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrPFallBackInternal</td><td>BooleanParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">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).</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.fallBackInternal</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>Fr</td><td colspan="2">Fr/FrMultipleConfiguration/FrController</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrPKeySlotId</td><td>IntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.

For Fr3.0: if the value is not provided in System Template it shall be configured to 0.
For Fr2.1: if the value is not provided in System Template it is driver implementation specific which value to configure.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">ID of the slot used to transmit the startup frame, sync frame, or designated single slot frame.
For Fr3.0: if the value is not provided in SysT it shall be configured to 0.
For Fr2.1: if the value is not provided in SysT it is driver impl. specific.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.keySlotID</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>

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPKeySlotOnlyEnabled		BooleanParamDef
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**	
SystemTemplate	Flag indicating whether or not the node shall enter key slot only mode following startup.	
M2 Parameter		
Fibex::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		BooleanParamDef
BSW Description		
Flag indicating whether the Key Slot is used to transmit a startup frame		
M2 Template	**M2 Description**	
SystemTemplate	Flag indicating whether the Key Slot is used to transmit a startup frame.	
M2 Parameter		
Fibex::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		BooleanParamDef
BSW Description		
Flag indicating whether the Key Slot is used to transmit a sync frame		
M2 Template	**M2 Description**	
SystemTemplate	Flag indicating whether the Key Slot is used to transmit a sync frame.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.keySlotUsedForSync		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPLatestTx		IntegerParamDef
BSW Description		
Number of the last minislot in which a frame transmission can start in the dynamic segment [Minislots]. Remark: Upper limit 7980 for FlexRay Protocol 2.1 Rev A compliance.		
M2 Template	**M2 Description**	

SystemTemplate	Number of the last minislot in which a frame transmission can start in the dynamic segment [Minislots]. Remark: Upper limit 7980 for FlexRay Protocol 2.1 Rev A compliance.	
M2 Parameter		
Fibex::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		IntegerParamDef
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**	
SystemTemplate	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		
Fibex::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		IntegerParamDef
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**	
SystemTemplate	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 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.macroInitialOffsetB		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPMicroInitialOffsetA		IntegerParamDef
BSW Description		
Number of microticks between the closest macrotick boundary described by pMacroInitialOffset[Ch] and the secondary time reference point. The parameter depends on pDelayCompensation[Ch] and therefore it has to be set independently for each channel [Microticks].		
M2 Template	**M2 Description**	
SystemTemplate	Number of microticks between the closest macrotick boundary described by pMacroInitialOffset[Ch] and the secondary time reference point.	

M2 Parameter	
Fibex::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		IntegerParamDef
BSW Description		
Number of microticks between the closest macrotick boundary described by pMacroInitialOffset[Ch] and the secondary time reference point. The parameter depends on pDelayCompensation[Ch] and therefore it has to be set independently for each channel [Microticks].		
M2 Template	**M2 Description**	
SystemTemplate	Number of microticks between the closest macrotick boundary described by gMacroInitialOffset and the secondary time reference point.	
M2 Parameter		
Fibex::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		IntegerParamDef
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**	
SystemTemplate	The nominal number of microticks in a communication cycle	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.microPerCycle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPMicroPerMacroNom		IntegerParamDef
BSW Description		
This parameter is deprecated and will be removed in future. Old description: Number of microticks per nominal macrotick that all implementations must support [Microticks].		
M2 Template	**M2 Description**	
SystemTemplate	Number of microticks per nominal macrotick that all implementations must support.	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.microPerMacroNom		

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPNmVectorEarlyUpdate		BooleanParamDef
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPOffsetCorrectionOut		IntegerParamDef
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**	
SystemTemplate	Magnitude of the maximum permissible offset correction value. Unit:microtick (pOffsetCorrectionOut)	
M2 Parameter		
Fibex::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		IntegerParamDef
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 Network Idle Time (NIT), expressed as the number of macroticks from the start of cycle. Unit: macroticks	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCluster.OffsetCorrectionStart		
Mapping Rule		**Mapping Type**

1:1 mapping	full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPPayloadLengthDynMax		IntegerParamDef
BSW Description		
Maximum payload length for dynamic frames [16 bit words].		
M2 Template	**M2 Description**	
SystemTemplate	Maximum payload length for the dynamic channel of a frame in 16 bit WORDS.	
M2 Parameter		
Fibex::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		IntegerParamDef
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**	
SystemTemplate	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		
Fibex::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		EnumerationParamDef
BSW Description		
Number of samples per microtick. Remark: Allowed range N1SAMPLES, N2SAMPLES for FlexRay Protocol 3.0 compliance.		
M2 Template	**M2 Description**	
SystemTemplate	Number of samples per microtick	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.samplesPerMicrotick		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context

Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
FrPSecondKeySlotId		IntegerParamDef	
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			
Fibex::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		BooleanParamDef	
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			
Fibex::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		EnumerationParamDef	
BSW Description			
Channel used by the node to send a wakeup pattern.			
M2 Template	**M2 Description**		
SystemTemplate	Referenced channel used by the node to send a wakeup pattern. (pWakeupChannel)		
M2 Parameter			
Fibex::Fibex4Flexray::FlexrayTopology::FlexRayCommunicationConnector.wakeUpChannel			
Mapping Rule			**Mapping Type**
If the FlexRayCommunicationConnector that contains a reference to the regarded CommunicationController is marked as a wakeUpChannel than the referenced PhysicalChannel (ChannelA, ChannelB) shall be used as input for this parameter.			full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPWakeupPattern		IntegerParamDef
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**	
SWComponentTemplate	Number of repetitions of the Tx-wakeup symbol to be sent during the CC_WakeupSend state of this Node in the cluster	
M2 Parameter		
Fibex::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		IntegerParamDef
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**	
SystemTemplate	Expanded range of measured clock deviation allowed for startup frames during integration. Unit:microtick	
M2 Parameter		
Fibex::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		IntegerParamDef
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**	
SystemTemplate	Upper limit for the start up listen timeout and wake up listen timeout. Unit: Microticks	
M2 Parameter		
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.listenTimeout		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context

Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrPdMicrotick	EnumerationParamDef
BSW Description	
Duration of a microtick. Remark: Allowed range T12_5NS, T25NS, T50NS for FlexRay Protocol 3.0 compliance.	
M2 Template	**M2 Description**
SystemTemplate	Duration of a microtick. This attribute can be derived from samplePerMicrotick and gdSampleClockPeriod. Unit: seconds
M2 Parameter	
Fibex::Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.microtickDuration	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrRelativeTimer	ParamConfContainerDef
BSW Description	
Specifies the relative timer configuration parameters of the Fr.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController/FrRelativeTimer
BSW Parameter	**BSW Type**
FrRelTimerIdx	IntegerParamDef
BSW Description	
Contains the index of a relative timer contained in Fr on a certain FlexRay CC.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

9.6 FlexRayTP Mapping

BSW Module	BSW Context	
FrTp	FrTp	
BSW Parameter		**BSW Type**
FrTpGeneral		ParamConfContainerDef
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	
FrTp	FrTp/FrTpGeneral	
BSW Parameter		**BSW Type**
FrTpChanNum		IntegerParamDef
BSW Description		
This reference is deprecated and will be removed in a future release. Old 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**
FrTpDevErrorDetect		BooleanParamDef
BSW Description		
Preprocessor switch for enabling development error detection.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrTp	FrTp/FrTpGeneral	
BSW Parameter		**BSW Type**
FrTpHaveAckRt		BooleanParamDef
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**	
FrTp	FrTp/FrTpGeneral	
BSW Parameter		**BSW Type**
FrTpHaveGrpSeg		BooleanParamDef
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**	
FrTp	FrTp/FrTpGeneral	
BSW Parameter		**BSW Type**
FrTpHaveLm		BooleanParamDef
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**	
FrTp	FrTp/FrTpGeneral	
BSW Parameter		**BSW Type**
FrTpHaveTc		BooleanParamDef
BSW Description		
Preprocessor switch for enabling Transmit Cancellation and Receive Cancellation.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**
FrTp	FrTp/FrTpGeneral

BSW Parameter			BSW Type
FrTpMainFuncCycle			FloatParamDef
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		
FrTp	FrTp/FrTpGeneral		
BSW Parameter			**BSW Type**
FrTpVersionInfoApi			BooleanParamDef
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		
FrTp	FrTp		
BSW Parameter			**BSW Type**
FrTpMultipleConfig			ParamConfContainerDef
BSW Description			
This container holds one or several multiple configuration sets.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context		
FrTp	FrTp/FrTpMultipleConfig		
BSW Parameter			**BSW Type**
FrTpChannel			ParamConfContainerDef
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 FlexRay Transport 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-ID		
M2 Parameter			

TransportProtocols::FlexrayTpChannel	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpAckType		EnumerationParamDef
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::FlexrayTpChannel.ackType		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpAdrType		EnumerationParamDef
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::FlexrayTpChannel.extendedAddressing		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpChannelId		IntegerParamDef
BSW Description		
Please note that this parameter is deprecated and will be removed in a future release. Old description: The Id of the channel.		
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>FrTp</td><td colspan="2">FrTp/FrTpMultipleConfig/FrTpChannel</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrTpConNum</td><td>IntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Please note that this parameter is deprecated and will be removed in a future release.

Old description:
This parameter states the number of connections used in this channel. At least 256 shall be configurable here.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">Group of connections that can be used in this channel.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">TransportProtocols::FlexrayTpChannel.tpConnection</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">Count aggregated TpConnections.</td><td>local</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/FrTpChannel</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrTpConcurrentConnections</td><td>IntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This parameter defines the number of connections that can be active at the same time. If set to 0, all configured connections can be active at the same time.</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>FrTp</td><td colspan="2">FrTp/FrTpMultipleConfig/FrTpChannel</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrTpConnection</td><td>ParamConfContainerDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This container contains the configuration (parameters) of one FlexRay TP connection.

A connection can only belong to one channel.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">A connection within a channel identifies the sender and the receiver of this particular communication. The FlexRayTp module routes a Pdu through this connection.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">TransportProtocols::FlexRayTpConnection</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>FrTp</td><td colspan="2">FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
</table>

FrTpConPduRef		ReferenceParamDef	
BSW Description			
This reference is deprecated and will be removed in a future release. It has been rendered irrelevant by the introduction of PDU pools. Old description: Each value defines a PDU to be used for this connection. Thus each value is a PDU-ID given in FrTpPdu and this array cannot be longer than the array FrTpPdu. 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::FlexRayTpConnection.transmitPdu			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection		
BSW Parameter		**BSW Type**	
FrTpConPrioPdus		IntegerParamDef	
BSW Description			
This parameter defines the number of TxNPdus to which this connection has prioritized access. It must be ensured that the number of prioritized PDUs of all connections is smaller than the total number of TxNPdus in the associated PDU pool.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	**BSW Context**		
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection		
BSW Parameter		**BSW Type**	
FrTpLa		IntegerParamDef	
BSW Description			
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	Reference to the TP Address that is used by the TpNode.		
M2 Parameter			
TransportProtocols::TpNode.tpAddress			
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	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection	
BSW Parameter		**BSW Type**
FrTpMultRec		BooleanParamDef
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::FlexRayTpConnection.multicast		
Mapping Rule		**Mapping Type**
If multicast is used set this parameter to true.		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection	
BSW Parameter		**BSW Type**
FrTpRa		IntegerParamDef
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	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.	
M2 Parameter		
TransportProtocols::TpNode.tpAddress		
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	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection	
BSW Parameter		**BSW Type**
FrTpRxSdu		ParamConfContainerDef
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::FlexRayTpConnection.directTpSdu		
Mapping Rule		**Mapping Type**
Create container for every IPdu that is received by the Tp.		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection/FrTpRxSdu	
BSW Parameter		**BSW Type**

FrTpRxSduRef		ReferenceParamDef	
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/FrTpChannel/FrTpConnection/FrTpRxSdu		
BSW Parameter		**BSW Type**	
FrTpSduRxId		IntegerParamDef	
BSW Description			
Please note that this parameter is deprecated and will be removed in a future release. Old 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**		
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection		
BSW Parameter		**BSW Type**	
FrTpTxSdu		ParamConfContainerDef	
BSW Description			
Describes the Tx SDU			
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 for every IPdu that is transmitted by the Tp.			full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection/FrTpTxSdu	
BSW Parameter		**BSW Type**
FrTpSduTxId		IntegerParamDef
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**
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection/FrTpTxSdu
BSW Parameter	**BSW Type**
FrTpTxSduRef	ReferenceParamDef
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/FrTpChannel
BSW Parameter	**BSW Type**
FrTpGrpSeg	BooleanParamDef
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::FlexrayTpChannel.multicastSegmentation	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel
BSW Parameter	**BSW Type**
FrTpLm	EnumerationParamDef
BSW Description	
This specifies the maximum message length for the particular channel.	
M2 Template	**M2 Description**
M2 Parameter	

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpMaxAr		IntegerParamDef
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	
M2 Parameter		
TransportProtocols::FlexrayTpChannel.maxAr		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpMaxAs		IntegerParamDef
BSW Description		
This parameter defines the maximum number of trying t osend 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	
M2 Parameter		
TransportProtocols::FlexrayTpChannel.maxAs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpMaxBs		IntegerParamDef
BSW Description		
This parameter defines the 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 defines number of consecutive CFs between two FCs (block size). Valid values are 1 .. 16 when retry is activated, and 0 .. 255 otherwise.	
M2 Parameter		
TransportProtocols::FlexrayTpChannel.maxBs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**

FrTpMaxBufReq	IntegerParamDef
BSW Description	
Please note that this parameter is deprecated and will be removed in a future release. FrTpMaxWft will be used instead to configure the maximum number of wait frames on receiver side. On the sender side, timeCs defines the maximum time for retries. Old description: This parameter 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 Template	**M2 Description**
SystemTemplate	This parameter 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::FlexrayTpChannel.maxBufferRequest	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel
BSW Parameter	**BSW Type**
FrTpMaxFrIf	IntegerParamDef
BSW Description	
This parameter is deprecated and will be removed in future. Old 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::FlexrayTpChannel.maxFrIf	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel
BSW Parameter	**BSW Type**
FrTpMaxRn	IntegerParamDef
BSW Description	
This parameter defines the maximum number of retries (if retry is configured for the particular channel).	
M2 Template	**M2 Description**
SystemTemplate	This parameter defines the maximum number of retries (if retry is configured for the particular channel)
M2 Parameter	
TransportProtocols::FlexrayTpChannel.maxRetries	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpMaxWft		IntegerParamDef
BSW Description		
This parameter defines the maximal number of wait frames to be sent for a pending connection.		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines the maximal number of wait frames to be sent for a pending connection.	
M2 Parameter		
TransportProtocols::FlexrayTpChannel.maxFcWait		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpPdu		ParamConfContainerDef
BSW Description		
Container to hold the PDU parameters. ImplementationType: FrTp_PduInfoType		
M2 Template	**M2 Description**	
SystemTemplate	This is a PDU of the Transport Layer. The main purpose of the TP Layer is to segment and reassemble I-PDUs.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::NPdu		
Mapping Rule		**Mapping Type**
Create container for each NPdu that is available in the ecu extract.		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpPdu	
BSW Parameter		**BSW Type**
FrTpPduDirection		EnumerationParamDef
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		
Fibex::FibexCore::CoreTopology::CommConnectorPort.communicationDirection		
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	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpPdu	
BSW Parameter		**BSW Type**
FrTpPduId		IntegerParamDef
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. The FrTpPduId is only required for Rx N-PDUs (FrTpPduDirection == FRTP_RX), and should be omitted for Tx N-PDUs (FrTpPduDirection == FRTP_TX). ImplementationType: PduIdType		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**		
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpPdu		
BSW Parameter		**BSW Type**	
FrTpPduRef		ReferenceParamDef	
BSW Description			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel		
BSW Parameter		**BSW Type**	
FrTpPduFc		ParamConfContainerDef	
BSW Description			
Please note that this container with all references and parameters is deprecated and will be removed in a future release. It has been rendered irrelevant by the introduction of PDU pools. Old 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**		
SystemTemplate	Reference to the Flow Control NPdu.		
M2 Parameter			
TransportProtocols::FlexRayTpConnection.flowControlPdu			
Mapping Rule			**Mapping Type**
Create container for every FlowControlPdu that is available in the ecu extract.			full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpPduFc	
BSW Parameter		**BSW Type**
FrTpPduFcDirection		EnumerationParamDef
BSW Description		

This parameter is deprecated and will be removed in a future release. It has been rendered irrelevant by the introduction of PDU pools. Old 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	
Fibex::FibexCore::CoreTopology::CommConnectorPort.communicationDirection	
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**	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpPduFc	
BSW Parameter		**BSW Type**
FrTpPduFcId		IntegerParamDef
BSW Description		
This parameter is deprecated and will be removed in a future release. It has been rendered irrelevant by the introduction of PDU pools. Old 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**	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpPduFc	
BSW Parameter		**BSW Type**
FrTpPduFcRef		ReferenceParamDef
BSW Description		
This reference is deprecated and will be removed in a future release. It has been rendered irrelevant by the introduction of PDU pools.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**

FrTpStMin	FloatParamDef
BSW Description	
This parameter defines the minimum amount of time between two succeeding CFs of a 1:1 segmented transmission in seconds. Valid values are 0, 100μs, 200μs .. 900μs, 1ms, 2ms .. 127ms. The value can be changed at runtime using the FrTp_ChangeParameter interface. FrTpStMin must be an integer multiple of the cycle length multiplied with the multiplexing factor, i.e. FrTpStMin = n * FrIfGdCycle * m, where n is an integer >= 0 and m is the cycle multiplexor of those cycles where PDUs of the PDU pool are scheduled. Please note: Due to the scheduling strategies of FrTp, FrTpStMin can only be kept to a degree defined by the maximum temporal distance of the PDUs of a PDU pool within one FlexRay cycle.	
M2 Template	**M2 Description**
SystemTemplate	This attribute defines the minimum amount of time between two succeeding CFs of a 1:1 segmented transmission in seconds. Valid values are 0, 100μs, 200μs .. 900μs, 1ms, 2ms .. 127ms.
M2 Parameter	
TransportProtocols::FlexrayTpChannel.minimumSeparationTime	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel
BSW Parameter	**BSW Type**
FrTpStMinGrpSeg	FloatParamDef
BSW Description	
This parameter defines the minimum amount of time between two succeeding CFs of a 1:n segmented transmission in seconds. Valid values are 0, 100μs, 200μs ... 900μs, 1ms, 2ms .. 127ms. The value can be changed at runtime using the FrTp_ChangeParameter interface. FrTpStMinGrpSeg must be an integer multiple of the cycle length multiplied with the multiplexing factor, i.e. FrTpStMinGrpSeg = n * FrIfGdCycle * m, where n is an integer >= 0 and m is the cycle multiplexor of those cycles where PDUs of the PDU pool are scheduled. Please note: Due to the scheduling strategies of FrTp, FrTpStMinGrpSeg can only be kept to a degree defined by the maximum temporal distance of the PDUs of a PDU pool within one FlexRay cycle.	
M2 Template	**M2 Description**
SystemTemplate	This attribute defines the minimum amount of time (separation Time) between two succeeding CFs of a 1:n segmented transmission in seconds.
M2 Parameter	
TransportProtocols::FlexrayTpChannel.minimumMulticastSeparationTime	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel
BSW Parameter	**BSW Type**
FrTpTc	BooleanParamDef
BSW Description	
With this switch Transmit Cancellation and Receive Cancellation can be turned on or off for this channel.	
M2 Template	**M2 Description**

SystemTemplate	With this switch Tx and Rx Cancellation can be turned on or off.
M2 Parameter	
TransportProtocols::FlexrayTpChannel.transmitCancellation	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel
BSW Parameter	**BSW Type**
FrTpTimeBr	FloatParamDef
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 FRTP_TIME_BR + (FRTP_TIMEOUT_AR * FRTP_MAX_AR) < FRTP_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 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.
M2 Parameter	
TransportProtocols::FlexrayTpChannel.timeBr	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel
BSW Parameter	**BSW Type**
FrTpTimeBuffer	FloatParamDef
BSW Description	
Please note that this parameter is deprecated and will be removed in a future release. FrTpTimeBr will be used instead to configure the delay between two wait frames (and thus two buffer requests) on receiver side. On sender side, the main task cycle will be used. Old 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 in seconds of waiting for the next try to get a Tx or Rx buffer.
M2 Parameter	
TransportProtocols::FlexrayTpChannel.timeBuffer	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel
BSW Parameter	**BSW Type**

FrTpTimeCs			FloatParamDef
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 FRTP_TIME_CS + (FRTP_TIMEOUT_AS * FRTP_MAX_AS) < FRTP_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	This parameter 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 FC.		
M2 Parameter			
TransportProtocols::FlexrayTpChannel.timeCs			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel		
BSW Parameter		**BSW Type**	
FrTpTimeFrIf		FloatParamDef	
BSW Description			
This parameter is deprecated and will be removed in future. Old description: This parameter defines the time in seconds of waiting for the next try to send via FrIf_Transmit.			
M2 Template	**M2 Description**		
SystemTemplate	This parameter defines the time in seconds of waiting for the next try to send via FrIf_Transmit.		
M2 Parameter			
TransportProtocols::FlexrayTpChannel.timeFrIf			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpTimeoutAr		FloatParamDef
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 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::FlexrayTpChannel.timeoutAr		

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpTimeoutAs		FloatParamDef
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 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	
M2 Parameter		
TransportProtocols::FlexrayTpChannel.timeoutAs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpTimeoutBs		FloatParamDef
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::FlexrayTpChannel.timeoutBs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpTimeoutCr		FloatParamDef
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 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 Parameter		
TransportProtocols::FlexrayTpChannel.timeoutCr		
Mapping Rule		**Mapping Type**

1:1 mapping	full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpUsePduFc		BooleanParamDef
BSW Description		
Please note that this parameter is deprecated and will be removed in a future release. It has been rendered irrelevant by the introduction of PDU pools. Old description: This switch defines, whether within this channel the dedicated FC/ACK PDU (FrTpPduFc) shall be used or not. If this is not used FC / ACK frames are sent using the normal IDs, otherwise only FrTpPduFc shall be used for sending / receiving FC / ACK frames.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

9.7 FlexRayIsoTP Mapping

BSW Module	BSW Context	
FrIsoTp	FrIsoTp	
BSW Parameter		**BSW Type**
FrIsoTpGeneral		ParamConfContainerDef
BSW Description		
This container contains the general configuration parameters of the FlexRay ISO Transport Protocol module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpGeneral	
BSW Parameter		**BSW Type**
FrIsoTpAckRt		BooleanParamDef
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	
FrIsoTp	FrIsoTp/FrIsoTpGeneral	
BSW Parameter		**BSW Type**
FrIsoTpChanNum		IntegerParamDef
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	
FrIsoTp	FrIsoTp/FrIsoTpGeneral	
BSW Parameter		**BSW Type**
FrIsoTpDevErrorDetect		BooleanParamDef

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		
FrIsoTp	FrIsoTp/FrIsoTpGeneral		
BSW Parameter		**BSW Type**	
FrIsoTpFullDuplexEnable		BooleanParamDef	
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::FlexrayIsoTpEcu.fullDuplexEnabled			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIsoTp	FrIsoTp/FrIsoTpGeneral		
BSW Parameter		**BSW Type**	
FrIsoTpMainFuncCycle		FloatParamDef	
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 AUTOSAR TP. Specified in seconds.		
M2 Parameter			
TransportProtocols::FlexrayIsoTpEcu.cycleTimeMainFunction			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpGeneral	
BSW Parameter		**BSW Type**
FrIsoTpTransmitCancellation		BooleanParamDef
BSW Description		

Preprocessor switch for enabling Transmit Cancellation and Receive Cancellation. True: Transmit/Receive Cancellation is enabled False: Transmit/Receive Cancellation is disabled		
M2 Template	**M2 Description**	
SystemTemplate	With this switch Tx and Rx Cancellation can be turned on or off.	
M2 Parameter		
TransportProtocols::FlexrayIsoTpEcu.transmitCancellation		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIsoTp	FrIsoTp/FrIsoTpGeneral	
BSW Parameter		**BSW Type**
FrIsoTpUnknownMsgLength		BooleanParamDef
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**	
FrIsoTp	FrIsoTp/FrIsoTpGeneral	
BSW Parameter		**BSW Type**
FrIsoTpVersionInfoApi		BooleanParamDef
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**	
FrIsoTp	FrIsoTp	
BSW Parameter		**BSW Type**
FrIsoTpMultipleConfig		ParamConfContainerDef
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	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig	
BSW Parameter		**BSW Type**
FrIsoTpConnection		ParamConfContainerDef
BSW Description		
This container contains the connection specific parameters to transfer N-PDUs via FlexRay ISO TP.		
M2 Template	**M2 Description**	
SystemTemplate	A connection identifies the sender and the receiver of this particular communication. The FlexRayIsoTp module routes a Pdu through this connection.	
M2 Parameter		
TransportProtocols::FlexrayIsoTpConnection		
Mapping Rule		**Mapping Type**
Create container for each FlexrayIsoTpConnection elementthat is available in the ECU Extract.		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection	
BSW Parameter		**BSW Type**
FrIsoTpBandwidthLimitation		BooleanParamDef
BSW Description		
This parameter indicates wheather the connection requires a bandwidth limitation or not. If FrIsoTpBandwidthLimitation=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 limitation or not.	
M2 Parameter		
TransportProtocols::FlexrayIsoTpConnection.bandwidthLimitation		
Mapping Rule		**Mapping Type**
1:1 mapping		local

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection	
BSW Parameter		**BSW Type**
FrIsoTpConCtrlRef		ReferenceParamDef
BSW Description		
FrIsoTpConnectionControlReference: This parameter defines a reference to a connection control container.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the connection control.	
M2 Parameter		
TransportProtocols::FlexrayIsoTpConnection.tpConnectionControl		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection	
BSW Parameter		**BSW Type**
FrIsoTpLa		IntegerParamDef
BSW Description		
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.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the TP Address that is used by the TpNode	
M2 Parameter		
TransportProtocols::TpNode.tpAddress		
Mapping Rule		**Mapping Type**
Use TpAddress that is referenced by the transmitter TpNode.		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection	
BSW Parameter		**BSW Type**
FrIsoTpMultipleReceiverCon		BooleanParamDef
BSW Description		
This parameter defines, whether this connection is an 1:1 ('false') or an 1:n ('true') connection. If this parameter is set to true the size of associated N-SDUs shall not be larger than the size of the associated N-PDUs.		
M2 Template	**M2 Description**	
SystemTemplate	TP address for 1:n connections.	
M2 Parameter		
TransportProtocols::FlexrayIsoTpConnection.multicast		
Mapping Rule		**Mapping Type**
If FlexRayTpConnection contains a mulicast reference to TpAddress than set this parameter to true		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection	
BSW Parameter		**BSW Type**
FrIsoTpRa		IntegerParamDef
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.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the TP Address that is used by the TpNode.	
M2 Parameter		
TransportProtocols::TpNode.tpAddress		
Mapping Rule		**Mapping Type**
Use TpAddress that is referenced by the receiver TpNode.		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection	
BSW Parameter		**BSW Type**
FrIsoTpRxPduPoolRef		ReferenceParamDef

BSW Description	
This parameter defines a reference to a RxPduPool.	
M2 Template	**M2 Description**
SystemTemplate	A connection has a reference to a set of NPdus (FrTpRxPduPool) which are defined for receiving data via this particular connection.
M2 Parameter	
TransportProtocols::FlexrayIsoTpConnection.rxPduPool	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection	
BSW Parameter		**BSW Type**
FrIsoTpRxSdu		ParamConfContainerDef
BSW Description		
This parameter defines the Rx Service Data Unit Identifier (Sdu Id) which uniquely identifies a data transfer (inter-module communication) between FrIsoTp and PDUR.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::IPdu		
Mapping Rule		**Mapping Type**
Create container for every IPdu that is received by FrTp.		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection/FrIsoTpRxSdu	
BSW Parameter		**BSW Type**
FrIsoTpRxSduId		IntegerParamDef
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	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection/FrIsoTpRxSdu	
BSW Parameter		**BSW Type**
FrIsoTpRxSduRef		ReferenceParamDef
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**	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection	
BSW Parameter		**BSW Type**
FrIsoTpTxPduPoolRef		ReferenceParamDef
BSW Description		
This parameter defines a reference to a TxPduPool.		
M2 Template	**M2 Description**	
SystemTemplate	A connection has a reference to a set of NPdus (FrTpTxPduPool) which are defined for sending data via this particular connection.	
M2 Parameter		
TransportProtocols::FlexrayIsoTpConnection.txPduPool		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection	
BSW Parameter		**BSW Type**
FrIsoTpTxSdu		ParamConfContainerDef
BSW Description		
This parameter defines the Tx Service Data Unit Identifier (Sdu Id) which uniquely identifies a data transfer (inter-module communication) between FrIsoTp and PDUR.		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::IPdu		
Mapping Rule		**Mapping Type**
Create contaier for each IPdu that is transmitted from Tp module to an upper layer module.		full

BSW Module	**BSW Context**	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection/FrIsoTpTxSdu	
BSW Parameter		**BSW Type**
FrIsoTpTxSduId		IntegerParamDef
BSW Description		
This is a unique identifier for a to be transmitted message from the PduR to the FrIsoTp. ImplementationType: PduIdType		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnection/FrIsoTpTxSdu	
BSW Parameter		**BSW Type**
FrIsoTpTxSduRef		ReferenceParamDef
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	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig	
BSW Parameter		**BSW Type**
FrIsoTpConnectionControl		ParamConfContainerDef
BSW Description		
This container contains the configuration parameters to control a FlexRay ISO TP connection.		
M2 Template	**M2 Description**	
SystemTemplate	Configuration parameters to control a FlexRay TP connection.	
M2 Parameter		
TransportProtocols::FlexrayIsoTpConnectionControl		
Mapping Rule		**Mapping Type**
Create container for each FlexrayIsoTpConnectionControl element that is available in the Ecu Extract.		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		**BSW Type**
FrIsoTpAckType		EnumerationParamDef
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. For FlexrayIsoTp only noAck and ackWithRt shall be used.	
M2 Parameter		
TransportProtocols::FlexrayIsoTpConnectionControl.ackType		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		**BSW Type**
FrIsoTpMaxAr		IntegerParamDef
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::FlexrayIsoTpConnectionControl.maxAr	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		**BSW Type**
FrIsoTpMaxAs		IntegerParamDef
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::FlexrayIsoTpConnectionControl.maxAs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		**BSW Type**
FrIsoTpMaxBufferSize		IntegerParamDef
BSW Description		
Limits the maximal buffer size the FrIsoTp 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 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 Parameter		
TransportProtocols::FlexrayIsoTpConnectionControl.maxBufferSize		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		**BSW Type**
FrIsoTpMaxFCWait		IntegerParamDef
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 Flow-State "WAIT".	
M2 Parameter		
TransportProtocols::FlexrayIsoTpConnectionControl.maxFcWait		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl		
BSW Parameter		BSW Type	
FrIsoTpMaxFrIf		IntegerParamDef	
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::FlexrayIsoTpConnectionControl.maxFrIf			
Mapping Rule			Mapping Type
1:1 mapping			full

BSW Module	BSW Context		
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl		
BSW Parameter		BSW Type	
FrIsoTpMaxNbrOfNPduPerCycle		IntegerParamDef	
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::FlexrayIsoTpConnectionControl.maxNumberOfNpduPerCycle			
Mapping Rule			Mapping Type
1:1 mapping			full

BSW Module	BSW Context		
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl		
BSW Parameter		BSW Type	
FrIsoTpMaxRn		IntegerParamDef	
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::FlexrayIsoTpConnectionControl.maxRetries			
Mapping Rule			Mapping Type
1:1 mapping			full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		BSW Type
FrIsoTpSCexp		IntegerParamDef
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 FrIsoTp 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::FlexrayIsoTpConnectionControl.separationCycleExponent		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		**BSW Type**
FrIsoTpTimeBr		FloatParamDef
BSW Description		
This parameter defines the time in seconds the FrIsoTp requires to transmit a corresponding Flow-Control 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::FlexrayIsoTpConnectionControl.timeBr		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		**BSW Type**
FrIsoTpTimeBuffer		FloatParamDef
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. This parameter is equivalent to the temporal distance between two FC.WT N-Pdus in case the buffer request returns busy.	
M2 Parameter		
TransportProtocols::FlexrayIsoTpConnectionControl.timeBuffer		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		**BSW Type**
FrIsoTpTimeFrIf		FloatParamDef
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::FlexrayIsoTpConnectionControl.timeFrIf		

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		**BSW Type**
FrIsoTpTimeoutAr		FloatParamDef
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::FlexrayIsoTpConnectionControl.timeoutAr		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		**BSW Type**
FrIsoTpTimeoutAs		FloatParamDef
BSW Description		
This parameter specifies the timeout in seconds the FrIf shall confirm a transmitted Pdu to the FrIsoTp.		
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 on the sender side	
M2 Parameter		
TransportProtocols::FlexrayIsoTpConnectionControl.timeoutAs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		**BSW Type**
FrIsoTpTimeoutBs		FloatParamDef
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::FlexrayIsoTpConnectionControl.timeoutBs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpConnectionControl	
BSW Parameter		**BSW Type**
FrIsoTpTimeoutCr		FloatParamDef
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::FlexrayIsoTpConnectionControl.timeoutCr		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig	
BSW Parameter		**BSW Type**
FrIsoTpRxPduPool		ParamConfContainerDef
BSW Description		
This container contains all Pdus that are assigned to that Pdu Pool.		
M2 Template	**M2 Description**	
SystemTemplate	FlexrayTpPduPool is a set of N-PDUs which are defined for FrTp sending or receiving purpose.	
M2 Parameter		
TransportProtocols::FlexrayIsoTpPduPool		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpRxPduPool	
BSW Parameter		**BSW Type**
FrIsoTpRxPdu		ParamConfContainerDef
BSW Description		
Container to hold the PDU parameters. ImplementationType: PduInfoType		
M2 Template	**M2 Description**	
SystemTemplate	This is a PDU of the Transport Layer. The main purpose of the TP Layer is to segment and reassemble I-PDUs.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::NPdu		
Mapping Rule		**Mapping Type**
Create container for each NPdu that is received by FrTp.		full

BSW Module	BSW Context	
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpRxPduPool/FrIsoTpRxPdu	
BSW Parameter		**BSW Type**
FrIsoTpRxPduId		IntegerParamDef
BSW Description		

<table>
<tr><td colspan="3">This is a unique identifier for a received message which is forwarded from the FrIf to the FrIsoTp.

ImplementationType: PduIdType</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="3">BSW Context</th></tr>
<tr><td>FrIsoTp</td><td colspan="3">FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpRxPduPool/FrIsoTpRxPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrIsoTpRxPduRef</td><td colspan="2">ReferenceParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Reference to a PDU in the global PDU structure.</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>FrIsoTp</td><td colspan="3">FrIsoTp/FrIsoTpMultipleConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrIsoTpTxPduPool</td><td colspan="2">ParamConfContainerDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This container contains all Pdus that are assigned to that Pdu Pool.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">FlexrayTpPduPool is a set of N-PDUs which are defined for FrTp sending or receiving purpose.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">TransportProtocols::FlexrayIsoTpPduPool</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="2">BSW Context</th></tr>
<tr><td>FrIsoTp</td><td colspan="2">FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpTxPduPool</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIsoTpTxPdu</td><td>ParamConfContainerDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Container to hold the PDU parameters.

ImplementationType: PduInfoType</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">This is a PDU of the Transport Layer.
The main purpose of the TP Layer is to segment and reassemble I-PDUs.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreCommunication::NPdu</td></tr>
</table>

Mapping Rule	Mapping Type
Create container for each NPdu that is referenced by the txPduPool in the ecu extract.	full

BSW Module	BSW Context		
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpTxPduPool/FrIsoTpTxPdu		
BSW Parameter		**BSW Type**	
FrIsoTpTxConfirmationPduId		IntegerParamDef	
BSW Description			
Handle Id to be used by the FrIf to confirm the transmission of the FrIsoTpTxPdu to the FrIf module.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrIsoTp	FrIsoTp/FrIsoTpMultipleConfig/FrIsoTpTxPduPool/FrIsoTpTxPdu		
BSW Parameter		**BSW Type**	
FrIsoTpTxPduRef		ReferenceParamDef	
BSW Description			
Reference to a PDU in the global PDU structure.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

9.8 FlexRay Nm Mapping

BSW Module	BSW Context	
FrNm	FrNm	
BSW Parameter		**BSW Type**
FrNmChannelConfig		ParamConfContainerDef
BSW Description		
This container contains all configuration parameters of FlexRay NM configured from the channel perspective.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig	
BSW Parameter		**BSW Type**
FrNmChannelIdentifiers		ParamConfContainerDef
BSW Description		
This container contains instance specific identifiers related to the respective FlexRay Channel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmActiveWakeupBitEnabled		BooleanParamDef
BSW Description		
Enables/Disables the handling of the Active Wakeup Bit in the FrNm module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmCarWakeUpBitPosition		IntegerParamDef
BSW Description		
Specifies the Bit position of the CWU within the NM-Message.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmCarWakeUpBytePosition		IntegerParamDef
BSW Description		
Specifies the Byte position of the CWU within the NM-Message.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmCarWakeUpFilterEnabled		BooleanParamDef
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmCarWakeUpFilterNodeId		IntegerParamDef
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 FrNmCarWakeUpFilterNodeId is considered as CWU request.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmCarWakeUpRxEnabled		BooleanParamDef
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmChannelHandle		SymbolicNameReferenceParamDef
BSW Description		
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.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmChannelIdRef		SymbolicNameReferenceParamDef
BSW Description		
NM-Network identifier configured for the respective FlexRay Channel. It is used for referring to the respective NM-Network handle.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmDataEnabled		BooleanParamDef
BSW Description		

This parameter is deprecated and will be removed in future. Old description: Enable the separated sending of NM-Data.		
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		
FlexrayTopology::FlexrayCluster.nmDataCycle		
Mapping Rule		**Mapping Type**
Is set to true if nmDataCycle is provided.		full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmNodeId		IntegerParamDef
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	An ECUs NM address on the referenced channel.	
M2 Parameter		
CoreTopology::CommunicationConnector.nmAddress		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmPduLength		IntegerParamDef
BSW Description		
Length of the NM-Data PDU.		
M2 Template	**M2 Description**	
SystemTemplate	The size of the NmPDU in bits.	
M2 Parameter		
CoreCommunication::NmPdu.length		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmPduScheduleVariant		EnumerationParamDef
BSW Description		
This parameter is deprecated and will be removed in future.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmPnEnabled		BooleanParamDef
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	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmPnEraCalcEnabled		BooleanParamDef
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**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmPnEraRxNSduRef		ReferenceParamDef
BSW Description		
Reference to a Pdu in the COM-Stack. The SduRef is required for every FrNm Channel, because ERA is reported per channel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmRxPdu		ParamConfContainerDef
BSW Description		
This container describes the FlexRay NM RX PDU:s.		
M2 Template	**M2 Description**	
SystemTemplate	receive NM Pdu	
M2 Parameter		
CoreCommunication::NmPdu		
Mapping Rule		**Mapping Type**
Create Container if the regarded NmNode recieves a Pdu		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers/FrNmRxPdu	
BSW Parameter		**BSW Type**
FrNmRxPduContainsData		BooleanParamDef
BSW Description		
This parameted defines if the PDU contains NM Data.		
M2 Template	**M2 Description**	
SystemTemplate	Defines if the Pdu contains NM Data. If the NmPdu does not aggregate any ISignalToIPduMappings it still may contain UserData that is set via Nm_SetUserData(). If the ISignalToIPduMapping exists then the nmDataInformation shall be ignored	
M2 Parameter		
CoreCommunication::NmPdu.nmDataInformation,CoreCommunication::NmPdu.iSignalToIPdu Mapping		
Mapping Rule		**Mapping Type**
Set to true if either the NmPdu aggregates one or more iSignalToIPduMappings, or - if none are aggregated - if nmDataInformation is true. Set to false in all other cases		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers/FrNmRxPdu	
BSW Parameter		**BSW Type**
FrNmRxPduContainsVote		BooleanParamDef
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		
CoreCommunication::NmPdu.nmVoteInformation		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers/FrNmRxPdu	
BSW Parameter		**BSW Type**
FrNmRxPduId		IntegerParamDef
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/FrNmChannelIdentifiers/FrNmRxPdu		
BSW Parameter		**BSW Type**	
FrNmRxPduRef		ReferenceParamDef	
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/FrNmChannelIdentifiers		
BSW Parameter		**BSW Type**	
FrNmTxPdu		ParamConfContainerDef	
BSW Description			
This container describes the FlexRay NM TX PDU:s.			
M2 Template	**M2 Description**		
SystemTemplate	transmit NM Pdu		
M2 Parameter			
CoreCommunication::NmPdu			
Mapping Rule			**Mapping Type**
Create Container if the regarded NmNode transmits a Pdu			full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers/FrNmTxPdu	
BSW Parameter		**BSW Type**
FrNmTxPduContainsData		BooleanParamDef
BSW Description		
This parameted defines if the PDU contains NM Data.		
M2 Template	**M2 Description**	

SystemTemplate	Defines if the Pdu contains NM Data. If the NmPdu does not aggregate any ISignalToIPduMappings it still may contain UserData that is set via Nm_SetUserData(). If the ISignalToIPduMapping exists then the nmDataInformation shall be ignored	
M2 Parameter		
CoreCommunication::NmPdu.nmDataInformation,CoreCommunication::NmPdu.iSignalToIPdu Mapping		
Mapping Rule		**Mapping Type**
Set to true if either the NmPdu aggregates one or more iSignalToIPduMappings, or - if none are aggregated - if nmDataInformation is true. Set to false in all other cases		full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers/FrNmTxPdu	
BSW Parameter		**BSW Type**
FrNmTxPduContainsVote		BooleanParamDef
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		
CoreCommunication::NmPdu.nmVoteInformation		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelIdentifiers/FrNmTxPdu	
BSW Parameter		**BSW Type**
FrNmTxPduRef		ReferenceParamDef
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/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmUserDataTxPdu		ParamConfContainerDef
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		
CoreCommunication::NmPdu.iSignalToIPduMapping		

Mapping Rule	Mapping Type
Create container for each NmPdu transmitted by the regarded ECU 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/FrNmChannelIdentifiers/FrNmUserDataTxPdu	
BSW Parameter		**BSW Type**
FrNmTxUserDataPduId		IntegerParamDef
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/FrNmChannelIdentifiers/FrNmUserDataTxPdu	
BSW Parameter		**BSW Type**
FrNmTxUserDataPduRef		ReferenceParamDef
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	
BSW Parameter		**BSW Type**
FrNmChannelTiming		ParamConfContainerDef
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/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmDataCycle		EnumerationParamDef
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		
FlexrayTopology::FlexrayCluster.nmDataCycle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmMsgTimeoutTime		IntegerParamDef
BSW Description		
Timeout of a NM-message [number of communication cycles]. It determines how long the NM shall wait with notification of transmission failure while communication errors occur on the bus.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmReadySleepCnt		IntegerParamDef
BSW Description		
FrNm switches to bus sleep mode at the end of the FrNmReadySleepCnt+1 repetition cycle without any NM vote. E.g. on a value of "1", the NM-State Machine will leave the Ready Sleep State after two NM Repetition Cycles with no "keep awake" votes.		
M2 Template	**M2 Description**	
SystemTemplate	FrNm switches to bus sleep mode at the end of the FrNmReadySleepCnt+1 repetition cycle without any NM vote. E.g. on a value of "1", the NM-State Machine will leave the Ready Sleep State after two NM Repetition Cycle with no "keep awake" votes.	
M2 Parameter		
FlexrayTopology::FlexrayCluster.nmRepetitionCycle FlexrayTopology::FlexRayCommunicationConnector.nmReadySleepTime		
Mapping Rule		**Mapping Type**
If nmReadySleepCount is available this parameter shall be mapped 1:1. If nmReadySleepTime is available the following calculation formula shall be taken: FrNmReadySleepCount = (nmReadySleepTime/cycle/nmRepetition Cycle)-1		full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannelTiming	
BSW Parameter		**BSW Type**

FrNmRemoteSleepIndTime		IntegerParamDef	
BSW Description			
Timeout for Remote Sleep Indication. It defines the time [number of communication cycles] 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			
FlexrayTopology::FlexrayCluster.nmRemoteSleepIndicationTime			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
FrNm	FrNm/FrNmChannelConfig/FrNmChannelTiming		
BSW Parameter		**BSW Type**	
FrNmRepeatMessageTime		FloatParamDef	
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 in seconds. Defines the time how long the NM shall stay in the Repeat Message State.		
M2 Parameter			
FlexrayTopology::FlexrayCluster.nmRepeatMessageTime			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
FrNm	FrNm/FrNmChannelConfig/FrNmChannelTiming		
BSW Parameter		**BSW Type**	
FrNmRepetitionCycle		EnumerationParamDef	
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			
FlexrayTopology::FlexrayCluster.nmRepetitionCycle			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**
FrNm	FrNm/FrNmChannelConfig/FrNmChannelTiming

BSW Parameter		BSW Type	
FrNmSyncLossTimer		FloatParamDef	
BSW Description			
This parameter is deprecated and will be removed in future.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrNm	FrNm/FrNmChannelConfig/FrNmChannelTiming		
BSW Parameter		**BSW Type**	
FrNmVotingCycle		EnumerationParamDef	
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			
FlexrayTopology::FlexrayCluster.nmVotingCycle			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrNm	FrNm		
BSW Parameter		**BSW Type**	
FrNmGlobalConfig		ParamConfContainerDef	
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		ParamConfContainerDef	
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		IntegerParamDef
BSW Description		
Number of AUTOSAR FR NM clusters allowed within one ECU.		
M2 Template	**M2 Description**	
SystemTemplate	Collection of NM Clusters	
M2 Parameter		
FlexrayTopology::FlexrayCluster		
Mapping Rule		**Mapping Type**
Count aggregated Clusters.		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalConstants	
BSW Parameter		**BSW Type**
FrNmPduScheduleVariant		EnumerationParamDef
BSW Description		
This parameter defines the PDU scheduling variant that should be used: 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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig	
BSW Parameter		**BSW Type**
FrNmGlobalFeatures		ParamConfContainerDef
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		BooleanParamDef	
BSW Description			
Pre-processor switch for enabling the bus synchronisation.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures		
BSW Parameter		**BSW Type**	
FrNmComUserDataSupport		BooleanParamDef	
BSW Description			
Enable/disable the user data support.			
M2 Template	**M2 Description**		
SystemTemplate	The ISignalToIPduMapping aggregation is used to describe NmUserData that is transmitted in the NmPdu.		
M2 Parameter			
SystemTemplate::Fibex::FibexCore::CoreCommunication::NmPdu.iSignalToIPduMapping			
Mapping Rule			**Mapping Type**
If the regarded EcuInstance sends a NmPdu that contains an ISignalToIPdu Mapping that refers to an ISignal then the value of this parameter shall be true.			full

BSW Module	**BSW Context**		
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures		
BSW Parameter		**BSW Type**	
FrNmControlBitVectorEnabled		DerivedBooleanParamDef	
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			
FlexrayTopology::FlexrayCommunicationController.nmControlBitVectorEnabled			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures		
BSW Parameter		**BSW Type**	
FrNmCoordinatorSyncSupport		BooleanParamDef	
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	BooleanParamDef
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	BooleanParamDef
BSW Description	
Pre-processor switch for enabling the support of dual channel transmission and reception of NM messages.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures
BSW Parameter	**BSW Type**
FrNmHwVoteEnable	BooleanParamDef
BSW Description	
Pre-processor switch for enabling the processing of FlexRay Hardware aggregated NM-Votes.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures
BSW Parameter	**BSW Type**

FrNmNmDataDisabled		BooleanParamDef	
BSW Description			
Pre-processor switch for enabling the transmission of NM-Data.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures		
BSW Parameter		**BSW Type**	
FrNmNodeDetectionEnabled		DerivedBooleanParamDef	
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	Enable/disable the node detection functionality.		
M2 Parameter			
CoreTopology::CommunicationCluster.nmNodeDetectionEnabled			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures		
BSW Parameter		**BSW Type**	
FrNmPassiveModeEnabled		DerivedBooleanParamDef	
BSW Description			
Pre-processor switch for enabling Passive Mode Configuration support. calculationFormula = Equal(NmPassiveModeEnabled)			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures		
BSW Parameter		**BSW Type**	
FrNmPduRxIndicationEnabled		BooleanParamDef	
BSW Description			
Pre-processor switch for enabling PDU reception indication.			
M2 Template	**M2 Description**		
M2 Parameter			

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmPnEiraCalcEnabled		BooleanParamDef
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**
		local

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmPnEiraRxNSduRef		ReferenceParamDef
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		ParamConfContainerDef
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

<table>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrNmPnFilterMaskByte</td><td colspan="2">ParamConfContainerDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Filter mask byte configuration</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>TPS_SYST</td><td colspan="3">For one EcuInstance all contributing pncWakeupDataMask will be bitwise ORed to obtain aggregated pncWakeupDataMask value for this ECU. Since the pncWakeupDataMask is calculated over the whole payload (8 Byte) of the NmPdu, the leading Bytes of this aggregated pncWakeupDataMask shall be ignored based on the System.pncVectorOffset value.

In order to get the FrNmPnFilterMaskByteIndex and FrNmPnFilterMaskByteValue for all the bytes aggregated pncWakeupDataMask shall be processed in a littleEndian way.
E.g. if pncVectorOffset = 2 and aggregated pncWakeupDataMask has the value 2ˆ 63 this will end up in a FrNmPnFilterMaskByte with FrNmPnFilterMaskByteIndex = 5 and FrNmPnFilterMaskByteValue = 128.</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">Since the pncWakeupDataMask element is not available in FrCommunicationConnector the configuration of this parameters needs to be done localy.</td><td>partial</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>FrNm</td><td colspan="3">FrNm/FrNmGlobalConfig/FrNmGlobalFeatures/FrNmPnInfo/FrNmPnFilterMaskByte</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrNmPnFilterMaskByteIndex</td><td colspan="2">IntegerParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Index of the filter mask byte. Specifies the position within the filter mask byte array.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>TPS_SYST</td><td colspan="3">see FrNmPnFilterMaskByte</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">see FrNmPnFilterMaskByte</td><td>partial</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>FrNm</td><td colspan="3">FrNm/FrNmGlobalConfig/FrNmGlobalFeatures/FrNmPnInfo/FrNmPnFilterMaskByte</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrNmPnFilterMaskByteValue</td><td colspan="2">IntegerParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Parameter to configure the filter mask byte.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>TPS_SYST</td><td colspan="3">see FrNmPnFilterMaskByte</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">see FrNmPnFilterMaskByte</td><td>partial</td></tr>
</table>

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures/FrNmPnInfo	
BSW Parameter		**BSW Type**
FrNmPnInfoLength		IntegerParamDef
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		IntegerParamDef
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		FloatParamDef
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	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmRemoteSleepIndicationEnabled		DerivedBooleanParamDef
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		
FlexrayTopology::FlexrayCluster.nmRemoteSleepIndicationTime		
Mapping Rule		**Mapping Type**
Set to true if nmRemoteSleepIndicationTime is provided.		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmRepeatMessageBitEnabled		DerivedBooleanParamDef
BSW Description		
Pre-processor switch for enabling the repeat message bit support. calculationFormula = If (FrNmControlBitVectorEnabled == False) then Equal(False) else Equal(False or True)		
M2 Template	**M2 Description**	
SystemTemplate	switch for enabling support for repeat message	
M2 Parameter		
CoreTopology::CommunicationCluster.nmRepeatMessageSupport		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmSourceNodeIdentifierEnabled		BooleanParamDef
BSW Description		
Pre-processor switch for enabling SourceNodeIdentifier support.		
M2 Template	**M2 Description**	
SystemTemplate	Enable/disable the source node identifier.	
M2 Parameter		
CoreTopology::CommunicationCluster.nmNodeIdEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmStateChangeIndicationEnabled		BooleanParamDef
BSW Description		
Pre-processor switch for enabling state change indication.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmUserDataEnabled		BooleanParamDef
BSW Description		
Pre-processor switch for enabling user data support.		
M2 Template	**M2 Description**	
SystemTemplate	Enable/disable the user data support.	
M2 Parameter		
CoreCommunication::NmPdu.nmUserDataLength		
Mapping Rule		**Mapping Type**
Set to true if the nmUserDataLength attribute is provided for NmPdus that are transmitted by the regarded Ecu.		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig	
BSW Parameter		**BSW Type**
FrNmGlobalProperties		ParamConfContainerDef
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		BooleanParamDef
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		BooleanParamDef
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		BooleanParamDef
BSW Description		
Pre-processor switch for enabling version info API support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

9.9 FlexRay SM Mapping

BSW Module	BSW Context	
FrSm	FrSm	
BSW Parameter		**BSW Type**
FrSmCluster		ParamConfContainerDef
BSW Description		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
FrSm	FrSm/FrSmCluster	
BSW Parameter		**BSW Type**
FrSmCheckWakeupReason		BooleanParamDef
BSW Description		
If FrSMCheckWakeupReason is true, the FrSM will check the wakeup reason in order to skip the wakeup in case of wakeup by bus. If FrSMCheckWakeupReason is false, the FrSM will always try to perform a wakeup.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
FrSm	FrSm/FrSmCluster	
BSW Parameter		**BSW Type**
FrSmDelayStartupWithoutWakeup		BooleanParamDef
BSW Description		
If true, timer t1 shall be started instead of immediately calling FrIf_AllowColdstart in case of a startup without wakeup.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
FrSm	FrSm/FrSmCluster	
BSW Parameter		**BSW Type**
FrSmDurationT1		IntegerParamDef
BSW Description		

The duration of timer t1 as multiples of the cycle time of the FrSm main function. A value of 0 shall imply that the timer is not used.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**		
FrSm	FrSm/FrSmCluster		
BSW Parameter		**BSW Type**	
FrSmDurationT2		IntegerParamDef	
BSW Description			
The duration of timer t2 as multiples of the cycle time of the FrSm main function. The value of this parameter shall be larger than the value of FrSMDurationT1 parameter. A value of 0 shall imply that the timer is not used.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	**BSW Context**		
FrSm	FrSm/FrSmCluster		
BSW Parameter		**BSW Type**	
FrSmDurationT3		IntegerParamDef	
BSW Description			
The duration of timer t3 as multiples of the cycle time of the FrSm main function. The value of this parameter shall be larger than the value of FrSMDurationT1 parameter. A value of 0 shall imply that the timer is not used.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	**BSW Context**	
FrSm	FrSm/FrSmCluster	
BSW Parameter		**BSW Type**
FrSmFrIfClusterRef		SymbolicNameReferenceParamDef
BSW Description		
References the cluster configuration in the FlexRay Interface configuration. Note that the assigned controllers and transceivers are defined in the FrIf configuration and can be accessed via this reference.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context	
FrSm	FrSm/FrSmCluster	
BSW Parameter		**BSW Type**
FrSmIsColdstartEcu		BooleanParamDef
BSW Description		
True: The ECU is a coldstart node for this FlexRay cluster. False: The ECU is no coldstart node for this FlexRay cluster.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
FrSm	FrSm/FrSmCluster	
BSW Parameter		**BSW Type**
FrSmIsWakeupEcu		BooleanParamDef
BSW Description		
True: FrSm shall perform a wakeup for this cluster. False: FrSm shall never perform a wakeup for this FlexRay cluster.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
FrSm	FrSm/FrSmCluster	
BSW Parameter		**BSW Type**
FrSmMainFunctionCycleTime		FloatParamDef
BSW Description		
This parameter defines the cycle time of the periodic calling of FrSm main function.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
FrSm	FrSm/FrSmCluster	
BSW Parameter		**BSW Type**

FrSmNetworkHandleRef			SymbolicNameReferenceParamDef
BSW Description			
Reference to the unique handle to identify one certain FlexRay network correspond to one of the network handles of the ComM configuration.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	**BSW Context**	
FrSm	FrSm/FrSmCluster	
BSW Parameter		**BSW Type**
FrSmNumWakeupPatterns		IntegerParamDef
BSW Description		
Maximum number of Wakeup Patterns the node may send before going to FRSM_STARTUP.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
FrSm	FrSm/FrSmCluster	
BSW Parameter		**BSW Type**
FrSmStartupRepetitions		IntegerParamDef
BSW Description		
The number of times an ECU may repeat the startup procedure for a FlexRay cluster.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
FrSm	FrSm/FrSmCluster	
BSW Parameter		**BSW Type**
FrSmStartupRepetitionsWithWakeup		IntegerParamDef
BSW Description		
The number of times an ECU may repeat the startup procedure including a wakeup for a FlexRay cluster.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
FrSm	FrSm/FrSmCluster	
BSW Parameter		**BSW Type**
FrSmTrcvStdbyDelay		FloatParamDef
BSW Description		
The duration of timer t_TrcvStdbyDelay in seconds. The granularity of this parameter shall be restricted to full FlexRay cycles (FrIfGdCycle). The transceiver status setting to STANDBY shall be delayed by this value. A value of 0 shall imply that the timer is not used.		
M2 Template	**M2 Description**	
SystemTemplate	The duration of timer t_TrcvStdbyDelay in seconds. The granularity of this parameter shall be restricted to full FlexRay cycles (cycle). The transceiver status setting to STANDBY shall be delayed by this value.	
M2 Parameter		
FlexrayTopology::FlexrayCluster.tranceiverStandbyDelay		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrSm	FrSm	
BSW Parameter		**BSW Type**
FrSmGeneral		ParamConfContainerDef
BSW Description		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
FrSm	FrSm/FrSmGeneral	
BSW Parameter		**BSW Type**
FrSmDevErrorDetect		BooleanParamDef
BSW Description		
Enables and disables the development error detection and notification mechanism.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
FrSm	FrSm/FrSmGeneral	
BSW Parameter		**BSW Type**
FrSmSyncLossErrorIndicationName		FunctionNameDef

BSW Description	
Name of <Cdd>_SyncLossErrorIndication function that shall be called on loss of synchronization. If this parameter is omitted no indication shall take place.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context
FrSm	FrSm/FrSmGeneral
BSW Parameter	**BSW Type**
FrSmVersionInfoApi	BooleanParamDef
BSW Description	
Enables and disables the version info API	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

9.10 Lin Interface Mapping

BSW Module	**BSW Context**	
LinIf	LinIf	
BSW Parameter		**BSW Type**
LinIfGeneral		ParamConfContainerDef
BSW Description		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
These parameters are global for the LIN interface, and will typically be config-ured late in the configuration process as they depend on the configuration total of other LIN parameters.		local

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGeneral	
BSW Parameter		**BSW Type**
LinIfDevErrorDetect		BooleanParamDef
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		BooleanParamDef
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**
LinIfNcOptionalRequestSupported		BooleanParamDef
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**	
LinIfTpSupported		BooleanParamDef	
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**
may be derived from other parameters, considering whether TP is being used on configured LIN channels.			local

BSW Module	**BSW Context**		
LinIf	LinIf/LinIfGeneral		
BSW Parameter		**BSW Type**	
LinIfVersionInfoApi		BooleanParamDef	
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		ParamConfContainerDef	
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**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig	
BSW Parameter		**BSW Type**
LinIfChannel		ParamConfContainerDef
BSW Description		
M2 Template	**M2 Description**	
SystemTemplate	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.	
M2 Parameter		
Fibex::FibexCore::CoreTopology::CommunicationConnector		
Mapping Rule		**Mapping Type**
In order to avoid the usage of unneeded resources, channels may only be configured if there is a need for them indicated by the existence of a CommunicationConnector belonging to the ECU's LINCommunicationController(s)		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfCddRef		ForeignReferenceParamDef
BSW Description		
Reference to the CDD module description. This parameter is only required when LinIfWakeupConfirmationUL, LinIfScheduleRequestConfirmationUL, and/or LinIfGotoSleepConfirmationUL is set to CDD.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfChannelId		IntegerParamDef
BSW Description		
Internal ID for the channel on LIN Interface level. This parameter shall map the NetworkHandleType to the physical LIN channel. 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	SymbolicNameReferenceParamDef
BSW Description	
Reference to the used channel in Lin. Replaces LINIF_CHANNEL_INDEX	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel
BSW Parameter	**BSW Type**
LinIfFrame	ParamConfContainerDef
BSW Description	
Generic container for all types of LIN frames.	
M2 Template	**M2 Description**
SystemTemplate	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	
Fibex::Fibex4Lin::LinCommunication::LinFrameTriggering	
Mapping Rule	**Mapping Type**
Each FrameTriggering aggregated by the PhysicalChannel representing the LIN channel forms a LinIfFrame.	full

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame
BSW Parameter	**BSW Type**
LinIfChecksumType	EnumerationParamDef
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.checksumType	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame
BSW Parameter	**BSW Type**
LinIfFixedFrameSdu	ParamConfContainerDef
BSW Description	
In case this is a fixed frame this is the SDU (response). This value should 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		ParamConfContainerDef	
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		IntegerParamDef	
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		IntegerParamDef	
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**
LinIfFrameName	StringParamDef
BSW Description	
Optional frame name used to cross-reference with a LDF	
M2 Template	**M2 Description**
SystemTemplate	Use <shortName> to generate a short name for the context element, which enables it to be ** .
M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::LinFrameTriggering	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame
BSW Parameter	**BSW Type**
LinIfFramePriority	IntegerParamDef
BSW Description	
Priority of an unconditional frame if used as a sporadic frame or in case of collision resolving of event triggered frames	
M2 Template	**M2 Description**
SystemTemplate	Collecting the frames that are substituted by the refering one. This reference is ordered. The order is used to describe the priority (Configuration parameter LinIfFramePriority). The first listed Substitution Frame has the highest priority.
M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::SubstitutionFrame.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
BSW Parameter	**BSW Type**
LinIfFrameType	EnumerationParamDef
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	
Fibex::Fibex4Lin::LinCommunication::SubstitutionFrame.substitutionType AND Fibex::Fibex4Lin::LinCommunication::AssignmentTimingType	
Mapping Rule	**Mapping Type**
RelativelyScheduledTiming, SubstitutionFrame, AssignNadTiming, AssignFrameIdTiming, UnassignFrameIdTiming, DataTiming can be used to derive this configuration parameter.	partial

BSW Module	**BSW Context**

<table>
<tr><td>LinIf</td><td colspan="3">LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">LinIfLength</td><td colspan="2">IntegerParamDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">Length of the LIN SDU in bytes.</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td>SystemTemplate</td><td colspan="3">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><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4">Fibex::FibexCore::CoreCommunication::Frame.frameLength</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>LinIf</td><td colspan="3">LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">LinIfPduDirection</td><td colspan="2">ChoiceContainerDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">Direction of the frame</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td>SystemTemplate</td><td colspan="3">The IPduTriggering describes on which channel the IPdu is transmitted.</td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4">Fibex::FibexCore::CoreCommunication::IPduTriggering</td></tr>
<tr><td colspan="3">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="3">The direction of the IPdu can be derived from the connection to the In- and Out-Ports of the Ecu.</td><td>full</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="3">BSW Context</td></tr>
<tr><td>LinIf</td><td colspan="3">LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">LinIfInternalPdu</td><td colspan="2">ParamConfContainerDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">Represents a Diagnostic or Configuration frame : no Message ID (no PduId).</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>LinIf</td><td colspan="3">LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">LinIfRxPdu</td><td colspan="2">ParamConfContainerDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">represents a received PDU/frame</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>
</table>

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection/LinIfRxPdu
BSW Parameter	**BSW Type**
LinIfRxIndicationUL	FunctionNameDef
BSW Description	
This parameter defines the name of the <User_RxIndication>. This parameter depends on the parameter LinIfUserRxIndicationUL. If LinIfUserRxIndicationUL equals PDUR, the name of the <User_RxIndication> is fixed. If LinIfUserRxIndicationUL equals CDD, the name of the <User_RxIndication> is selectable.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection/LinIfRxPdu
BSW Parameter	**BSW Type**
LinIfRxPduId	IntegerParamDef
BSW Description	
Identifier of the frame for the LIN Interface	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection/LinIfRxPdu
BSW Parameter	**BSW Type**
LinIfRxPduRef	ReferenceParamDef
BSW Description	
Reference to the PDU that is received in this frame.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection/LinIfRxPdu
BSW Parameter	**BSW Type**
LinIfUserRxIndicationUL	EnumerationParamDef

BSW Description	
This parameter defines the upper layer (UL) module to which the indication of the successfully received LINRXPDUID has to be routed via <User_RxIndication>. This <User_RxIndication> has to be invoked when the indication of the configured LINRXPDUID will be received by a Rx indication event from the LIN Driver module.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection
BSW Parameter	**BSW Type**
LinIfSlaveToSlavePdu	ParamConfContainerDef
BSW Description	
represents a slave-to-slave PDU/frame. Master does only send the header but doesn't receive the response.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection
BSW Parameter	**BSW Type**
LinIfTxPdu	ParamConfContainerDef
BSW Description	
represents a transmitted PDU/frame	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection/LinIfTxPdu
BSW Parameter	**BSW Type**
LinIfTxConfirmationUL	FunctionNameDef
BSW Description	
This parameter defines the name of the <User_TxConfirmation>. This parameter depends on the parameter LinIfUserTxUL. If LinIfUserTxUL equals PDUR, the name of the <User_TxConfirmation> is fixed. If LinIfUserTxUL equals CDD, the name of the <User_TxConfirmation> is selectable.	
M2 Template	**M2 Description**
M2 Parameter	

Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection/LinIfTxPdu	
BSW Parameter		**BSW Type**
LinIfTxPduId		IntegerParamDef
BSW Description		
Identifier of the frame for the upper layer. This id is only relevant for sporadic frames.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection/LinIfTxPdu	
BSW Parameter		**BSW Type**
LinIfTxPduRef		ReferenceParamDef
BSW Description		
Reference to the PDU that is transmitted in this frame.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection/LinIfTxPdu	
BSW Parameter		**BSW Type**
LinIfTxTriggerTransmitUL		FunctionNameDef
BSW Description		
This parameter defines the name of the <User_TriggerTransmit>. This parameter depends on the parameter LinIfUserTxUL. If LinIfUserTxUL equals PDUR, the name of the <User_TriggerTransmit> is fixed. If LinIfUserTxUL equals CDD, the name of the <User_TriggerTransmit> is selectable.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection/LinIfTxPdu

BSW Parameter	**BSW Type**
LinIfUserTxUL	EnumerationParamDef
BSW Description	
This parameter defines the upper layer (UL) module to which the trigger of the transmitted LinTxPdu (via the <User_TriggerTransmit>) or the confirmation of the successfully transmitted LinTxPdu has to be routed (via the <User_TxConfirmation>).	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame
BSW Parameter	**BSW Type**
LinIfPid	IntegerParamDef
BSW Description	
Protected ID of the LIN frame. There is no reason to calculate the Parity in run-time.	
M2 Template	**M2 Description**
SystemTemplate	To describe a frames identifier on the communication system, usualy with a fixed identifierValue.
M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::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	ParamConfContainerDef
BSW Description	
List of unconditional Frames that can be sent in an event-triggered Frame or a sporadic Frame slot.	
M2 Template	**M2 Description**
SystemTemplate	A LIN specific extension of the common FRAME to enable the usual frame handling of a placeholder frame that is substituted at runtime. Substitution frame must not declare signal instances nor multiplexers.
M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::SubstitutionFrame	
Mapping Rule	**Mapping Type**
Create container if SubstitutionFrame is defined	full

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfSubstitutionFrames
BSW Parameter	**BSW Type**
LinIfSubstitutionFrameRef	ReferenceParamDef
BSW Description	
Reference to an unconditional Frame that can be sent in an event-triggered Frame or a sporadic Frame slot.	

M2 Template	M2 Description	
SystemTemplate	Collecting the frames that are substituted by the refering one. This reference is ordered. The order is used to describe the priority (Configuration parameter LinIfFramePriority). The first listed Substitution Frame has the highest priority.	
M2 Parameter		
Fibex::Fibex4Lin::LinCommunication::SubstitutionFrame.substitutedFrame		
Mapping Rule		**Mapping Type**
Create reference to frames that are referenced by the SubstitutionFrame		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame	
BSW Parameter		**BSW Type**
LinIfTxTargetPduId		IntegerParamDef
BSW Description		
Identifier of the frame for the upper layer		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfGotoSleepConfirmationUL		EnumerationParamDef
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**

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfMaster		ParamConfContainerDef
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**	
SystemTemplate	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		
Fibex::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		FloatParamDef
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**	
SystemTemplate	timeBaseJitter is a mandatory attribute for the master and not used for slaves. The jitter shall be specified AUTOSAR conform in seconds.	
M2 Parameter		
Fibex::Fibex4Lin::LinTopology::LinMaster.timeBaseJitter		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfScheduleRequestConfirmationUL		EnumerationParamDef
BSW Description		
This parameter defines the upper layer (UL) module to which the confirmation of the successfully performed schedule table change shall be sent.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfScheduleRequestQueueLength		IntegerParamDef
BSW Description		
Number of schedule requests the schedule table manager can handle for this channel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
highly depending on how many upper layer modules are issuing LIN requests (diagnosis, initial configuration). Note that LIN schedule table switching is not supported by the System Template in Release 3.x.		local

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel

BSW Parameter			BSW Type
LinIfScheduleTable			ParamConfContainerDef
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**		
SystemTemplate	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**
1:1 mapping			full

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable		
BSW Parameter			**BSW Type**
LinIfEntry			ParamConfContainerDef
BSW Description			
Describes an entry in the schedule table (also known as Frame Slot).			
M2 Template	**M2 Description**		
SystemTemplate	Specification of a sending behavior where the transmission order is predefined, e.g. used on LIN buses		
M2 Parameter			
Fibex::Fibex4Lin::LinCommunication::RelativelyScheduledTiming			
Mapping Rule			**Mapping Type**
An existing RelativelyScheduledTiming is the reason for a LinIfEntry. RelativelyScheduledTiming.scheduleTableName decides to which schedule ta- ble the LinIfEntry belongs.			full

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable/LinIfEntry		
BSW Parameter			**BSW Type**
LinIfDelay			FloatParamDef
BSW Description			
Delay to next frame in schedule table in [s]			
M2 Template	**M2 Description**		
SystemTemplate	Relative delay between this frame and the start of the successor frame in the schedule table in seconds.		
M2 Parameter			
Fibex::Fibex4Lin::LinCommunication::RelativelyScheduledTiming.delay			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable/LinIfEntry		
BSW Parameter			**BSW Type**
LinIfEntryIndex			IntegerParamDef
BSW Description			

Position of the Frame Entry in the Schedule Table.		
M2 Template	**M2 Description**	
SystemTemplate	Relative position of the frame described by this timing in the schedule table	
M2 Parameter		
Fibex::Fibex4Lin::LinCommunication::RelativelyScheduledTiming.positionInTable		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable/LinIfEntry	
BSW Parameter		**BSW Type**
LinIfFrameRef		ReferenceParamDef
BSW Description		
Reference to the frames that belong to this schedule table entry.		
M2 Template	**M2 Description**	
SystemTemplate	Specification of a sending behavior where the transmission order is predefined, e.g. used on LIN buses	
M2 Parameter		
Fibex::Fibex4Lin::LinCommunication::RelativelyScheduledTiming		
Mapping Rule		**Mapping Type**
Reference to the frame that contains the RelativelyScheduledTiming with the schedule table position.		full

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable	
BSW Parameter		**BSW Type**
LinIfResumePosition		EnumerationParamDef
BSW Description		
Defines, where a schedule table shall be proceeded in case it has been interrupted by a RUN_ONCE table or MRF/SRF.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable	
BSW Parameter		**BSW Type**
LinIfRunMode		EnumerationParamDef
BSW Description		
The schedule table can be executed in two different modes.		
M2 Template	**M2 Description**	
SystemTemplate	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**
LinIfSchedulePriority	IntegerParamDef
BSW Description	
Priority of the schedule table. The priority is used in the schedule table manager. The RUN_ONCE run mode schedules shall not have equal priority. 0 Reserved for NULL_SCHEDULE 1..254 Only for RUN_ONCE 255 Only RUN_CONTINUOUS	
M2 Template	**M2 Description**
SystemTemplate	Priority of the schedule table. The priority is used in the schedule table manager. The RUN_ONCE run mode schedules shall not have equal priority. Priority 0 is reserved for the NULL_SCHEDULE. Priority 255 is reserved for the RUN_CONTINUOUS run mode.
M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::LinScheduleTable.priority	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable
BSW Parameter	**BSW Type**
LinIfScheduleTableIndex	IntegerParamDef
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	StringParamDef
BSW Description	
Optional schedule name used to cross-reference with a LDF. This parameter shall always be accompanied by LINIF_SCHEDULE_INDEX	
M2 Template	**M2 Description**
SystemTemplate	
M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::LinScheduleTable.shortName	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel

BSW Parameter		BSW Type
LinIfSlave		ParamConfContainerDef
BSW Description		
The Node attributes of the Slaves are provided with these parameter.		
M2 Template	**M2 Description**	
SystemTemplate	Describing the properties of the refering ecu as a LIN slave.	
M2 Parameter		
Fibex::Fibex4Lin::LinTopology::LinSlave		
Mapping Rule		**Mapping Type**
Container must be created for each existing LinSlave in the System Description.		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave	
BSW Parameter		**BSW Type**
LinIfConfiguredNad		IntegerParamDef
BSW Description		
Definition of the initial node address		
M2 Template	**M2 Description**	
SystemTemplate	To distinguish LIN slaves that are used twice or more within the same cluster.	
M2 Parameter		
Fibex::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		IntegerParamDef
BSW Description		
LIN function ID		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave	
BSW Parameter		**BSW Type**
LinIfNodeComposition		ParamConfContainerDef
BSW Description		
Generic container that describes the node composition		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave/LinIfNodeComposition	
BSW Parameter		**BSW Type**
LinIfNodeName		StringParamDef
BSW Description		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
Fibex::Fibex4Lin::LinTopology::LinSlave.shortName		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave	
BSW Parameter		**BSW Type**
LinIfProtocolVersion		StringParamDef
BSW Description		
Defines the LIN Protocol version which is used by the slave.		
M2 Template	**M2 Description**	
SystemTemplate	Version specifier for a communication protocol.	
M2 Parameter		
Fibex::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		IntegerParamDef
BSW Description		
LIN Supplier ID		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave	
BSW Parameter		**BSW Type**
LinIfVariant		IntegerParamDef
BSW Description		
Specifies the Variant ID		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

	local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfWakeupConfirmationUL		EnumerationParamDef
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		FloatParamDef
BSW Description		
Defines the interval of calls to main functions in seconds.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

9.11 Lin Driver Mapping

BSW Module	BSW Context	
Lin	Lin	
BSW Parameter		**BSW Type**
LinGeneral		ParamConfContainerDef
BSW Description		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Lin	Lin/LinGeneral	
BSW Parameter		**BSW Type**
LinDevErrorDetect		BooleanParamDef
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		IntegerParamDef
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		IntegerParamDef
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		BooleanParamDef
BSW Description		
Switches the Lin_GetVersionInfo function ON or OFF.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Lin	Lin	
BSW Parameter		**BSW Type**
LinGlobalConfig		ParamConfContainerDef
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		ParamConfContainerDef
BSW Description		
This container contains the configuration (parameters) of the LIN Controller(s).		
M2 Template	**M2 Description**	
SystemTemplate	The connection between the referencing ECU and the referenced channel via the referenced controller.	
M2 Parameter		
Fibex::FibexCore::CoreTopology::CommunicationConnector		
Mapping Rule		**Mapping Type**
Create container for each CommunicationConnector that is connencted to a LI N Physical Channel.		full

BSW Module	**BSW Context**
Lin	Lin/LinGlobalConfig/LinChannel
BSW Parameter	**BSW Type**
LinChannelBaudRate	IntegerParamDef
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	SymbolicNameReferenceParamDef
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**
Lin	Lin/LinGlobalConfig/LinChannel
BSW Parameter	**BSW Type**
LinChannelId	IntegerParamDef
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	BooleanParamDef
BSW Description	

Specifies if the LIN hardware channel supports wake up functionality		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
ECU-local parameter, dependent on communication peripheral capabilities. Typically pre-configured for the supported HW by BSW-vendor in VSMD.		local

BSW Module	**BSW Context**		
Lin	Lin/LinGlobalConfig/LinChannel		
BSW Parameter		**BSW Type**	
LinClockRef		ReferenceParamDef	
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

9.12 LinTP Mapping

BSW Module	**BSW Context**		
LinTp	LinTp		
BSW Parameter		**BSW Type**	
LinTpGeneral		ParamConfContainerDef	
BSW Description			
Container that holds all LIN transport protocol general parameters.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
LinTp	LinTp/LinTpGeneral		
BSW Parameter		**BSW Type**	
LinTpVersionInfoApi		BooleanParamDef	
BSW Description			
Switches the LinTp_GetVersionInfo function ON or OFF.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
LinTp	LinTp		
BSW Parameter		**BSW Type**	
LinTpGlobalConfig		ParamConfContainerDef	
BSW Description			
This container contains the global configuration parameter of the LinTp. It 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**	
LinTp	LinTp/LinTpGlobalConfig	
BSW Parameter		**BSW Type**
LinTpChannelConfig		ParamConfContainerDef
BSW Description		
This container contains the channel specific configuration parameter of LinTp.		
M2 Template	**M2 Description**	

SystemTemplate	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. The LinTp module routes a Pdu through the connection channel
M2 Parameter	
TransportProtocols::LinTpChannel	
Mapping Rule	**Mapping Type**
Create container if LinTpChannel exists in the EcuExtract.	full

BSW Module	**BSW Context**	
LinTp	LinTp/LinTpGlobalConfig/LinTpChannelConfig	
BSW Parameter		**BSW Type**
LinTpChannelRef		SymbolicNameReferenceParamDef
BSW Description		
Index of the channel this LinTp channel belongs to.		
M2 Template	**M2 Description**	
SystemTemplate	Optional configuration of Transport Protocol channels.	
M2 Parameter		
CoreTopology::PhysicalChannel.tpChannel		
Mapping Rule		**Mapping Type**
This reference shall be derived from the aggregation between the LIN Physical Channel and LinTpChannel.		full

BSW Module	**BSW Context**	
LinTp	LinTp/LinTpGlobalConfig/LinTpChannelConfig	
BSW Parameter		**BSW Type**
LinTpDropNotRequestedNad		BooleanParamDef
BSW Description		
Configures if TP Frames of not requested LIN-Slaves are dropped or not. TRUE: Drop TP frames of not requested LIN-Slaves. FALSE: Keep TP frames of not requested LIN-Slaves.		
M2 Template	**M2 Description**	
SystemTemplate	Configures if TP Frames of not requested LIN-Slaves are dropped or not.	
M2 Parameter		
TransportProtocols::LinTpChannel.dropNotRequestedNad		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
LinTp	LinTp/LinTpGlobalConfig/LinTpChannelConfig	
BSW Parameter		**BSW Type**
LinTpScheduleChangeDiag		BooleanParamDef
BSW Description		
Enables or disables the call of BswM_LinTp_RequestMode() to diagnostic request/response schedule.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig	
BSW Parameter		**BSW Type**
LinTpMaxNumberOfRespPendingFrames		IntegerParamDef
BSW Description		
Configures the maximum number of allowed response pending frames.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig	
BSW Parameter		**BSW Type**
LinTpNumberOfRxNSdu		IntegerParamDef
BSW Description		
Number of transport protocol messages that can be received for all channels this node is connected to. Please note that this parameter is deprecated and will be removed in the next major version. The value of this parameter can be calculated from the multiplicity of the container LinTpRxNSdu.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig	
BSW Parameter		**BSW Type**
LinTpNumberOfTxNSdu		IntegerParamDef
BSW Description		
Number of transport protocol messages that can be transmitted for all channels this node is connected to. Please note that this parameter is deprecated and will be removed in the next major version. The value of this parameter can be calculated from the multiplicity of the container LinTpTxNSdu.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig	
BSW Parameter		**BSW Type**
LinTpP2Max		FloatParamDef
BSW Description		
P2 Timeout when a response pending frame is expected in seconds.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig	
BSW Parameter		**BSW Type**
LinTpP2Timing		FloatParamDef
BSW Description		
Definition of the P2 timeout observation parameter in seconds.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig	
BSW Parameter		**BSW Type**
LinTpRxNSdu		ParamConfContainerDef
BSW Description		
For each received N-SDU on any channel the node is connected to.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the IPdu that is segmented by the Transport Protocol.	
M2 Parameter		
TransportProtocols::LinTpChannel.linTpNSdu		
Mapping Rule		**Mapping Type**
The direction of the IPdu (Tx or Rx) can be derived from the related Triggering elements.		full

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig/LinTpRxNSdu	
BSW Parameter		**BSW Type**
LinTpDl		IntegerParamDef
BSW Description		

Data Length Code of this RxNsdu. In case of variable length message, this value indicates the minimum data length. Range of minimum length is 1 to 4095. Note that this is not relevant for Tx. The reason for this is to have identical structures for Tx and Rx.	
M2 Template	**M2 Description**
SystemTemplate	The size of the IPDU in bits.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::IPdu.length	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
LinTp	LinTp/LinTpGlobalConfig/LinTpRxNSdu
BSW Parameter	**BSW Type**
LinTpRxNSduChannelRef	SymbolicNameReferenceParamDef
BSW Description	
Index of the channel this N-SDU belongs to.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
LinTp	LinTp/LinTpGlobalConfig/LinTpRxNSdu
BSW Parameter	**BSW Type**
LinTpRxNSduId	IntegerParamDef
BSW Description	
The identifier of the Transport Protocol message. This ID will be the one that is communicated with upper layers.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
LinTp	LinTp/LinTpGlobalConfig/LinTpRxNSdu
BSW Parameter	**BSW Type**
LinTpRxNSduNad	IntegerParamDef
BSW Description	
A N-SDU transported on LIN is identified using the NAD for the specific slave.	
M2 Template	**M2 Description**
M2 Parameter	

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
LinTp	LinTp/LinTpGlobalConfig/LinTpRxNSdu	
BSW Parameter		**BSW Type**
LinTpRxNSduPduRef		ReferenceParamDef
BSW Description		
Reference to the global PDU		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
LinTp	LinTp/LinTpGlobalConfig	
BSW Parameter		**BSW Type**
LinTpTxNSdu		ParamConfContainerDef
BSW Description		
For each transmitted N-SDU on any channel the node is connected to.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the IPdu that is segmented by the Transport Protocol.	
M2 Parameter		
TransportProtocols::LinTpChannel.linTpNSdu		
Mapping Rule		**Mapping Type**
The direction of the IPdu (Tx or Rx) can be derived from the related Triggering elements.		full

BSW Module	**BSW Context**	
LinTp	LinTp/LinTpGlobalConfig/LinTpTxNSdu	
BSW Parameter		**BSW Type**
LinTpTxNSduChannelRef		SymbolicNameReferenceParamDef
BSW Description		
Index of the channel this N-SDU belongs to.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
LinTp	LinTp/LinTpGlobalConfig/LinTpTxNSdu	
BSW Parameter		**BSW Type**
LinTpTxNSduId		IntegerParamDef
BSW Description		

The identifier of the Transport Protocol message. This ID will be the one that is communicated with upper layers.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
LinTp	LinTp/LinTpGlobalConfig/LinTpTxNSdu	
BSW Parameter		**BSW Type**
LinTpTxNSduNad		IntegerParamDef
BSW Description		
A N-SDU transported on LIN is identified using the NAD for the specific slave.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
LinTp	LinTp/LinTpGlobalConfig/LinTpTxNSdu	
BSW Parameter		**BSW Type**
LinTpTxNSduPduRef		ReferenceParamDef
BSW Description		
Reference to the global PDU		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

9.13 Can Interface Mapping

BSW Module	BSW Context		
CanIf	CanIf		
BSW Parameter		**BSW Type**	
CanIfControllerConfig		ParamConfContainerDef	
BSW Description			
This container contains the configuration (parameters) of all addressed CAN controllers by each underlying CAN driver.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfControllerConfig		
BSW Parameter		**BSW Type**	
CanIfCtrlCanCtrlRef		SymbolicNameReferenceParamDef	
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/CanIfControllerConfig		
BSW Parameter		**BSW Type**	
CanIfCtrlId		IntegerParamDef	
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**

BSW Module	BSW Context		
CanIf	CanIf/CanIfControllerConfig		
BSW Parameter		**BSW Type**	
CanIfCtrlWakeupSupport		BooleanParamDef	
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/CanIfControllerConfig		
BSW Parameter		**BSW Type**	
CanIfDriverNameRef		ReferenceParamDef	
BSW Description			
Refers to the CAN Driver Name to which the controller belongs to. This parameter refers to CanIf-DriverConfig container.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf		
BSW Parameter		**BSW Type**	
CanIfDispatchConfig		ParamConfContainerDef	
BSW Description			
Callout functions with respect to the upper layers. This callout functions defined in this container are common to all configured underlying CAN Drivers / CAN Transceiver Drivers.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
CanIf	CanIf/CanIfDispatchConfig	
BSW Parameter		**BSW Type**
CanIfDispatchUserCheckTrcvWakeFlagIndicationName		FunctionNameDef
BSW Description		

This parameter defines the name of <User_CheckTrcvWakeFlagIndication>. This parameter depends on the parameter CANIF_DISPATCH_USERCHECKTRCVWAKEFLAGINDICATION_UL. If CANIF_DISPATCH_USERCHECKTRCVWAKEFLAGINDICATION_UL equals CAN_SM the name of <User_CheckTrcvWakeFlagIndication> is fixed. If CANIF_DISPATCH_USERCHECKTRCVWAKEFLAGINDICATION_UL equals CDD, the name of <User_CheckTrcvWakeFlagIndication> is selectable. minLength: 1 maxLength: 32	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
CanIf	CanIf/CanIfDispatchConfig
BSW Parameter	**BSW Type**
CanIfDispatchUserCheckTrcvWakeFlagIndicationUL	EnumerationParamDef
BSW Description	
This parameter defines the upper layer (UL) module to which the CanIf_CheckTrcvWakeFlagIndication has to be routed via <User_CheckTrcvWakeFlagIndication>. If no UL module is configured, no upper layer callback function will be called.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
CanIf	CanIf/CanIfDispatchConfig
BSW Parameter	**BSW Type**
CanIfDispatchUserClearTrcvWufFlagIndicationName	FunctionNameDef
BSW Description	
This parameter defines the name of <User_ClearTrcvWufFlagIndication>. This parameter depends on the parameter CANIF_DISPATCH_USERCLEARTRCVWUFFLAGINDICATION_UL. If CANIF_DISPATCH_USERCLEARTRCVWUFFLAGINDICATION_UL equals CAN_SM the name of <User_ClearTrcvWufFlagIndication> is fixed. If CANIF_DISPATCH_USERCLEARTRCVWUFFLAGINDICATION_UL equals CDD, the name of <User_ClearTrcvWufFlagIndication> is selectable. minLength: 1 maxLength: 32	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
CanIf	CanIf/CanIfDispatchConfig
BSW Parameter	**BSW Type**
CanIfDispatchUserClearTrcvWufFlagIndicationUL	EnumerationParamDef
BSW Description	
This parameter defines the upper layer (UL) module to which the CanIf_ClearTrcvWufFlagIndication has to be routed via <User_ClearTrcvWufFlagIndication>. If no UL module is configured, no upper layer callback function will be called.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
CanIf	CanIf/CanIfDispatchConfig
BSW Parameter	**BSW Type**
CanIfDispatchUserConfirmPnAvailabilityName	FunctionNameDef
BSW Description	
This parameter defines the name of <User_ConfirmPnAvailability>. This parameter depends on the parameter CANIF_DISPATCH_USERCONFIRMPNAVAILABILITY_UL. If CANIF_DISPATCH_USERCONFIRMPNAVAILABILITY_UL equals CAN_SM the name of <User_ConfirmPnAvailability> is fixed. If CANIF_DISPATCH_USERCONFIRMPNAVAILABILITY_UL equals CDD, the name of <User_ConfirmPnAvailability> is selectable. minLength: 1 maxLength: 32	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
CanIf	CanIf/CanIfDispatchConfig
BSW Parameter	**BSW Type**
CanIfDispatchUserConfirmPnAvailabilityUL	EnumerationParamDef
BSW Description	
This parameter defines the upper layer (UL) module to which the CanIf_ConfirmPnAvailability has to be routed via <User_ConfirmPnAvailability>. If no UL module is configured, no upper layer callback function will be called.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context
CanIf	CanIf/CanIfDispatchConfig
BSW Parameter	**BSW Type**
CanIfDispatchUserCtrlBusOffName	FunctionNameDef
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/CanIfDispatchConfig
BSW Parameter	**BSW Type**
CanIfDispatchUserCtrlBusOffUL	EnumerationParamDef
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/CanIfDispatchConfig
BSW Parameter	**BSW Type**
CanIfDispatchUserTrcvModeIndicationName	FunctionNameDef
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. minLength: 1 maxLength: 32	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context

CanIf	CanIf/CanIfDispatchConfig	
BSW Parameter		**BSW Type**
CanIfDispatchUserTrcvModeIndicationUL		EnumerationParamDef
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**

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfDispatchConfig	
BSW Parameter		**BSW Type**
CanIfDispatchUserValidateWakeupEventName		FunctionNameDef
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/CanIfDispatchConfig	
BSW Parameter		**BSW Type**
CanIfDispatchUserValidateWakeupEventUL		EnumerationParamDef
BSW Description		
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.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**
CanIf	CanIf

BSW Parameter		BSW Type
CanIfDriverConfig		ParamConfContainerDef
BSW Description		
Configuration parameters for all the underlying CAN drivers are aggregated under this container.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
CanIf	CanIf/CanIfDriverConfig		
BSW Parameter		**BSW Type**	
CanIfBusoffNotification		BooleanParamDef	
BSW Description			
Selects whether BusOff indication notification is supported. True: Enabled False: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfDriverConfig		
BSW Parameter		**BSW Type**	
CanIfDriverNameRef		ReferenceParamDef	
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/CanIfDriverConfig	
BSW Parameter		**BSW Type**
CanIfInitHohConfigRef		ReferenceParamDef
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/CanIfDriverConfig		
BSW Parameter		**BSW Type**	
CanIfReceiveIndication		BooleanParamDef	
BSW Description			
Selects whether receive indication notification is supported. True: Enabled False: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfDriverConfig		
BSW Parameter		**BSW Type**	
CanIfTransmitCancellation		BooleanParamDef	
BSW Description			
Selects whether transmit cancellation is supported. True: Enabled False: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfDriverConfig		
BSW Parameter		**BSW Type**	
CanIfTxConfirmation		BooleanParamDef	
BSW Description			
Selects whether transmit confirmation notification is supported.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfDriverConfig
BSW Parameter	**BSW Type**
CanIfWakeupNotification	BooleanParamDef
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**
CanIf	CanIf
BSW Parameter	**BSW Type**
CanIfInitConfiguration	ParamConfContainerDef
BSW Description	
This container contains the init parameters of the CAN Interface.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfInitConfiguration
BSW Parameter	**BSW Type**
CanIfConfigSet	StringParamDef
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/CanIfInitConfiguration
BSW Parameter	**BSW Type**
CanIfInitControllerConfig	ParamConfContainerDef
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/CanIfInitConfiguration/CanIfInitControllerConfig		
BSW Parameter		**BSW Type**	
CanIfControllerRefConfigSet		SymbolicNameReferenceParamDef	
BSW Description			
References the corresponding CAN Controller configuration setup of the corresponding CAN Driver.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitConfiguration		
BSW Parameter		**BSW Type**	
CanIfInitControllerRef		ReferenceParamDef	
BSW Description			
Reference to the Init Controller Configuration.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitConfiguration		
BSW Parameter		**BSW Type**	
CanIfInitHohConfig		ParamConfContainerDef	
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/CanIfInitConfiguration/CanIfInitHohConfig

BSW Parameter	**BSW Type**
CanIfHrhConfig	ParamConfContainerDef
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/CanIfInitConfiguration/CanIfInitHohConfig/CanIfHrhConfig
BSW Parameter	**BSW Type**
CanIfCanControllerHrhIdRef	ReferenceParamDef
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/CanIfInitConfiguration/CanIfInitHohConfig/CanIfHrhConfig
BSW Parameter	**BSW Type**
CanIfHrhIdSymRef	SymbolicNameReferenceParamDef
BSW Description	
The parameter refers to a particular HRH object in the CAN Driver Module configuration. The HRH id is unique in a given CAN Driver. The HRH Ids are defined in the CAN Driver Module and hence it is derived from CAN Driver Configuration.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfInitConfiguration/CanIfInitHohConfig/CanIfHrhConfig
BSW Parameter	**BSW Type**
CanIfHrhRangeConfig	ParamConfContainerDef
BSW Description	
Defines the parameters required for configuraing 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/CanIfInitConfiguration/CanIfInitHohConfig/CanIfHrhConfig/CanIfHrhRangeConfig	
BSW Parameter		**BSW Type**
CanIfRxPduLowerCanId		IntegerParamDef
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**	
SystemTemplate		
M2 Parameter		
Fibex::Fibex4Can::CanCommunication::RxIdentifierRange.lowerCanId		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfiguration/CanIfInitHohConfig/CanIfHrhConfig/CanIfHrhRangeConfig	
BSW Parameter		**BSW Type**
CanIfRxPduRangeCanIdType		EnumerationParamDef
BSW Description		
Specifies whether a configured Range of CAN Ids shall only consider standard CAN Ids or extended CAN Ids.		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
Fibex::Fibex4Can::CanCommunication::CanFrameTriggering.canAddressingMode		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfiguration/CanIfInitHohConfig/CanIfHrhConfig/CanIfHrhRangeConfig	
BSW Parameter		**BSW Type**
CanIfRxPduUpperCanId		IntegerParamDef
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**	
SystemTemplate		
M2 Parameter		
Fibex::Fibex4Can::CanCommunication::RxIdentifierRange.upperCanId		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context

CanIf	CanIf/CanIfInitConfiguration/CanIfInitHohConfig/CanIfHrhConfig		
BSW Parameter		**BSW Type**	
CanIfHrhType		EnumerationParamDef	
BSW Description			
Defines the HRH type i.e, whether its a BasicCan or FullCan. If BasicCan is configured, software filtering is enabled.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitConfiguration/CanIfInitHohConfig/CanIfHrhConfig		
BSW Parameter		**BSW Type**	
CanIfSoftwareFilterHrh		BooleanParamDef	
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/CanIfInitConfiguration/CanIfInitHohConfig		
BSW Parameter		**BSW Type**	
CanIfHthConfig		ParamConfContainerDef	
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/CanIfInitConfiguration/CanIfInitHohConfig/CanIfHthConfig	
BSW Parameter		**BSW Type**
CanIfCanControllerIdRef		ReferenceParamDef
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/CanIfInitConfiguration/CanIfInitHohConfig/CanIfHthConfig	
BSW Parameter		**BSW Type**
CanIfHthIdSymRef		SymbolicNameReferenceParamDef
BSW Description		
The parameter refers to a particular HTH object in the CAN Driver Module configuration. The HTH id is unique in a given CAN Driver. The HTH Ids are defined in the CAN Driver Module and hence it is derived from CAN Driver Configuration.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitConfiguration/CanIfInitHohConfig/CanIfHthConfig	
BSW Parameter		**BSW Type**
CanIfHthType		EnumerationParamDef
BSW Description		
Transmission method of the corresponding HTH.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitConfiguration/CanIfInitHohConfig	
BSW Parameter		**BSW Type**
CanIfRefConfigSet		ReferenceParamDef
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/CanIfInitConfiguration		
BSW Parameter		**BSW Type**	
CanIfNumberOfCanRxPduIds		IntegerParamDef	
BSW Description			
Total number of CanRxPduIds to be handled. Range: 0..max number of defined CanRxPduIds			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitConfiguration		
BSW Parameter		**BSW Type**	
CanIfNumberOfCanTXPduIds		IntegerParamDef	
BSW Description			
Total number of CanTxPduIds to be handled. Range: 0..max number of defined CanTxPduIds			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitConfiguration		
BSW Parameter		**BSW Type**	
CanIfNumberOfDynamicCanTXPduIds		IntegerParamDef	
BSW Description			
Total number of dynamic CanTxPduIds to be handled. Range: 0..max. nember of defined CanTxPduIds			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitConfiguration	
BSW Parameter		**BSW Type**
CanIfRxPduConfig		ParamConfContainerDef
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/CanIfInitConfiguration/CanIfRxPduConfig
BSW Parameter	**BSW Type**
CanIfCanRxPduCanId	IntegerParamDef
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/CanIfInitConfiguration/CanIfRxPduConfig
BSW Parameter	**BSW Type**
CanIfCanRxPduDlc	IntegerParamDef
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/CanIfInitConfiguration/CanIfRxPduConfig
BSW Parameter	**BSW Type**
CanIfCanRxPduHrhRef	ReferenceParamDef
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/CanIfInitConfiguration/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
CanIfCanRxPduId		IntegerParamDef
BSW Description		
ECU wide unique, symbolic handle for receive CAN L-PDU. The CanRxPduId 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/CanIfInitConfiguration/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
CanIfReadRxPduData		BooleanParamDef
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/CanIfInitConfiguration/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
CanIfReadRxPduNotifyStatus		BooleanParamDef
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/CanIfInitConfiguration/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
CanIfRxPduCanIdRange		ParamConfContainerDef
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/CanIfInitConfiguration/CanIfRxPduConfig/CanIfRxPduCanIdRange	
BSW Parameter		**BSW Type**
CanIfRxPduCanIdRangeLowerCanId		IntegerParamDef
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/CanIfInitConfiguration/CanIfRxPduConfig/CanIfRxPduCanIdRange	
BSW Parameter		**BSW Type**
CanIfRxPduCanIdRangeUpperCanId		IntegerParamDef
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/CanIfInitConfiguration/CanIfRxPduConfig	
BSW Parameter		**BSW Type**

CanIfRxPduIdCanIdType		EnumerationParamDef	
BSW Description			
CAN Identtifier 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/CanIfInitConfiguration/CanIfRxPduConfig		
BSW Parameter		**BSW Type**	
CanIfRxUserType		EnumerationParamDef	
BSW Description			
This parameter defines the type of the receive indication call-outs called to the corresponding upper layer the used TargetRxPduId belongs to.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitConfiguration/CanIfRxPduConfig		
BSW Parameter		**BSW Type**	
CanIfUserRxIndication		FunctionNameDef	
BSW Description			
Name of target indication services to target upper layers (PduRouter, CanNm, CanTp and ComplexDeviceDrivers). If parameter is not present no call-out function is configured.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitConfiguration/CanIfRxPduConfig		
BSW Parameter		**BSW Type**	
PduIdRef		ReferenceParamDef	
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**

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfiguration	
BSW Parameter		**BSW Type**
CanIfTxPduConfig		ParamConfContainerDef
BSW Description		
This container contains the configuration (parameters) of each transmit CAN L-PDU. 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/CanIfInitConfiguration/CanIfTxPduConfig	
BSW Parameter		**BSW Type**
CanIfCanTxPduHthRef		ReferenceParamDef
BSW Description		
Handle, that defines the hardware object or the pool of hardware objects configured for transmission. The parameter refers HTH Id, to which the L-PDU belongs to.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfiguration/CanIfTxPduConfig	
BSW Parameter		**BSW Type**
CanIfCanTxPduId		IntegerParamDef
BSW Description		
ECU wide unique, symbolic handle for transmit CAN L-PDU. The CanIfCanTxPduId 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/CanIfInitConfiguration/CanIfTxPduConfig		
BSW Parameter		**BSW Type**	
CanIfCanTxPduIdCanId		IntegerParamDef	
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/CanIfInitConfiguration/CanIfTxPduConfig		
BSW Parameter		**BSW Type**	
CanIfCanTxPduIdDlc		IntegerParamDef	
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/CanIfInitConfiguration/CanIfTxPduConfig		
BSW Parameter		**BSW Type**	
CanIfCanTxPduType		EnumerationParamDef	
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/CanIfInitConfiguration/CanIfTxPduConfig	
BSW Parameter		**BSW Type**
CanIfReadTxPduNotifyStatus		BooleanParamDef
BSW Description		

<table>
<tr><td colspan="3">Enables and disables transmit confirmation for each transmit CAN L-PDU for reading its notification status.

True: Enabled
False: Disabled</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="3">BSW Context</th></tr>
<tr><td>CanIf</td><td colspan="3">CanIf/CanIfInitConfiguration/CanIfTxPduConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanIfTxPduIdCanIdType</td><td colspan="2">EnumerationParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">CAN Identifier of transmit CAN L-PDUs used by the CAN Driver for CAN L-PDU transmission.</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>CanIf</td><td colspan="3">CanIf/CanIfInitConfiguration/CanIfTxPduConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanIfTxPduPnFilterPdu</td><td colspan="2">BooleanParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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.</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>CanIf</td><td colspan="2">CanIf/CanIfInitConfiguration/CanIfTxPduConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanIfTxUserType</td><td>EnumerationParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This parameter defines the type of the transmit confirmation call-out called to the corresponding upper layer the used TargetTxPduId belongs to.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td></td><td colspan="2"></td></tr>
</table>

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfiguration/CanIfTxPduConfig	
BSW Parameter		**BSW Type**
CanIfUserTxConfirmation		FunctionNameDef
BSW Description		
Name of target confirmation services to target upper layers (PduR, CanNm, CanTp, Complex Device Driver). If parameter is not configured then no call-out function is provided by the upper layer for this Tx L-PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfiguration/CanIfTxPduConfig	
BSW Parameter		**BSW Type**
PduIdRef		ReferenceParamDef
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**

BSW Module	BSW Context	
CanIf	CanIf	
BSW Parameter		**BSW Type**
CanIfPrivateConfiguration		ParamConfContainerDef
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/CanIfPrivateConfiguration		
BSW Parameter		**BSW Type**	
CanIfDlcCheck		BooleanParamDef	
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/CanIfPrivateConfiguration		
BSW Parameter		**BSW Type**	
CanIfNumberOfTxBuffers		IntegerParamDef	
BSW Description			
Defined the number of L-PDU elements for the transmit buffering. The Tx L-PDU buffers shall be used to store an L-PDU once for each different L-PDU handle. Range: 0..max. number of Tx L-PDUs to be used. Default Value: NUMBER_OF_TX_PDUS			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfPrivateConfiguration		
BSW Parameter		**BSW Type**	
CanIfSoftwareFilterType		EnumerationParamDef	
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	
BSW Parameter		**BSW Type**

CanIfPublicConfiguration		ParamConfContainerDef
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/CanIfPublicConfiguration	
BSW Parameter		**BSW Type**
CanIfDevErrorDetect		BooleanParamDef
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/CanIfPublicConfiguration	
BSW Parameter		**BSW Type**
CanIfMultipleDriverSupport		BooleanParamDef
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/CanIfPublicConfiguration	
BSW Parameter		**BSW Type**
CanIfNumberOfCanHwUnits		IntegerParamDef
BSW Description		
Number of served CAN hardware units. Range: 1..max. number of underlying supported CAN Hardware units		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
CanIf	CanIf/CanIfPublicConfiguration		
BSW Parameter		**BSW Type**	
CanIfPublicCddHeaderFile		StringParamDef	
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/CanIfPublicConfiguration		
BSW Parameter		**BSW Type**	
CanIfPublicHandleTypeEnum		EnumerationParamDef	
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/CanIfPublicConfiguration		
BSW Parameter		**BSW Type**	
CanIfPublicPnSupport		BooleanParamDef	
BSW Description			
Selects support of Partial Networking features in CanIf. True: Enabled False: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfPublicConfiguration		
BSW Parameter		**BSW Type**	
CanIfPublicTxConfirmPollingSupport		BooleanParamDef	
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/CanIfPublicConfiguration		
BSW Parameter		**BSW Type**	
CanIfPublicWakeupCheckValidSupport		BooleanParamDef	
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/CanIfPublicConfiguration		
BSW Parameter		**BSW Type**	
CanIfReadRxPduDataApi		BooleanParamDef	
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/CanIfPublicConfiguration	
BSW Parameter		**BSW Type**
CanIfReadRxPduNotifyStatusApi		BooleanParamDef
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/CanIfPublicConfiguration	
BSW Parameter		**BSW Type**
CanIfReadTxPduNotifyStatusApi		BooleanParamDef
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/CanIfPublicConfiguration	
BSW Parameter		**BSW Type**
CanIfSetDynamicTxIdApi		BooleanParamDef
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/CanIfPublicConfiguration	
BSW Parameter		**BSW Type**
CanIfVersionInfoApi		BooleanParamDef
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	
BSW Parameter		**BSW Type**
CanIfTransceiverDrvConfig		ParamConfContainerDef
BSW Description		
This container contains the configuration (parameters) of all addressed CAN transceivers by each underlying CAN Transceiver Driver.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfTransceiverDrvConfig	
BSW Parameter		**BSW Type**
CanIfTrcvId		IntegerParamDef
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**

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfTransceiverDrvConfig	
BSW Parameter		**BSW Type**
CanIfTrcvIdRef		SymbolicNameReferenceParamDef
BSW Description		
Logical handle of the underlying CAN transceiver to be served by the CAN Interface.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfTransceiverDrvConfig	
BSW Parameter		**BSW Type**
CanIfTrcvWakeupSupport		BooleanParamDef
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

9.14 Can Driver Mapping

BSW Module	BSW Context	
Can	Can	
BSW Parameter		**BSW Type**
CanConfigSet		ParamConfContainerDef
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		ParamConfContainerDef
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		EnumerationParamDef
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		BooleanParamDef
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**
CanControllerBaudRate		IntegerParamDef
BSW Description		
Specifies the buadrate 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	
BSW Parameter		**BSW Type**
CanControllerId		IntegerParamDef
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**
CanControllerPropSeg		IntegerParamDef
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	
BSW Parameter		**BSW Type**
CanControllerSeg1		IntegerParamDef

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	
BSW Parameter		**BSW Type**
CanControllerSeg2		IntegerParamDef
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	
BSW Parameter		**BSW Type**
CanControllerTimeQuanta		FloatParamDef
BSW Description		
Specifies the time quanta for the controller. The calculation of the resulting prescaler value depending on module clocking and time quanta shall be done offline Hardware specific.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanCpuClockRef		ReferenceParamDef
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

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Can</td><td colspan="2">Can/CanConfigSet/CanController</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanFilterMask</td><td>ParamConfContainerDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This container contains the configuration (parameters) of the CAN Filter Mask(s).

This container is set to obsolete and will be removed in future.
Use CanHwFilterMask instead.</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>Can</td><td colspan="2">Can/CanConfigSet/CanController/CanFilterMask</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanFilterMaskValue</td><td>IntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Describes a mask for hardware-based filtering of CAN identifiers

It shall be distinguished between
- Standard identifier mask
- Extended identifier mask.</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>Can</td><td colspan="2">Can/CanConfigSet/CanController</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanRxProcessing</td><td>EnumerationParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Enables / disables API Can_MainFunction_Read() for handling PDU reception events in polling mode.</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>Can</td><td colspan="2">Can/CanConfigSet/CanController</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
</table>

CanTxProcessing			EnumerationParamDef
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		EnumerationParamDef	
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		SymbolicNameReferenceParamDef	
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	
BSW Parameter		**BSW Type**
CanHardwareObject		ParamConfContainerDef
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		ReferenceParamDef
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		ReferenceParamDef
BSW Description		
Reference to the filter mask that is used for hardware filtering togerther 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. This container is set to obsolete and will be removed in future. Use CanHwFilterMask instead.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanHandleType		EnumerationParamDef
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**
CanHwFilter		ParamConfContainerDef
BSW Description		
This container is only valid for HRHs and contains the configuration (parameters) of one hardware filter.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject/CanHwFilter	
BSW Parameter		**BSW Type**
CanHwFilterCode		IntegerParamDef
BSW Description		
Specifies (together with the filter mask) the identifiers range that passes the hardware filter.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject/CanHwFilter	
BSW Parameter		**BSW Type**
CanHwFilterMask		IntegerParamDef
BSW Description		
Describes a mask for hardware-based filtering of CAN identifiers It shall be distinguished between - Standard identifier mask - Extended identifier mask.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanHwObjectCount		IntegerParamDef
BSW Description		

Number of hardware objects used to implement one HOH. In case of a HRH this parameter defines the number of elements in the hardware FIFO or the number of shadow buffers, in case of a HTH it defines the number of hardware objects used for multiplexed transmission or for a hardware FIFO used by a FullCAN HTH.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
Can	Can/CanConfigSet/CanHardwareObject
BSW Parameter	**BSW Type**
CanIdType	EnumerationParamDef
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	IntegerParamDef
BSW Description	
This parameter is deprecated and will be removed in the future. Old 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**

CanObjectId		IntegerParamDef
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 share a common ID range. 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		EnumerationParamDef
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	
BSW Parameter		**BSW Type**
CanGeneral		ParamConfContainerDef
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**
CanDevErrorDetection		BooleanParamDef
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		BooleanParamDef
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**
CanIndex		IntegerParamDef
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		FunctionNameDef
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		FloatParamDef
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**
CanMainFunctionReadPeriod		FloatParamDef
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	
BSW Parameter		**BSW Type**
CanMainFunctionReadPeriodRef		ReferenceParamDef
BSW Description		
Reference to CAN Hardware Object which shall be polled with the configured CanMainFunction-ReadPeriod. This reference shall only be configurable if more than one period is configured via CanMainFunctionReadPeriod.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Can	Can/CanGeneral	
BSW Parameter		**BSW Type**
CanMainFunctionWakeupPeriod		FloatParamDef
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**
CanMainFunctionWritePeriod	FloatParamDef
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**
CanMainFunctionWritePeriodRef	ReferenceParamDef
BSW Description	
Reference to CAN Hardware Object which shall be polled with the configured CanMainFunction-WritePeriod. This reference shall only be configurable if more than one period is configured via CanMainFunctionWritePeriod.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
Can	Can/CanGeneral
BSW Parameter	**BSW Type**
CanMultiplexedTransmission	BooleanParamDef
BSW Description	
Specifies if multiplexed transmission 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**

CanTimeoutDurationFactor		IntegerParamDef
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**	
Can	Can/CanGeneral	
BSW Parameter		**BSW Type**
CanVersionInfoApi		BooleanParamDef
BSW Description		
Switches the Can_GetVersionInfo() API 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**
CanWakeupSupport		BooleanParamDef
BSW Description		
CAN driver support for wakeup over CAN Bus.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

9.15 CanTP Mapping

BSW Module	**BSW Context**	
CanTp	CanTp	
BSW Parameter		**BSW Type**
CanTpChannel		ParamConfContainerDef
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::TpChannel		
Mapping Rule		**Mapping Type**
Create Container ifor each CanTpChannel that exist in ECU Extract.		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel	
BSW Parameter		**BSW Type**
CanTpChannelMode		EnumerationParamDef
BSW Description		
The CAN Transport Layer supports half and full duplex channel modes.		
M2 Template	**M2 Description**	
SystemTemplate	The CAN Transport Layer supports half and full duplex channel modes.	
M2 Parameter		
TransportProtocols::CanTpConnectionChannel.channelModeType		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel	
BSW Parameter		**BSW Type**
CanTpRxNSdu		ParamConfContainerDef
BSW Description		
The following parameters needs to be configured for each CAN N-SDU that the CanTp module receives via the CanTpChannel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpAddressingFormat		EnumerationParamDef
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.		
M2 Template	**M2 Description**	
SystemTemplate	Declares which communication addressing mode is supported	
M2 Parameter		
TransportProtocols::CanTpConnectionChannel.addressingFormat		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**		
CanTp	CanTp/CanTpChannel/CanTpRxNSdu		
BSW Parameter		**BSW Type**	
CanTpBs		IntegerParamDef	
BSW Description			
Sets 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			
M2 Template	**M2 Description**		
SystemTemplate	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.		
M2 Parameter			
TransportProtocols::CanTpConnectionChannel.blockSize			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
CanTp	CanTp/CanTpChannel/CanTpRxNSdu		
BSW Parameter		**BSW Type**	
CanTpNSa		ParamConfContainerDef	
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::CanTpConnectionChannel.addressingFormat			
Mapping Rule			**Mapping Type**
Create container if addressingFormat is set to "extended".			full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpRxNSdu/CanTpNSa	
BSW Parameter		**BSW Type**
CanTpNSa		IntegerParamDef
BSW Description		

If an RxNSdu ora TxNSdu is configured for extended addressing format, this parameter contains the transport protocol source 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/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpNTa		ParamConfContainerDef
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::CanTpConnectionChannel.addressingFormat		
Mapping Rule		**Mapping Type**
Create container if addressingFormat is set to "extended".		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpRxNSdu/CanTpNTa	
BSW Parameter		**BSW Type**
CanTpNTa		IntegerParamDef
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/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpNar		FloatParamDef
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**	
M2 Parameter		

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpNbr		FloatParamDef
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	N_Br is the elapsed time (in seconds) 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 Parameter		
TransportProtocols::CanTpConnectionChannel.timeoutBr		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpNcr		FloatParamDef
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	Value in seconds of the N_Cr timeout. N_Cr is the time until reception of the next Consecutive Frame N_PDU.	
M2 Parameter		
TransportProtocols::CanTpConnectionChannel.timeoutCr		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpRxDl		IntegerParamDef
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. This parameter is set to obsolete and will be removed in future.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
CanTp	CanTp/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpRxNPdu		ParamConfContainerDef
BSW Description		
Used for grouping of the ID of a PDU and the Reference to a PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanTp	CanTp/CanTpChannel/CanTpRxNSdu/CanTpRxNPdu	
BSW Parameter		**BSW Type**
CanTpRxNPduId		IntegerParamDef
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 addressing format, the same N-PDU identifier can be used for several N-SDU identifiers. The distinction is made by the N_TA 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/CanTpChannel/CanTpRxNSdu/CanTpRxNPdu	
BSW Parameter		**BSW Type**
CanTpRxNPduRef		ReferenceParamDef
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/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**

CanTpRxNSduId		IntegerParamDef	
BSW Description			
Unique identifier to a structure that contains all useful information to process the reception of a RxNsdu.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanTp	CanTp/CanTpChannel/CanTpRxNSdu		
BSW Parameter		**BSW Type**	
CanTpRxNSduRef		ReferenceParamDef	
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/CanTpChannel/CanTpRxNSdu		
BSW Parameter		**BSW Type**	
CanTpRxPaddingActivation		EnumerationParamDef	
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::CanTpConnectionChannel.paddingActivation			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
CanTp	CanTp/CanTpChannel/CanTpRxNSdu		
BSW Parameter		**BSW Type**	
CanTpRxTaType		EnumerationParamDef	
BSW Description			

Declares the communication type of this Rx N-SDU.	
M2 Template	**M2 Description**
SystemTemplate	Network Target Address type.
M2 Parameter	
TransportProtocols::CanTpConnectionChannel.taType	
Mapping Rule	**Mapping Type**
In the System Template the TpConnection resembles both directions (Rx and Tx)	full

BSW Module	**BSW Context**
CanTp	CanTp/CanTpChannel/CanTpRxNSdu
BSW Parameter	**BSW Type**
CanTpRxWftMax	IntegerParamDef
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**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanTp	CanTp/CanTpChannel/CanTpRxNSdu
BSW Parameter	**BSW Type**
CanTpSTmin	IntegerParamDef
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**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanTp	CanTp/CanTpChannel/CanTpRxNSdu
BSW Parameter	**BSW Type**
CanTpStaticBlockSize	BooleanParamDef
BSW Description	
If this parameter is TRUE, the BlockSize will remain the same as in the first flow control. If this parameter is FALSE, the BlockSize is dynamic.	
M2 Template	**M2 Description**

SystemTemplate		
M2 Parameter		
TransportProtocols::CanTpConnectionChannel.blockSize		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpTxFcNPdu		ParamConfContainerDef
BSW Description		
Used for grouping of the ID of a PDU and the Reference to a PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpRxNSdu/CanTpTxFcNPdu	
BSW Parameter		**BSW Type**
CanTpTxFcNPduRef		ReferenceParamDef
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/CanTpChannel	
BSW Parameter		**BSW Type**
CanTpTxNSdu		ParamConfContainerDef
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**

CanTpAddressingMode			EnumerationParamDef
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).			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanTp	CanTp/CanTpChannel/CanTpTxNSdu		
BSW Parameter		**BSW Type**	
CanTpNSa		ParamConfContainerDef	
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::CanTpConnectionChannel.addressingFormat			
Mapping Rule			**Mapping Type**
Create container if addressingFormat is set to "extended".			full

BSW Module	**BSW Context**		
CanTp	CanTp/CanTpChannel/CanTpTxNSdu/CanTpNSa		
BSW Parameter		**BSW Type**	
CanTpNSa		IntegerParamDef	
BSW Description			
If an RxNSdu ora TxNSdu is configured for extended addressing format, this parameter contains the transport protocol source 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/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpNTa		ParamConfContainerDef
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::CanTpConnectionChannel.addressingFormat		
Mapping Rule		**Mapping Type**
Create container if addressingFormat is set to "extended".		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpTxNSdu/CanTpNTa	
BSW Parameter		**BSW Type**
CanTpNTa		IntegerParamDef
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/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpNas		FloatParamDef
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	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 Parameter		
TransportProtocols::CanTpConnectionChannel.timeoutAs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpNbs		FloatParamDef
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	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 Parameter		

TransportProtocols::CanTpConnectionChannel.timeoutBs	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpNcs		FloatParamDef
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	N_Cs is 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.	
M2 Parameter		
TransportProtocols::CanTpConnectionChannel.timeoutCs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpRxFcNPdu		ParamConfContainerDef
BSW Description		
Used for grouping of the ID of a PDU and the Reference to a PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpChannel/CanTpTxNSdu/CanTpRxFcNPdu	
BSW Parameter		**BSW Type**
CanTpRxFcNPduId		IntegerParamDef
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/CanTpChannel/CanTpTxNSdu/CanTpRxFcNPdu		
BSW Parameter		**BSW Type**	
CanTpRxFcNPduRef		ReferenceParamDef	
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/CanTpChannel/CanTpTxNSdu		
BSW Parameter		**BSW Type**	
CanTpTxDl		IntegerParamDef	
BSW Description			
Data Length Code of this TxNsdu. In case of variable length message, this value indicates the minimum data length. This parameter is set to obsolete and will be removed in future.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanTp	CanTp/CanTpChannel/CanTpTxNSdu		
BSW Parameter		**BSW Type**	
CanTpTxNPdu		ParamConfContainerDef	
BSW Description			
Used for grouping of the ID of a PDU and the Reference to a PDU.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
CanTp	CanTp/CanTpChannel/CanTpTxNSdu/CanTpTxNPdu	
BSW Parameter		**BSW Type**
CanTpTxNPduRef		ReferenceParamDef
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/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpTxNSduId		IntegerParamDef
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/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpTxNSduRef		ReferenceParamDef
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/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpTxPaddingActivation		EnumerationParamDef
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::CanTpConnectionChannel.paddingActivation	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**		
CanTp	CanTp/CanTpChannel/CanTpTxNSdu		
BSW Parameter		**BSW Type**	
CanTpTxTaType		EnumerationParamDef	
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::CanTpConnectionChannel.taType			
Mapping Rule			**Mapping Type**
In the System Template the TpConnection resembles both directions (Rx and Tx)			full

BSW Module	**BSW Context**		
CanTp	CanTp		
BSW Parameter		**BSW Type**	
CanTpGeneral		ParamConfContainerDef	
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		BooleanParamDef	
BSW Description			
This parameter, if set to true, enables the CanTp_ChangeParameter 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		BooleanParamDef	
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**	
CanTpMainFunctionPeriod		FloatParamDef	
BSW Description			
Allow to configure the time for the MainFunction (as float in seconds). Please note: This configuration value shall be equal to the value in the ScheduleManger module.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanTp	CanTp/CanTpGeneral		
BSW Parameter		**BSW Type**	
CanTpPaddingByte		IntegerParamDef	
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**	
CanTpTc		BooleanParamDef	
BSW Description			
Preprocessor switch for enabling Transmit Cancellation and Receive Cancellation.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

	local

9.16 Can Nm

BSW Module	BSW Context
CanNm	CanNm
BSW Parameter	**BSW Type**
CanNmGlobalConfig	ParamConfContainerDef
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**
CanNmActiveWakeupBitEnabled	BooleanParamDef
BSW Description	
Please note that this parameter is obsolete and replaced by the CanNmActiveWakeupBitEnabled parameter in the CanNmChannelConfig container. Enables/Disables the handling of the Active Wakeup Bit in the CanNm module.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanNm	CanNm/CanNmGlobalConfig
BSW Parameter	**BSW Type**
CanNmBusLoadReductionEnabled	BooleanParamDef
BSW Description	
Pre-processor switch for enabling busload reduction support.	
M2 Template	**M2 Description**
SystemTemplate	
M2 Parameter	
Fibex::Fibex4Can::CanTopology::CanCluster.nmBusLoadReductionEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
CanNm	CanNm/CanNmGlobalConfig

BSW Parameter		BSW Type
CanNmBusSynchronizationEnabled		DerivedBooleanParamDef
BSW Description		
Pre-processor switch for enabling bus synchronization support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmChannelConfig		ParamConfContainerDef
BSW Description		
This container contains the channel specific configuration parameter of the CanNm.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmActiveWakeupBitEnabled		BooleanParamDef
BSW Description		
Enables/Disables the handling of the Active Wakeup Bit in the CanNm module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmAllNmMessagesKeepAwake		BooleanParamDef
BSW Description		
Specifies if CanNm drops irrelevant NM PDUs. false: Only NM PDUs with an with PNI bit = true and containing an PN request for this ECU triggers the standard RX indication handling true: Every NM PDU 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		BooleanParamDef
BSW Description		
This parameter defines if bus load reduction for the respective NM channel is active or not.		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
Fibex::Fibex4Can::CanTopology::CanCluster.nmBusLoadReductionActive		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmCarWakeUpBitPosition		IntegerParamDef
BSW Description		
Specifies the Bit position of the CWU within the NM PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmCarWakeUpBytePosition		IntegerParamDef
BSW Description		
Specifies the Byte position of the CWU within the NM PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmCarWakeUpFilterEnabled		BooleanParamDef
BSW Description		

If CWU filtering is supported, only the CWU bit within the NM PDU 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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**		
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig		
BSW Parameter		**BSW Type**	
CanNmCarWakeUpFilterNodeId		IntegerParamDef	
BSW Description			
Source node identifier for CWU filtering. If CWU filtering is supported, only the CWU bit within the NM PDU with source node identifier CanNmCarWakeUpFilterNodeId is considered as CWU request.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig		
BSW Parameter		**BSW Type**	
CanNmCarWakeUpRxEnabled		BooleanParamDef	
BSW Description			
Enables or disables support of CarWakeUp bit evaluation in received NM PDUs. FALSE - CarWakeUp not supported TRUE - CarWakeUp supported			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig		
BSW Parameter		**BSW Type**	
CanNmChannelActive		BooleanParamDef	
BSW Description			
Please note that this parameter is obsolete and will be removed in future revisions. It determines if the respective NM channel is active or not. Indicates whether a particular NM-channel shall be initialized (TRUE) or not (FALSE). If this parameter is set to FALSE the respective CanNm channel shall not be used during runtime.			
M2 Template	**M2 Description**		

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmChannelIdRef	SymbolicNameReferenceParamDef
BSW Description	
Channel identifier configured for the respective AUTOSAR NM cluster. It is used by referring to the respective NM channel handle. It must be unique for each AUTOSAR NM cluster within one ECU.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmImmediateNmCycleTime	FloatParamDef
BSW Description	
Defines the immediate NM PDU cycle time in seconds which is used for CanNmImmediateNmTransmissions NM PDU transmissions.	
M2 Template	**M2 Description**
SystemTemplate	
M2 Parameter	
Fibex::Fibex4Can::CanTopology::CanCluster.nmImmediateNmCycleTime	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmImmediateNmTransmissions	IntegerParamDef
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	
M2 Parameter	
Fibex::Fibex4Can::CanTopology::CanCluster.nmImmediateNmTransmissions	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmMainFunctionPeriod		FloatParamDef
BSW Description		
Call cycle in seconds of CanNm_MainFunction_x for the respective CanNm channel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmMsgCycleOffset		FloatParamDef
BSW Description		
Time offset in the periodic transmission node. It determines the start delay of the transmission. Specified in seconds.		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
Fibex::Fibex4Can::CanTopology::CanCommunicationController.nmMsgCycleOffset		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmMsgCycleTime		FloatParamDef
BSW Description		
Period of a NM PDU 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 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
Fibex::Fibex4Can::CanTopology::CanCluster.nmMsgCycleTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmMsgReducedtime		FloatParamDef
BSW Description		
Node specific bus cycle time in the periodic transmission mode with bus load reduction. Specified in seconds.		
M2 Template	**M2 Description**	

SystemTemplate	
M2 Parameter	
Fibex::Fibex4Can::CanTopology::CanCommunicationController.nmMsgReducedTime	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmMsgTimeoutTime	FloatParamDef
BSW Description	
Transmission Timeout of NM PDU.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmNetworkHandle	ReferenceParamDef
BSW Description	
Channel identifier configured for the respective CanNm channel. The CanNmChannelHandle shall be encoded in the canNmRxPduId parameter which is passed to CanNm_RxIndication() function called by the CanIf.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmNodeId	IntegerParamDef
BSW Description	
Node identifier of local node	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**

CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmPduCbvPosition		EnumerationParamDef
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). ImplementationType: CanNm_PduPositionType		
M2 Template	**M2 Description**	
SystemTemplate	Defines the position of the control bit vector within the NM PDU (Bitpositon). If this attribute is not configured the Control Bit Vector is not used.	
M2 Parameter		
CoreCommunication::NmPdu.nmCbvPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmPduLength		IntegerParamDef
BSW Description		
Defines the length of the NM PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmPduNidPosition		EnumerationParamDef
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) ImplementationType: CanNm_PduPositionType		
M2 Template	**M2 Description**	
SystemTemplate	Defines the position of the source node identifier within the NM PDU.	
M2 Parameter		
CoreCommunication::NmPdu.nmNidPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmPnEnabled		BooleanParamDef
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		BooleanParamDef
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		ReferenceParamDef
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		BooleanParamDef

BSW Description	
Specifies if CanNm performs an additional transition from Network Mode to Repeat Message State (true) or not (false).	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmRemoteSleepIndTime	FloatParamDef
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.	
M2 Template	**M2 Description**
SystemTemplate	
M2 Parameter	
Fibex::Fibex4Can::CanTopology::CanCluster.nmRemoteSleepIndicationTime	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmRepeatMessageTime	FloatParamDef
BSW Description	
Timeout for Repeat Message State. It defines the time in seconds how long the NM shall stay in the Repeat Message State.	
M2 Template	**M2 Description**
SystemTemplate	
M2 Parameter	
Fibex::Fibex4Can::CanTopology::CanCluster.nmRepeatMessageStateTime	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmRetryFirstMessageRequest	BooleanParamDef
BSW Description	
Specifies if first message request in CanNm is repeated until accepted by CanIf.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmRxPdu		ParamConfContainerDef
BSW Description		
This container is used to configure the Rx PDU Properties that are used for the CanNm Channel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig/CanNmRxPdu	
BSW Parameter		**BSW Type**
CanNmRxPduId		IntegerParamDef
BSW Description		
This parameter defines the Rx PDU ID of the CanIf L-PDU range that is associated with this CanNmChannel. ImplementationType: PduIdType		
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		ReferenceParamDef
BSW Description		
Reference to the global PDU that is used by this CanNmChannel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmTimeoutTime		FloatParamDef
BSW Description		
Network Timeout for NM PDUs. 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.		
M2 Template	**M2 Description**	

SystemTemplate		
M2 Parameter		
Fibex::Fibex4Can::CanTopology::CanCluster.nmTimeoutTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmTxPdu		ParamConfContainerDef
BSW Description		
This container is used to configure the Tx PDU Properties that are used for the CanNm Channel.		
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		ReferenceParamDef
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		IntegerParamDef
BSW Description		
Defines the length of the user data contained in the NM PDU. Please note that this parameter is deprecated and will be removed in future.		
M2 Template	**M2 Description**	
SystemTemplate	Defines the length in Bytes of the user data contained in the NM PDU.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::NmPdu.nmUserDataLength		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**

CanNmUserDataTxPdu		ParamConfContainerDef	
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			
SystemTemplate::Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping			
Mapping Rule			**Mapping Type**
Create container for each NmPdu transmitted by the regarded ECU 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		IntegerParamDef	
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		ReferenceParamDef	
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		FloatParamDef
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.		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		

Fibex::Fibex4Can::CanTopology::CanCluster.nmWaitBusSleepTime	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmComControlEnabled		DerivedBooleanParamDef
BSW Description		
Pre-processor switch for enabling the Communication Control support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmComUserDataSupport		BooleanParamDef
BSW Description		
Enable/disable the user data support.		
M2 Template	**M2 Description**	
SystemTemplate	The ISignalToIPduMapping aggregation is used to describe NmUserData that is transmitted in the NmPdu.	
M2 Parameter		
SystemTemplate::Fibex::FibexCore::CoreCommunication::NmPdu.iSignalToIPduMapping		
Mapping Rule		**Mapping Type**
If the regarded EcuInstance sends a NmPdu that contains an ISignalToIPdu Mapping that refers to an ISignal then the value of this parameter shall be true.		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmConfigPtr		IntegerParamDef
BSW Description		
Pointer to configuration of CanNm. This parameter is set to obsolete and will be removed in future.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig

BSW Parameter		BSW Type
CanNmCoordinatorSyncSupport		BooleanParamDef
BSW Description		
Enables/disables the coordinator synchronisation support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context		
CanNm	CanNm/CanNmGlobalConfig		
BSW Parameter		**BSW Type**	
CanNmDevErrorDetect		BooleanParamDef	
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		BooleanParamDef	
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			
M2 Parameter			
Fibex::Fibex4Can::CanTopology::CanCluster.nmImmediateRestartEnabled			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
CanNm	CanNm/CanNmGlobalConfig		
BSW Parameter		**BSW Type**	
CanNmImmediateTxconfEnabled		BooleanParamDef	
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**
CanNmNodeDetectionEnabled		DerivedBooleanParamDef
BSW Description		
Pre-processor switch for enabling the node detection support.		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
Fibex::FibexCore::CoreTopology::CommunicationCluster.nmNodeDetectionEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmNodeIdEnabled		DerivedBooleanParamDef
BSW Description		
Pre-processor switch for enabling the source node identifier.		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
Fibex::FibexCore::CoreTopology::CommunicationCluster.nmNodeIdEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmNumberOfChannels		IntegerParamDef
BSW Description		
Number of Can NM channels allowed within one ECU. Please note that this parameter is deprecated and will be removed in future.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmPassiveModeEnabled		DerivedBooleanParamDef
BSW Description		
Pre-processor switch for enabling support of the Passive Mode.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmPduRxIndicationEnabled		DerivedBooleanParamDef
BSW Description		
Pre-processor switch for enabling the PDU Rx Indication.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmPnEiraCalcEnabled		BooleanParamDef
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		ReferenceParamDef
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		ParamConfContainerDef

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		ParamConfContainerDef	
BSW Description			
Filter mask byte configuration			
M2 Template	**M2 Description**		
TPS_SYST	For one EcuInstance all contributing pncWakeupDataMask will be bitwise ORed to obtain aggregated pncWakeupDataMask value for this ECU. Since the pncWakeupDataMask is calculated over the whole payload (8 Byte) of the NmPdu, the leading Bytes of this aggregated pncWakeupDataMask shall be ignored based on the System.pncVectorOffset value. In order to get the CanNmPnFilterMaskByteIndex and CanNmPnFilterMaskByteValue for all the bytes aggregated pncWakeupDataMask shall be processed in a littleEndian way. E.g. if pncVectorOffset = 2 and aggregated pncWakeupDataMask has the value 2ˆ 63 this will end up in a CanNmPnFilterMaskByte with CanNmPnFilterMaskByteIndex = 5 and CanNmPnFilterMaskByteValue = 128.		
M2 Parameter			
SystemTemplate::Fibex::Fibex4Can::CanTopology::CanCommunicationConnector.pncWakeupDataMask			
Mapping Rule			**Mapping Type**
calculated			partial

BSW Module	BSW Context		
CanNm	CanNm/CanNmGlobalConfig/CanNmPnInfo/CanNmPnFilterMaskByte		
BSW Parameter		**BSW Type**	
CanNmPnFilterMaskByteIndex		IntegerParamDef	
BSW Description			
Index of the filter mask byte. Specifies the position within the filter mask byte array.			
M2 Template	**M2 Description**		
TPS_SYST	see CanNmPnFilterMaskByte		
M2 Parameter			
SystemTemplate::Fibex::Fibex4Can::CanTopology::CanCommunicationConnector.pncWakeupDataMask			
Mapping Rule			**Mapping Type**
see CanNmPnFilterMaskByte			full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmPnInfo/CanNmPnFilterMaskByte	
BSW Parameter		**BSW Type**

CanNmPnFilterMaskByteValue		IntegerParamDef
BSW Description		
Parameter to configure the filter mask byte.		
M2 Template	**M2 Description**	
TPS_SYST	see CanNmPnFilterMaskByte	
M2 Parameter		
SystemTemplate::Fibex::Fibex4Can::CanTopology::CanCommunicationConnector.pncWakeupDataMask		
Mapping Rule		**Mapping Type**
see CanNmPnFilterMaskByte		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmPnInfo	
BSW Parameter		**BSW Type**
CanNmPnInfoLength		IntegerParamDef
BSW Description		
Specifies the length of the PN request information in the NM PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmPnInfo	
BSW Parameter		**BSW Type**
CanNmPnInfoOffset		IntegerParamDef
BSW Description		
Specifies the offset of the PN request information in the NM PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmPnResetTime		FloatParamDef
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		DerivedBooleanParamDef
BSW Description		
Pre-processor switch for enabling remote sleep indication support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmRepeatMsgIndEnabled		DerivedBooleanParamDef
BSW Description		
Enable/disable the notification that a RepeatMessageRequest bit has been received.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmStateChangeIndEnabled		DerivedBooleanParamDef
BSW Description		
Pre-processor switch for enabling the CAN NM state change notification.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmUserDataEnabled		DerivedBooleanParamDef
BSW Description		
Pre-processor switch for enabling user data support.		
M2 Template	**M2 Description**	
SystemTemplate	Switch for enabling user data support.	

M2 Parameter	
Fibex::Fibex4Can::CanTopology::CanCommunicationController.nmUserDataEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmVersionInfoApi		BooleanParamDef
BSW Description		
Pre-processor switch for enabling version info API support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

A 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.

A.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 A.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).

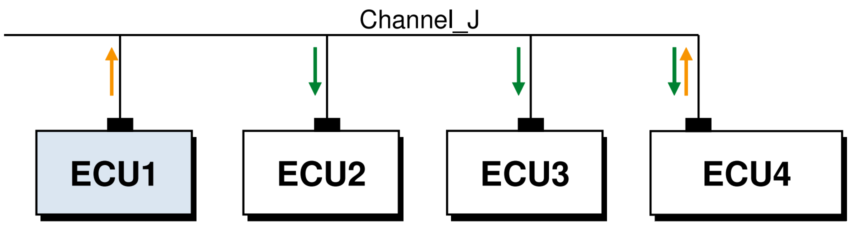

Figure A.1: Example of network setup with one Frame being received and sent on the same ECU and channel

In the system description there exists one definition for the Frame_X and one FrameTriggering for the channel (figure A.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.

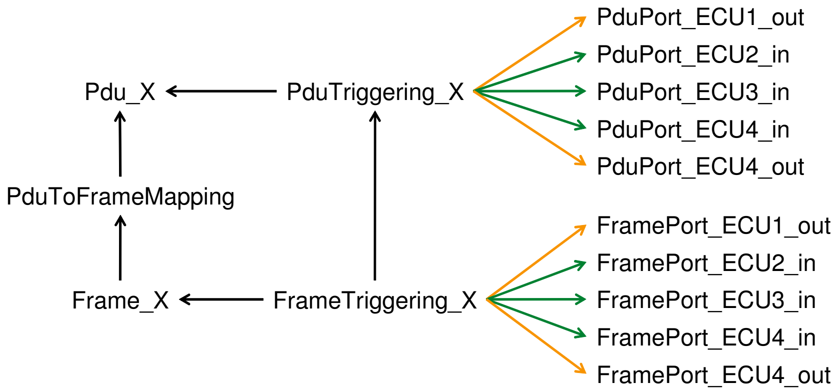

Figure A.2: Structure to reflect the frame- and pdu-triggering setup of one Frame being received and sent on the same ECU and channel

In case an 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.

A.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` with length equal to zero. Such empty signals will be mapped into `Pdus` and `Frames` and therefore `Pdus` and `Frames` with length zero are also supported by the System Template.

B Detailed Representation of InstanceRef Associations in the System Template

Class tables included in this chapter are not fully filled out in the sense that most of the notes inside the class tables are missing. The primary purpose of these class tables is to provide information about the intended order in which InstanceRefs are serialized in M1 AUTOSAR models. In particular, the information about the order in serialized M1 models can be obtained from the value of the tag xml.sequenceOffset of each attribute of an InstanceRef meta-class.

B.1 Data Mapping

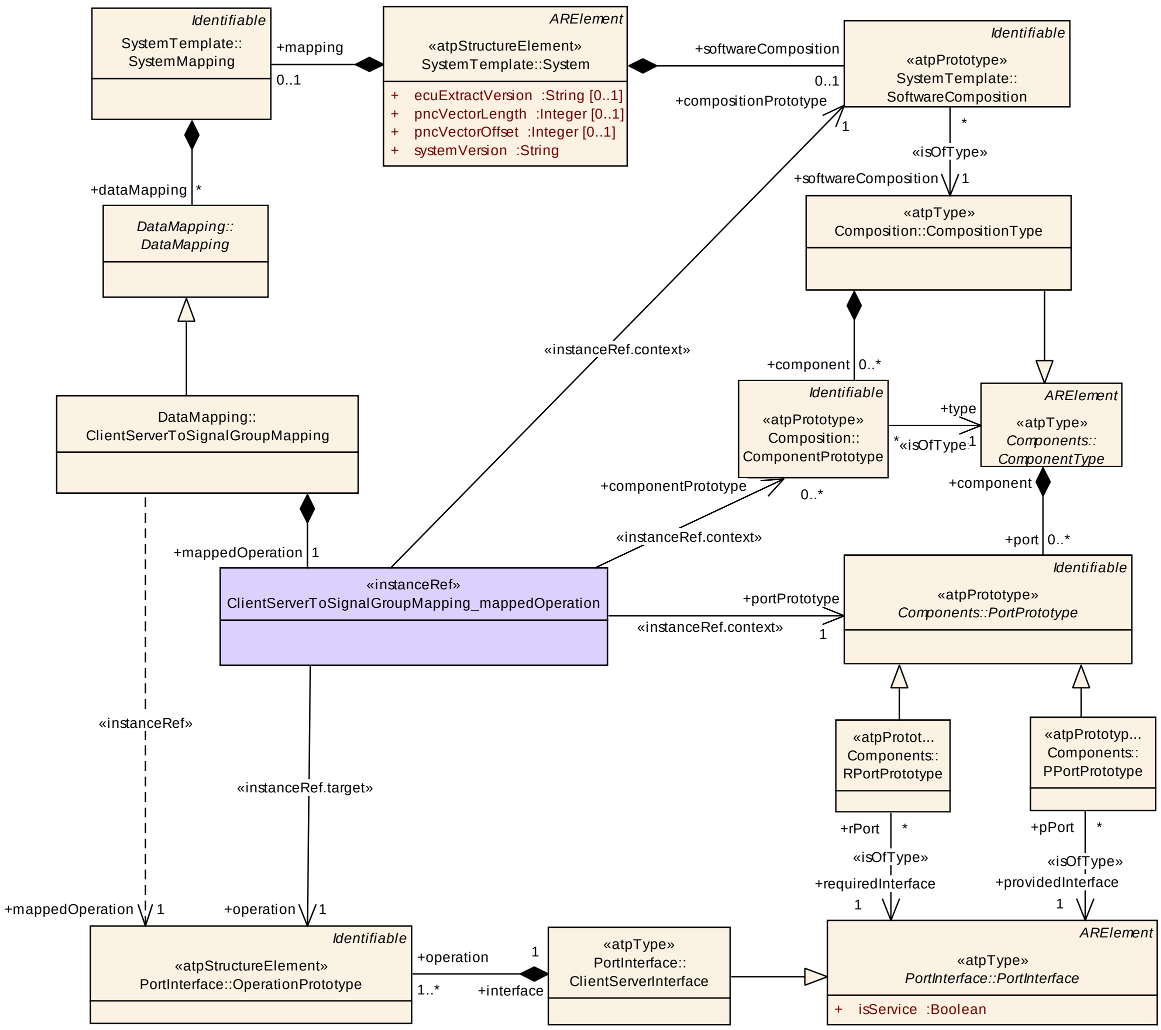

Figure B.1: Operation Mapping

Class	**ClientServerToSignalGroupMapping_mappedOperation**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping::_instanceRef			
Note				
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
componentPrototype	ComponentPrototype	*	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=20
compositionPrototype	SoftwareComposition	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=10
operation	OperationPrototype	1	ref	**Stereotypes:** instanceRef.target **Tags:** xml.sequenceOffset=40
portPrototype	PortPrototype	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=30

Table B.1: ClientServerToSignalGroupMapping_mappedOperation

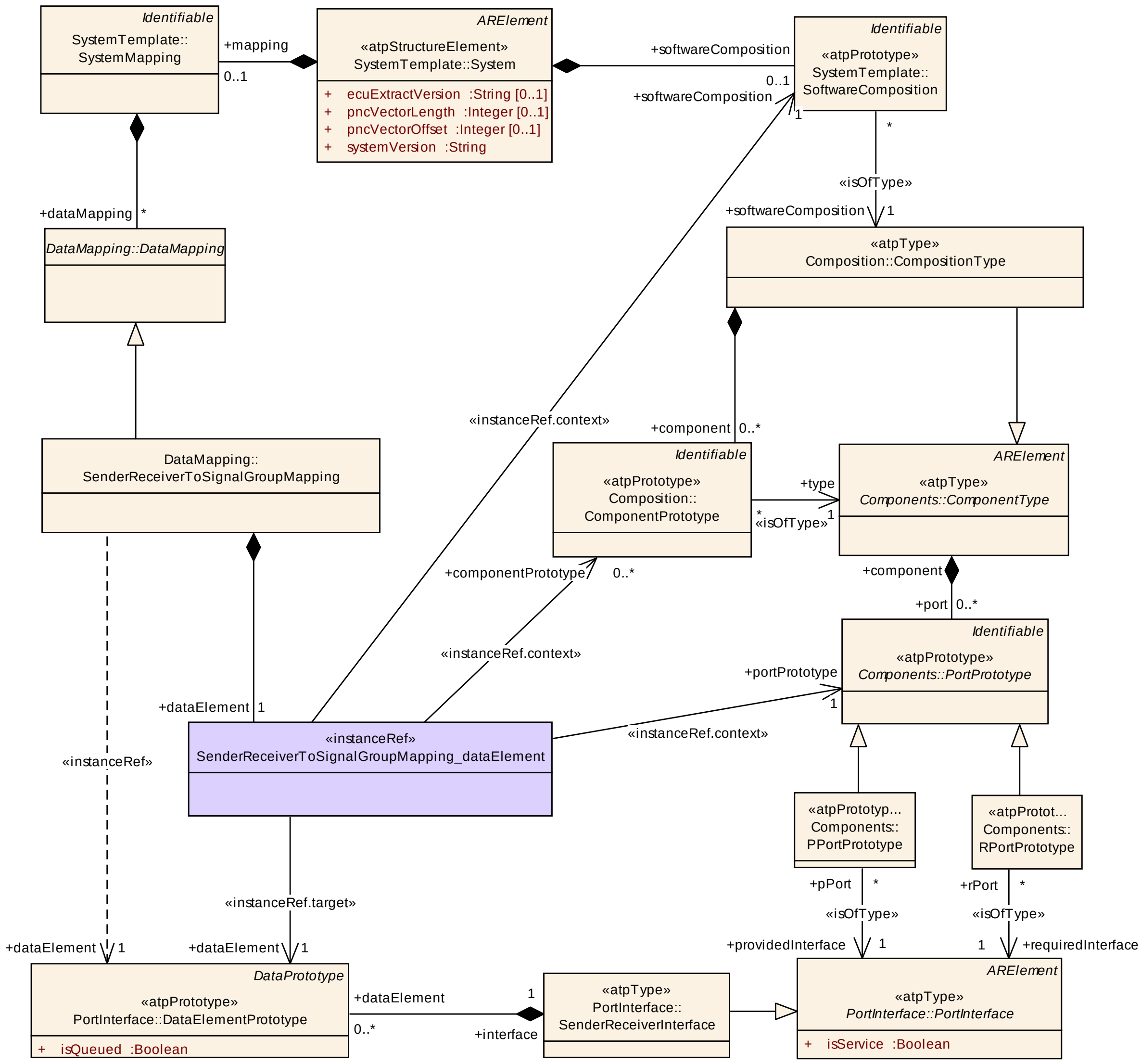

Figure B.2: composite Datatype Mapping

Class	**SenderReceiverToSignalGroupMapping_dataElement**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping::_instanceRef			
Note				
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
componentPrototype	ComponentPrototype	*	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=20
dataElement	DataElementPrototype	1	ref	**Stereotypes:** instanceRef.target **Tags:** xml.sequenceOffset=40
portPrototype	PortPrototype	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=30
softwareComposition	SoftwareComposition	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=10

Table B.2: SenderReceiverToSignalGroupMapping_dataElement

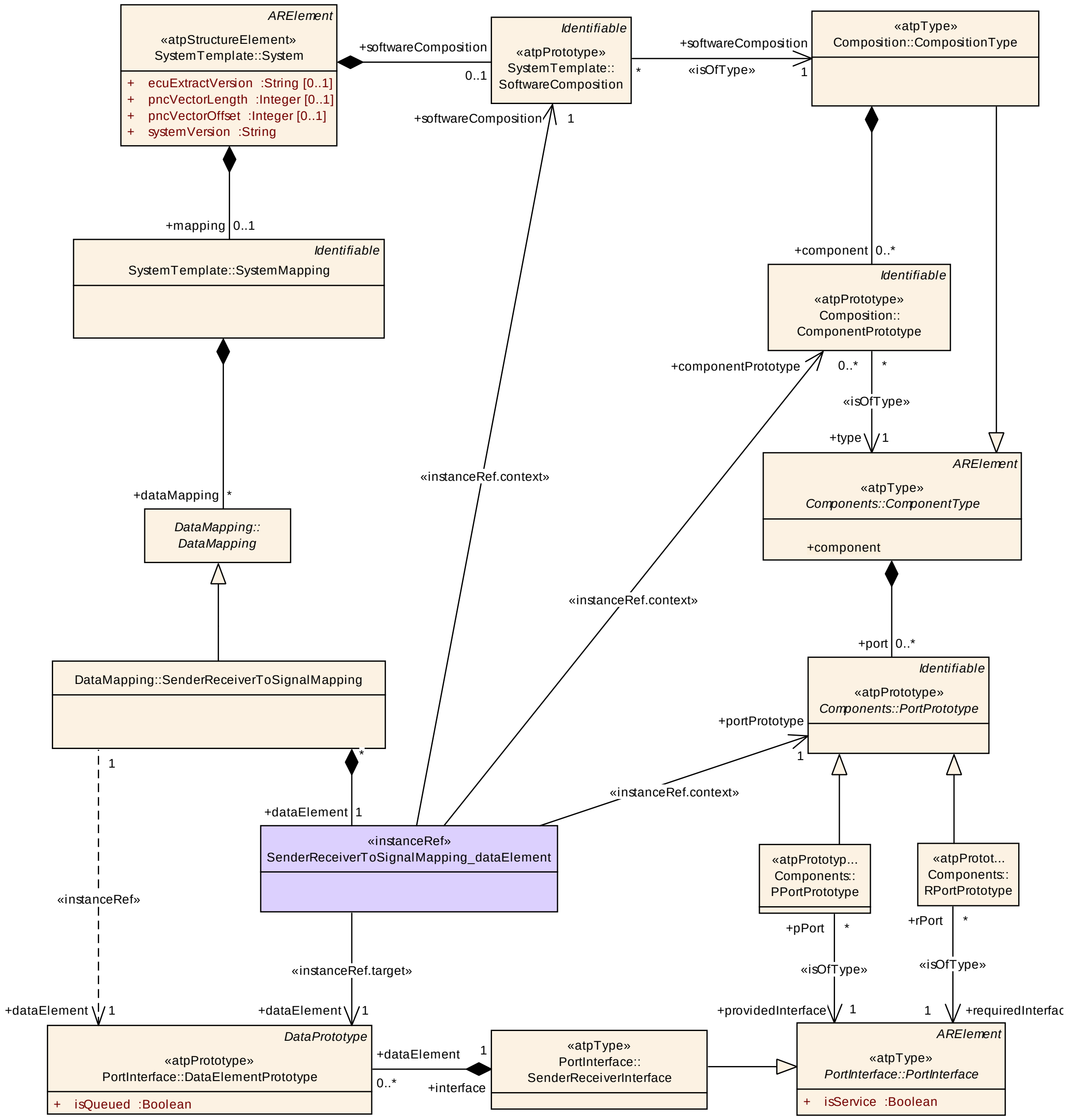

Figure B.3: primitive Datatype Mapping

Class	**SenderReceiverToSignalMapping_dataElement**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping::_instanceRef			
Note				
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
component Prototype	ComponentPrototype	*	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=20
dataElement	DataElementPrototype	1	ref	**Stereotypes:** instanceRef.target **Tags:** xml.sequenceOffset=40
portPrototype	PortPrototype	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=30

Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
softwareComposition	SoftwareComposition	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=10

Table B.3: SenderReceiverToSignalMapping_dataElement

B.2 Software Component Mapping

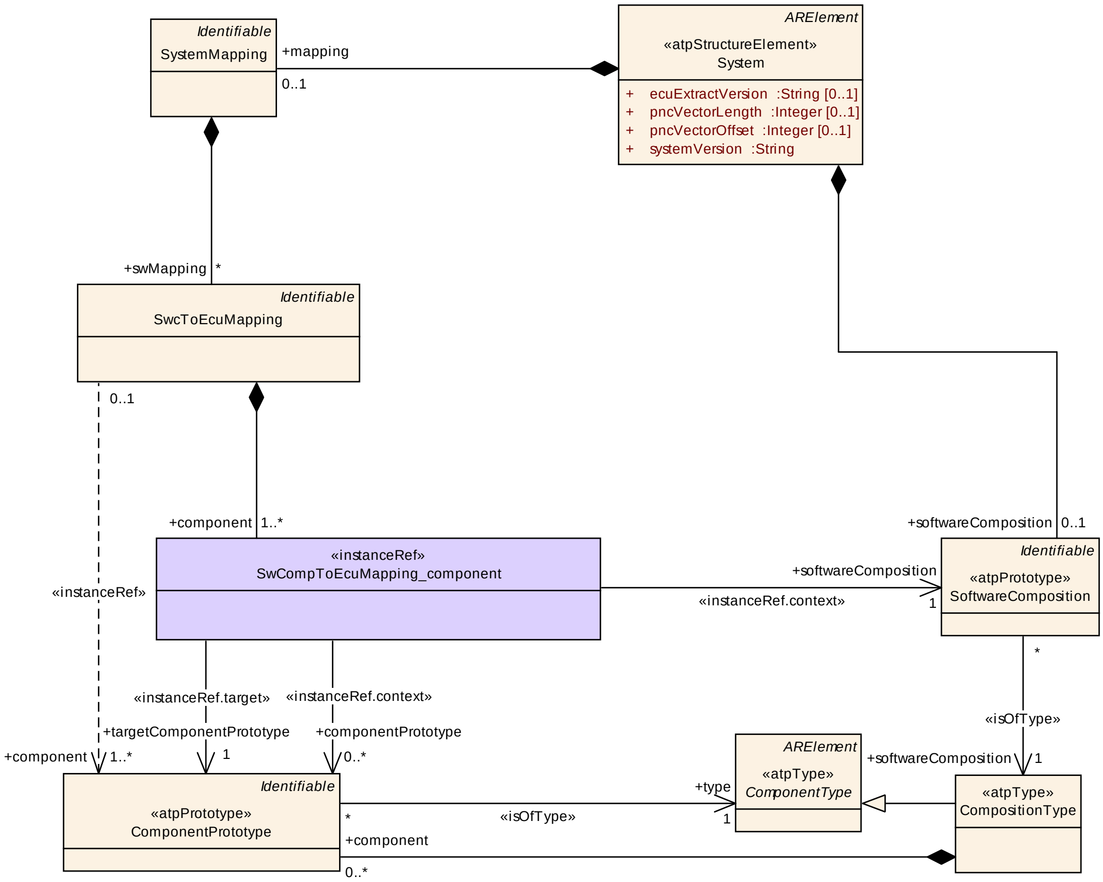

Figure B.4: SW Component To ECU Mapping

Class	**SwCompToEcuMapping_component**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping::_instanceRef			
Note				
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
componentPrototype	ComponentPrototype	*	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=20
softwareComposition	SoftwareComposition	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=10
targetComponentPrototype	ComponentPrototype	1	ref	**Stereotypes:** instanceRef.target **Tags:** xml.sequenceOffset=30

Table B.4: SwCompToEcuMapping_component

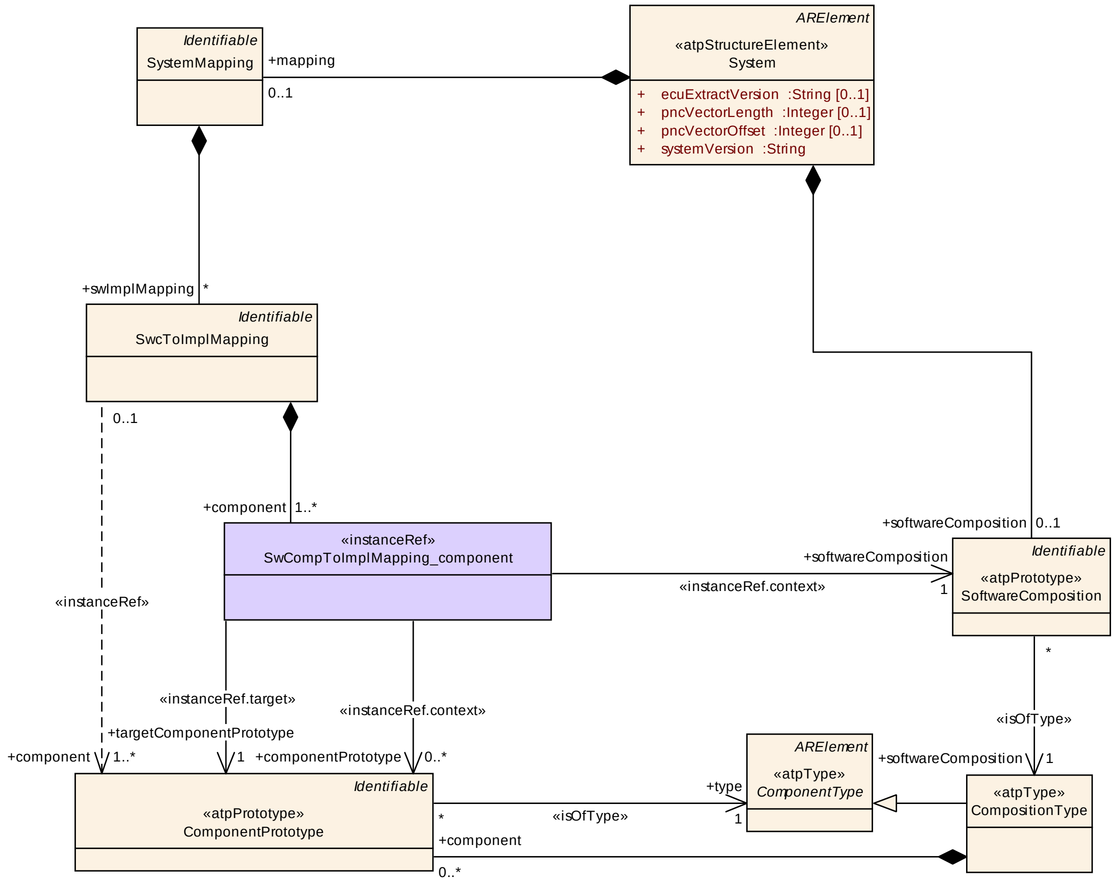

Figure B.5: SW Component To SWC Implementation Mapping

Class	**SwCompToImplMapping_component**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping::_instanceRef			
Note				
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
componentPrototype	ComponentPrototype	*	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=20
softwareComposition	SoftwareComposition	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=10
targetComponentPrototype	ComponentPrototype	1	ref	**Stereotypes:** instanceRef.target **Tags:** xml.sequenceOffset=30

Table B.5: SwCompToImplMapping_component

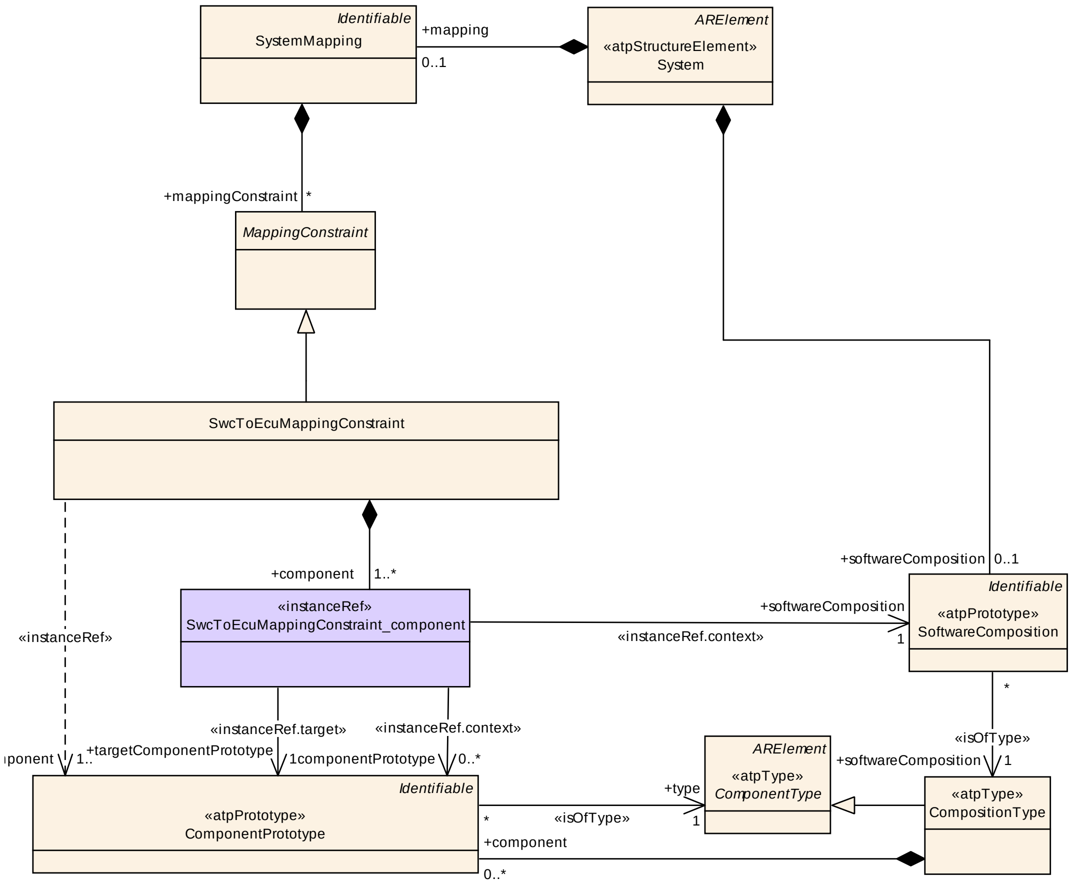

Figure B.6: SW Component To ECU Mapping Constraint

Class	**SwcToEcuMappingConstraint_component**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping::_instanceRef			
Note				
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
componentPrototype	ComponentPrototype	*	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=20
softwareComposition	SoftwareComposition	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=10
targetComponentPrototype	ComponentPrototype	1	ref	**Stereotypes:** instanceRef.target **Tags:** xml.sequenceOffset=30

Table B.6: SwcToEcuMappingConstraint_component

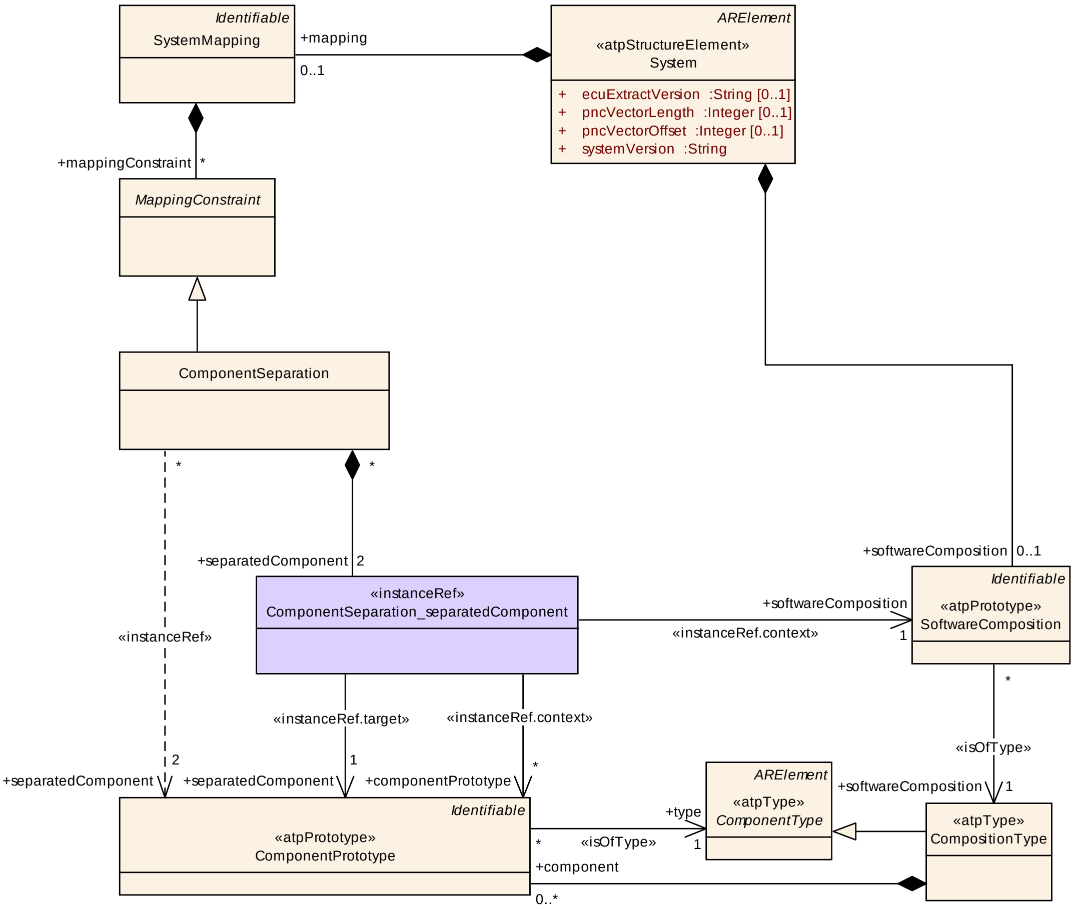

Figure B.7: SW Component Separation

Class	**ComponentSeparation_separatedComponent**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping::_instanceRef			
Note				
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
componentPrototype	ComponentPrototype	*	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=20
separatedComponent	ComponentPrototype	1	ref	**Stereotypes:** instanceRef.target **Tags:** xml.sequenceOffset=30
softwareComposition	SoftwareComposition	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=10

Table B.7: ComponentSeparation_separatedComponent

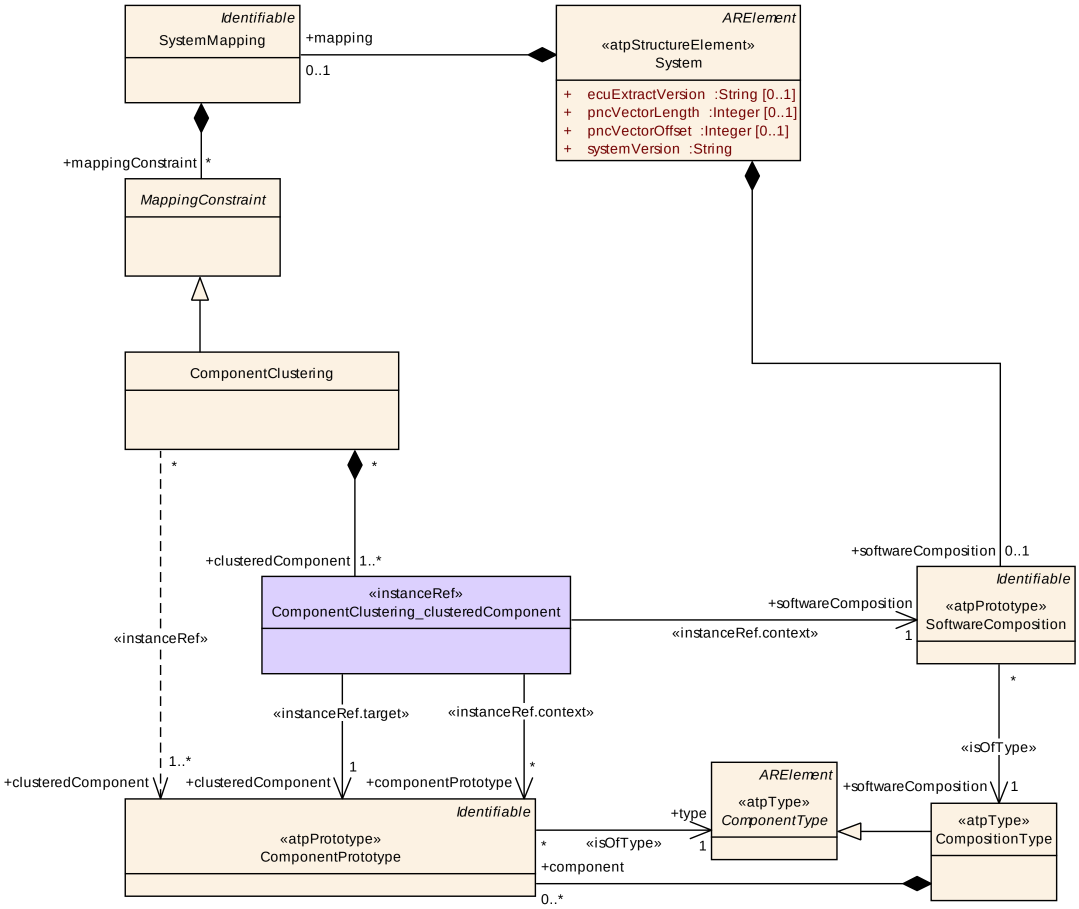

Figure B.8: SW Component Clustering

Class	**ComponentClustering_clusteredComponent**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping::_instanceRef			
Note				
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
clusteredComponent	ComponentPrototype	1	ref	**Stereotypes:** instanceRef.target **Tags:** xml.sequenceOffset=30
componentPrototype	ComponentPrototype	*	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=20
softwareComposition	SoftwareComposition	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=10

Table B.8: ComponentClustering_clusteredComponent

B.3 Signal Paths

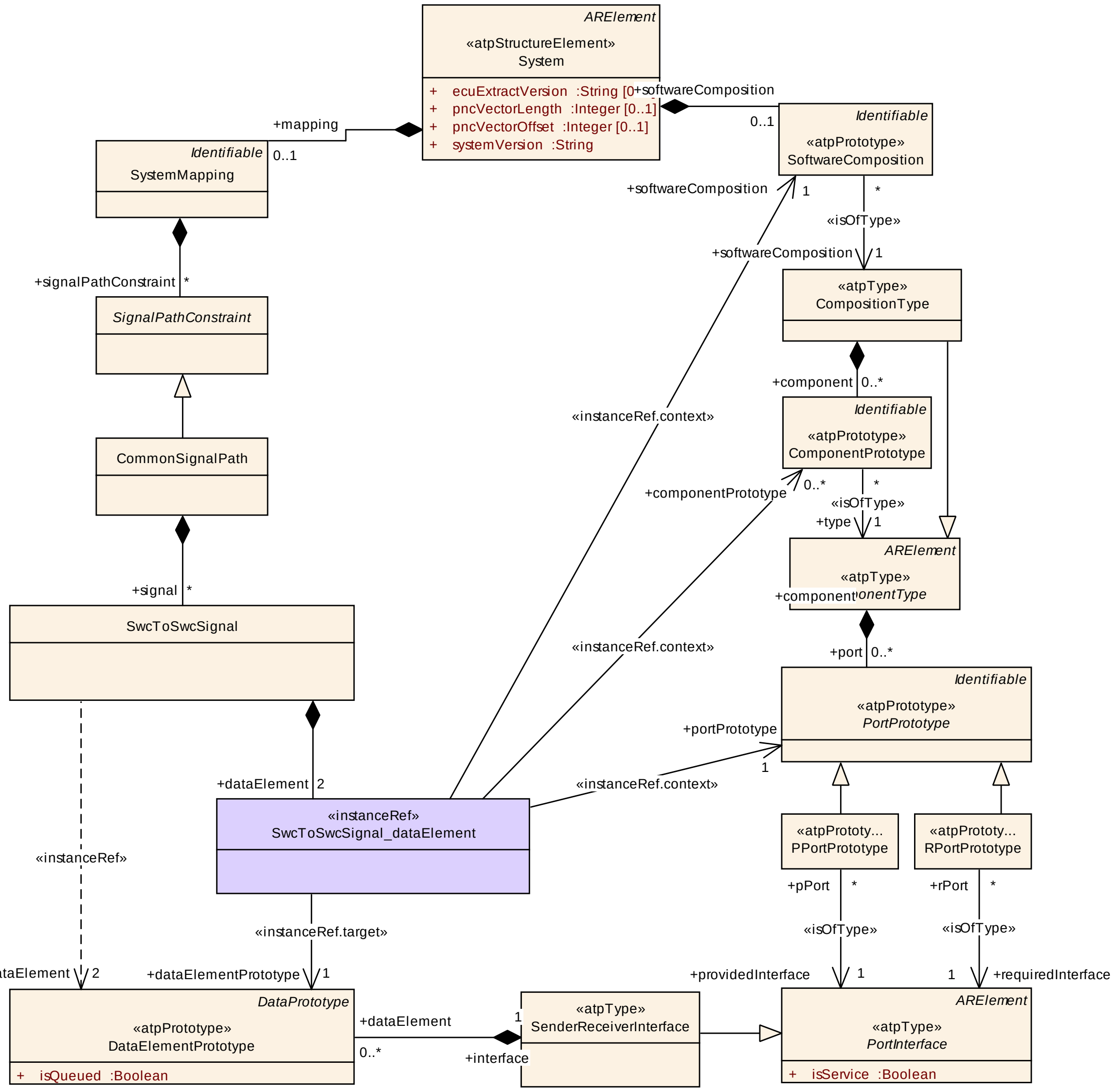

Figure B.9: SwcToSwcSignal

Class	**SwcToSwcSignal_dataElement**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths::_instanceRef			
Note				
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
componentPrototype	ComponentPrototype	*	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=20
dataElementPrototype	DataElementPrototype	1	ref	**Stereotypes:** instanceRef.target **Tags:** xml.sequenceOffset=40
portPrototype	PortPrototype	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=30

Attribute	Datatype	Mul.	Kind	Note
softwareComposition	SoftwareComposition	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=10

Table B.9: SwcToSwcSignal_dataElement

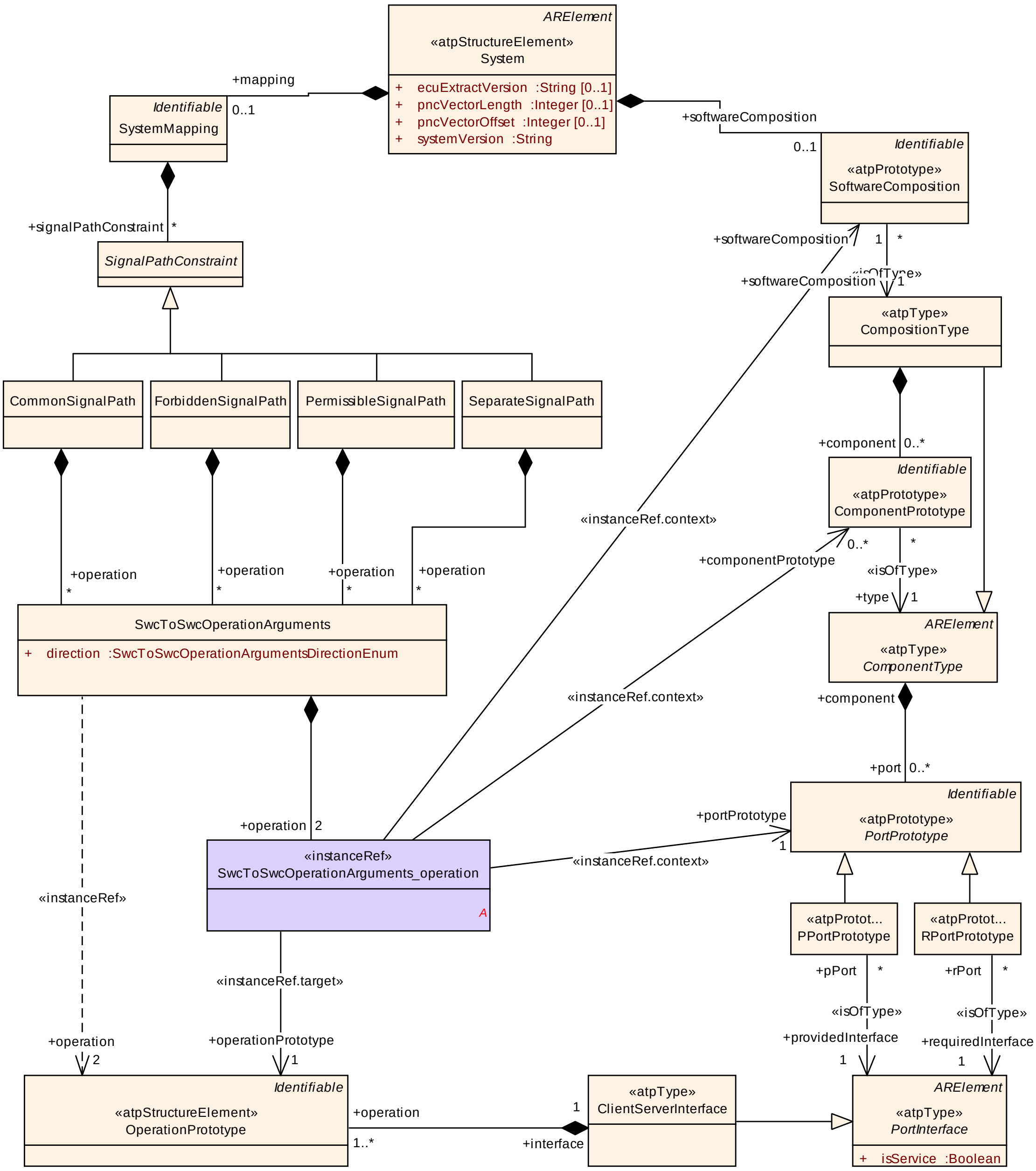

Figure B.10: SwcToSwcOperationArguments

Class	SwcToSwcOperationArguments_operation			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths::_instanceRef			
Note				
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
componentPrototype	ComponentPrototype	*	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=20
operationPrototype	OperationPrototype	1	ref	**Stereotypes:** instanceRef.target **Tags:** xml.sequenceOffset=40
portPrototype	PortPrototype	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=30
softwareComposition	SoftwareComposition	1	ref	**Stereotypes:** instanceRef.context **Tags:** xml.sequenceOffset=10

Table B.10: SwcToSwcOperationArguments_operation

C Constraint and Specification Item History

C.1 Constraint and Specification Item History of this Document according to AUTOSAR R3.2.2

C.1.1 Added Constraints in R3.2.2

Number	Heading
[constr_3020]	CommunicationDirection of containedIPduGroups
[constr_3024]	Usage of triggeredWithoutRepetition and triggeredOnChangeWithoutRepetition is not allowed for signal groups and group signals.
[constr_3025]	Usage of NPdus in TpConnections
[constr_3034]	Values of LinSlaveConfig and LinSlave attributes
[constr_3035]	User Data configuration in case NID/CBV are enabled
[constr_3039]	Restriction of pncIdentifier values
[constr_3040]	pncIdentifier range
[constr_3041]	pncVectorOffset range
[constr_3042]	pncVectorLength range
[constr_3043]	pncVector configuration in AUTOSAR Com
[constr_3044]	CBV configuration in case partial network is used

Table C.1: Added Constraints in R3.0.2

C.1.2 Added Specification Items in R3.2.2

Number	Heading
[TPS_SYST_01046]	ShortNames of LinSlaveConfig and LinSlave
[TPS_SYST_01058]	Pdu Gateway where an Ecu only routes a IPduTriggering without being interested in the content
[TPS_SYST_01059]	Relationship between FrameTriggering and CommConnectorPort
[TPS_SYST_01060]	Relationship between IPduTriggering and CommConnectorPort
[TPS_SYST_01061]	Relationship between ISignalTriggering and CommConnectorPort
[TPS_SYST_01064]	Transmit/Receive Semantics of Pdu Pools

Table C.2: Added Specification Items in R3.0.2

C.2 Constraint and Specification Item History of this Document according to AUTOSAR R3.2.3

C.2.1 Added Constraints in R3.2.3

Number	Heading
[constr_3069]	Allowed nmNidPosition values
[constr_3070]	Allowed nmCbvPosition values
[constr_3071]	Values of nmCbvPosition and nmNidPosition shall never have the same value
[constr_3072]	Allowed SystemSignal dataType references

[constr_3073]	`nmVoteInformation` only valid for FrNm
[constr_3074]	No `TransmissionAcknowledgementRequest` for multiple senders

Table C.3: Added Constraints in R3.2.3

C.2.2 Added Specification Items in R3.2.3

Number	Heading

Table C.4: Added Specification Items in R3.2.3