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

Document Version	3.3.0
Document Status	Final
Part of Release	3.1
Revision	5

Document Change History

Date	Version	Changed by	Description
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.

Disclaimer

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

Bibliography

[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 Communication
AUTOSAR_SWS_COM.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.

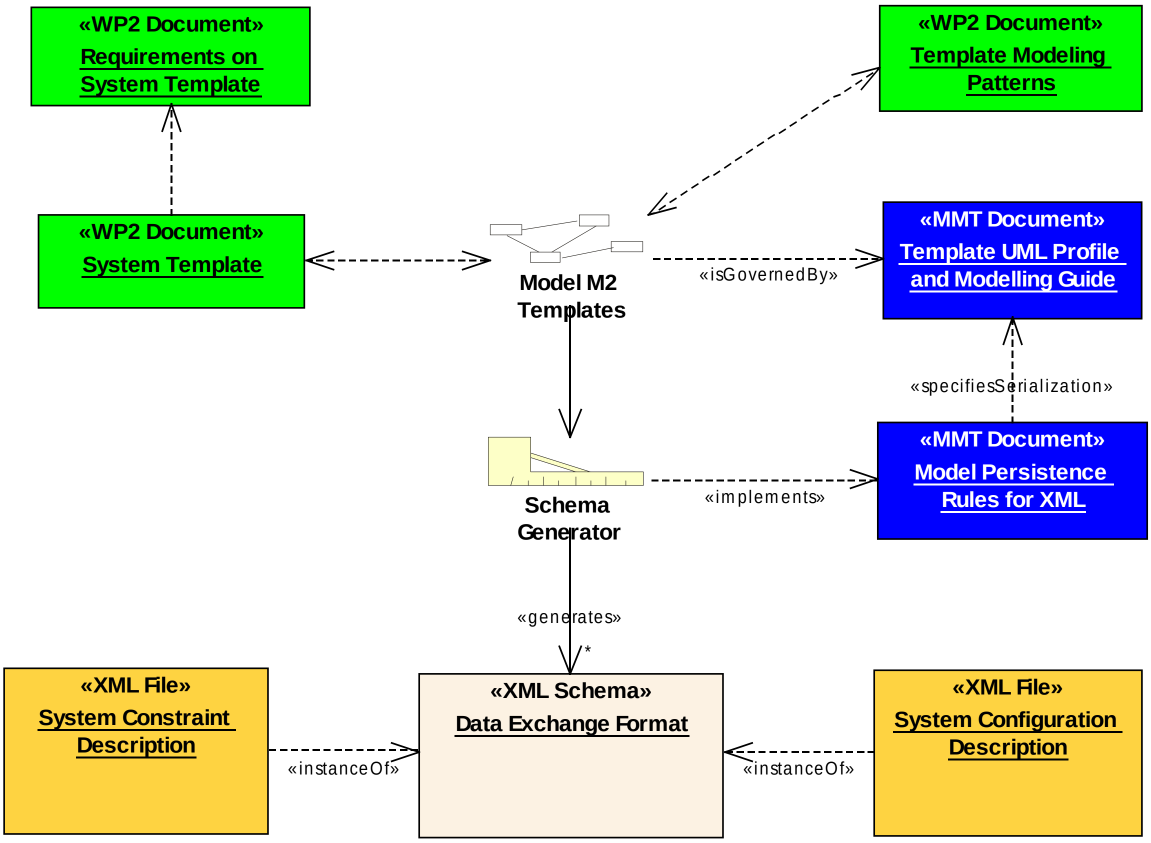

Figure 1.1: Methodology to define templates in AUTOSAR

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.

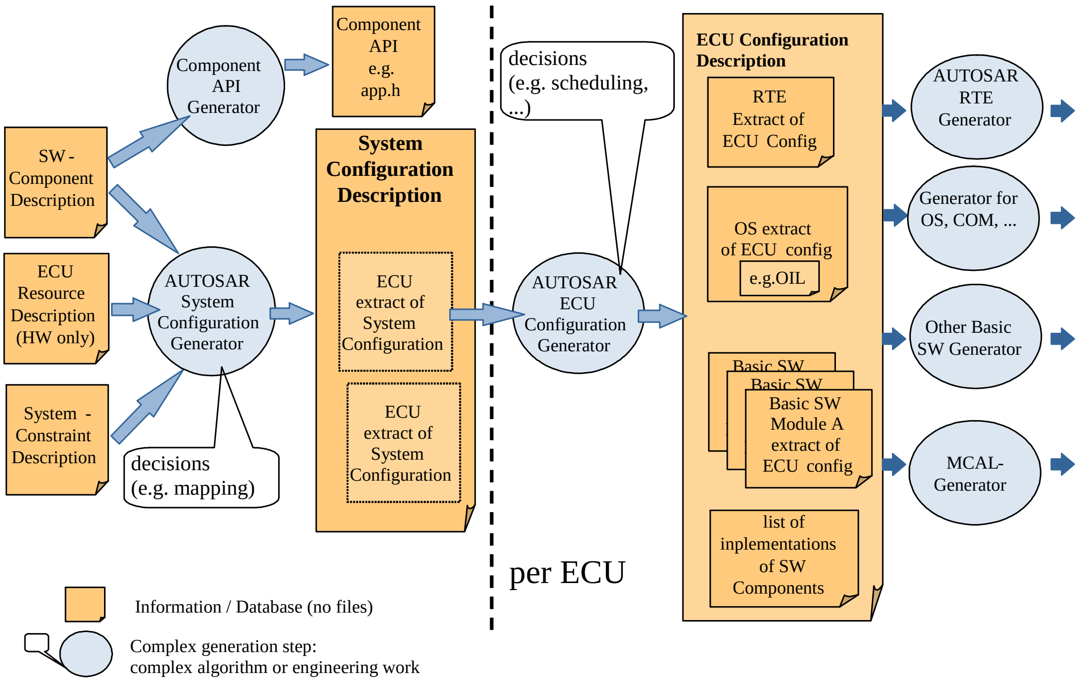

Figure 1.2: AUTOSAR Methodology

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.

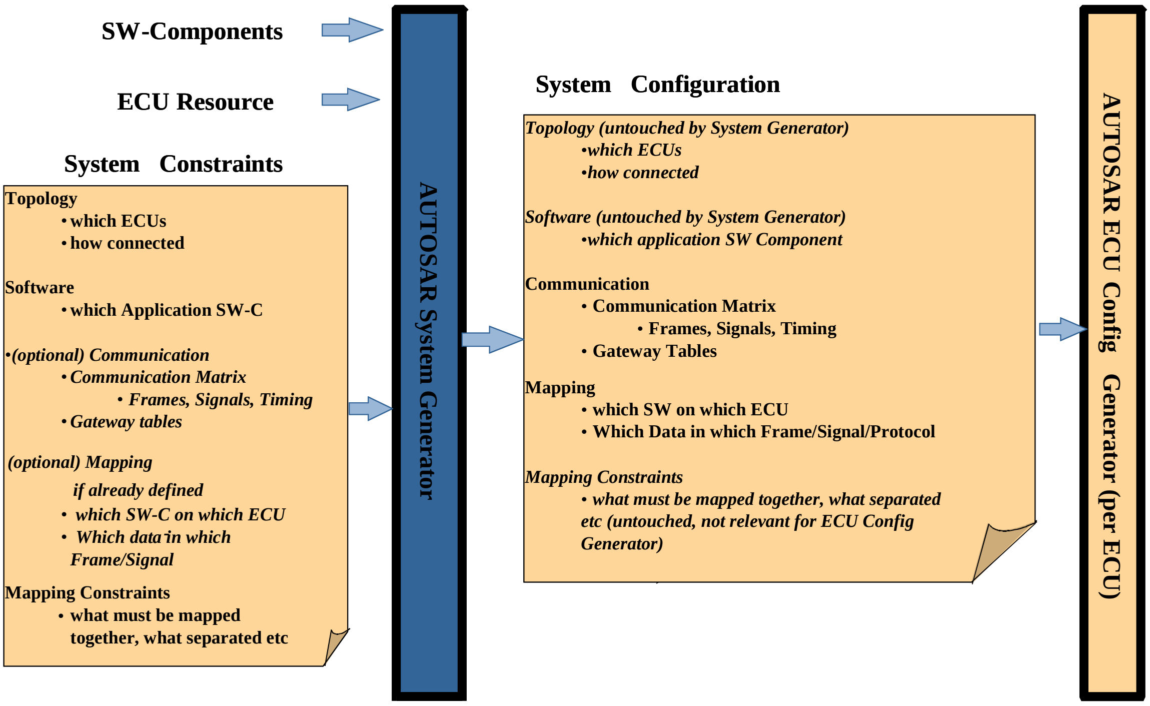

Figure 1.3: Scope of System Constraint Description and System Configuration Description

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.

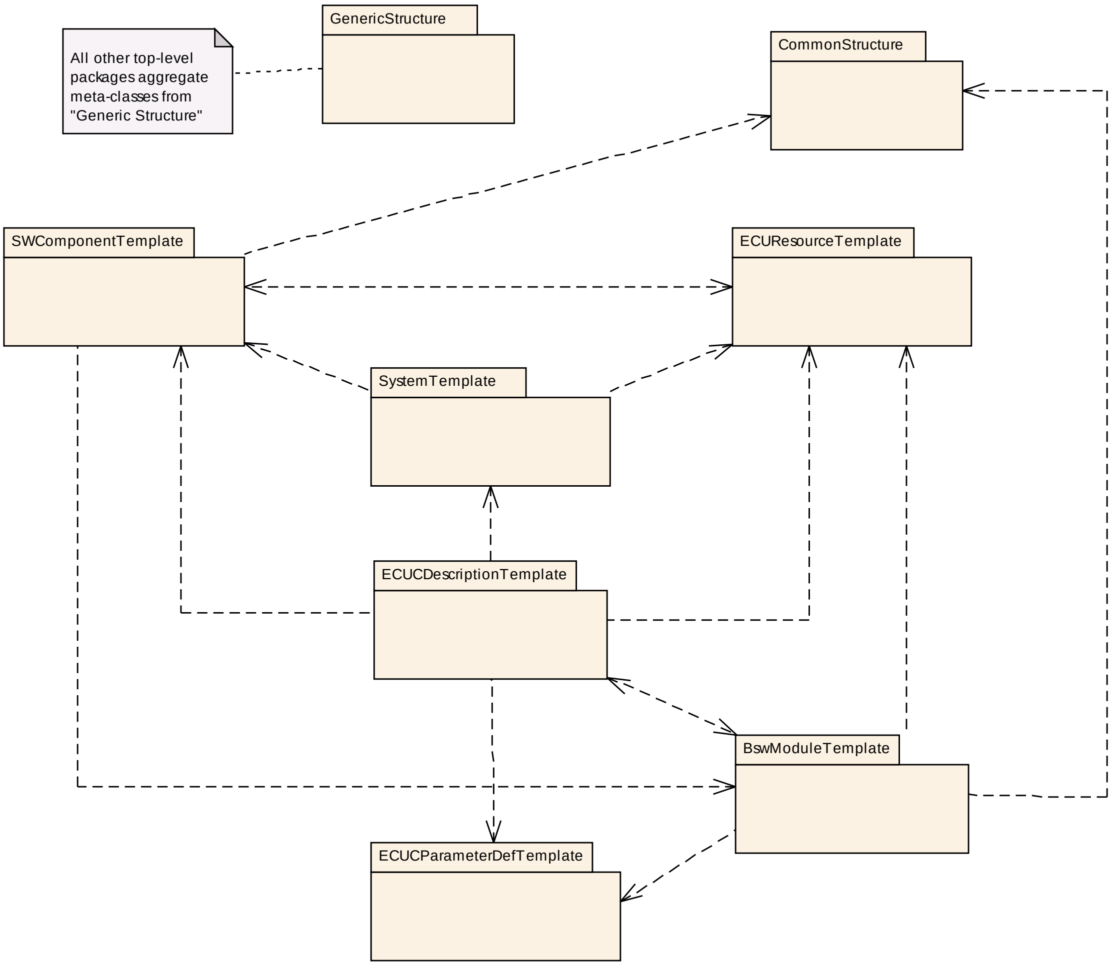

Figure 1.4: AUTOSAR Package Overview

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.

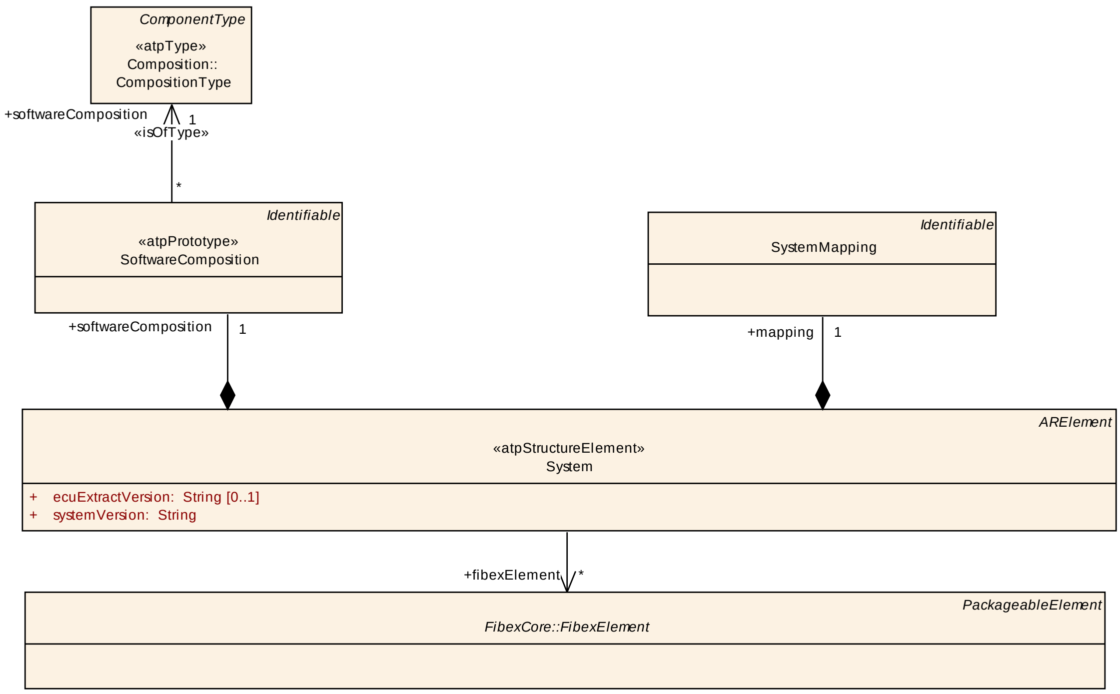

Figure 1.5: System Template Overview

[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 Software Composition elements. The SystemMapping area deals with mapping of software components to ECUs as well as with the mapping of data elements that are to be exchanged between software components onto signals and frames. The Software-Composition 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	⟨⟨**atpStructureElement**⟩⟩ **System**			
Package	M2::AUTOSARTemplates::SystemTemplate			
Class Desc.	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 Class(es)	ARElement			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
ecuExtract Version	String	0..1	aggregation	Version number of the Ecu Extract.
fibexEle-ment	Fibex Element	*	reference	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	System Mapping	1	aggregation	Aggregation of all mapping aspects (mapping of SW components to ECUs, mapping of data elements to signals, and mapping constraints).
software Composi-tion	Software Composi-tion	1	aggregation	Aggregation of the top-level software composition, containing all software components in the System in a hierarchical structure.
system Version	String	1	aggregation	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)			
Class Desc.	class description			
Base Class(es)	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.***	***Link Type***	***Description***
Attribute name	Integer	0..1	aggregation	Attribute description
Role name	referenced class	1..*	reference	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.

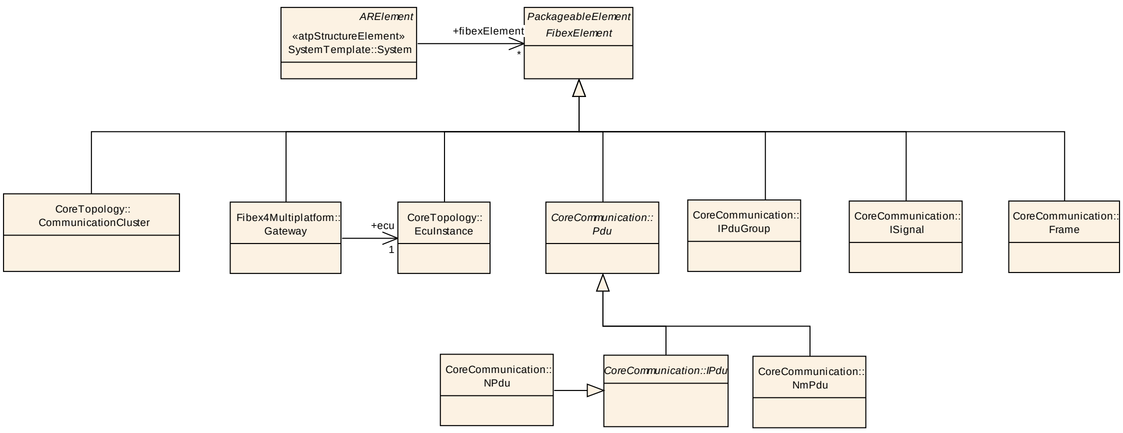

Figure 1.6: Fibex Elements

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.

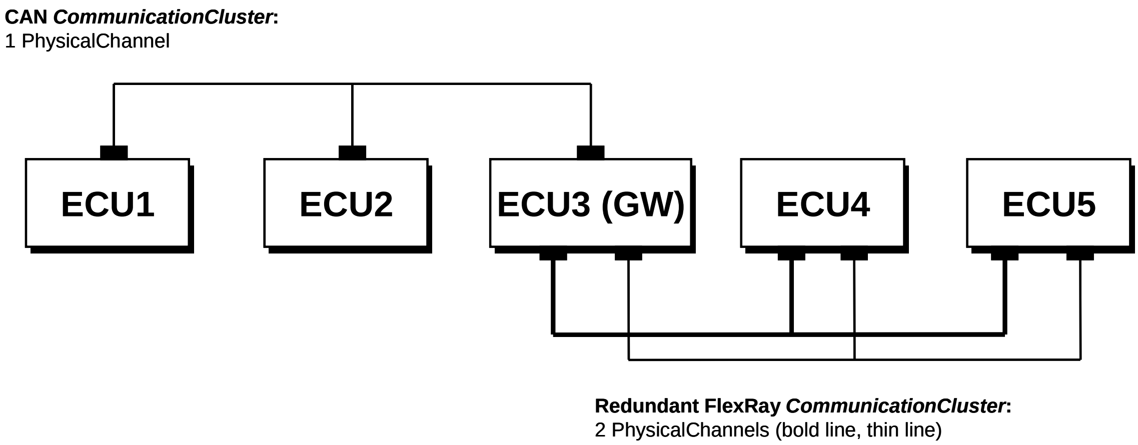

Figure 2.1: Example for a Communication Cluster within a physical network topology

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	⟨⟨**atpObject**⟩⟩ **EcuInstance**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Class Desc.	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 Class(es)	FibexElement			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
associated IPduGroup	IPduGroup	*	reference	With this reference it is possible to identify which IPduGroups are applicable for which CommunicationConnector/ ECU.
comCon-figurationId	Integer	0..1	aggregation	This ID is returned by a call to Com_GetConfigurationId()
comPro-cessing Period	Float	1	aggregation	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.
commCon-troller	Comm unication Controller	1..*	aggregation	CommunicationControllers of the ECU.
connector	Comm unication Connector	*	aggregation	All channels controlled by a single controller.
diagnostic Address	Integer	0..1	aggregation	An ECU specific ID for responses of diagnostic routines.
pduRCon-figurationId	Integer	0..1	aggregation	unique PDURconfiguration identifier
response Address	Integer	*	aggregation	An ECU specific ID for responses of diagnostic routines.
sleepMode Supported	Boolean	1	aggregation	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.
wakeUp OverBus Supported	Boolean	1	aggregation	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	⟨⟨atpObject⟩⟩ **CommunicationController**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Class Desc.	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 Class(es)	Identifiable			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
wakeUpByControllerSupported	Boolean	1	aggregation	May the ECU be woken up by this CAN Controller? TRUE: wake up is possible FALSE: wake up is not supported Note: This flag may only be set to TRUE if the feature is 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	⟨⟨**atpObject**⟩⟩ **CommunicationConnector**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Class Desc.	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.			
Base Class(es)	Identifiable			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
channel	Physical Channel	1	reference	Reference to the channel to which the ECU is connected.
commCon-troller	Comm unication Controller	1	reference	Reference to the communication controller. The CommunicationConnector and referenced CommunicationController must be aggregated by the same ECUInstance.
ecuComm PortIn-stance	Comm Connector Port	*	aggregation	An ECUs reception or send ports.
nmAd-dress	Integer	0..1	aggregation	An ECUs NM address on the referenced channel.
tpAddress	Integer	0..1	aggregation	An ECUs TP address on the referenced channel. This represents the diagnostic Address.

Table 2.3: CommunicationConnector

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.

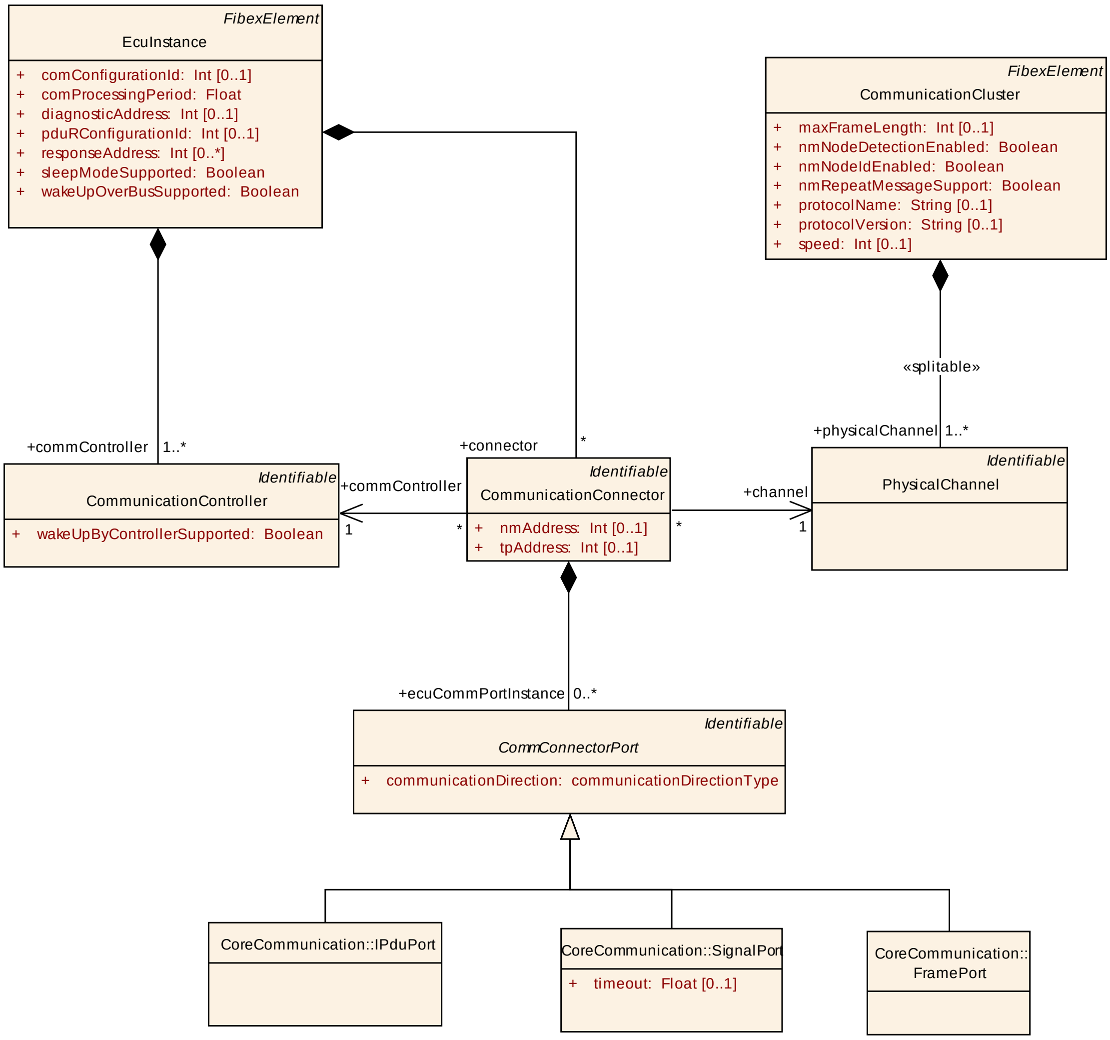

Figure 2.2: Topology elements (Topology)

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	⟨⟨**atpObject**⟩⟩ **CommunicationCluster**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Class Desc.	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 Class(es)	FibexElement			
Attribute	*Datatype*	*Mul.*	*Link Type*	*Description*
maxFrame Length	Integer	0..1	aggregation	Maximal supported length in bytes for frames in this cluster.
nmNode Detection Enabled	Boolean	1	aggregation	Enable/disable the node detection functionality.
nmNodeId Enabled	Boolean	1	aggregation	Enable/disable the source node identifier.
nmRepeat Message Support	Boolean	1	aggregation	switch for enabling support for repeat message
physical Channel	Physical Channel	1..*	aggregation	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.
protocol Name	String	0..1	aggregation	The name of the protocol used.
protocol Version	String	0..1	aggregation	The version of the protocol used.
speed	Integer	0..1	aggregation	channels speed in bits per second

Table 2.4: 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	⟨⟨**atpObject**⟩⟩ **PhysicalChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Class Desc.	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 Class(es)	Identifiable			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
TpChannel	TpChannel	*	aggregation	
frameTrig-gerings	Frame Triggering	*	aggregation	One frame triggering is defined for exactly one channel. Channels may have assigned an arbitrary number of frame triggerings.
iPduTrig-gering	IPduTrig-gering	*	aggregation	One I-Pdu triggering is defined for exactly one channel. Channels may have assigned an arbitrary number of I-Pdu triggerings.
iSignal Triggering	ISignal Triggering	*	aggregation	One ISignalTriggering is defined for exactly one channel. Channels may have assigned an arbitrary number of ISignaltriggerings.

Table 2.5: 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`.

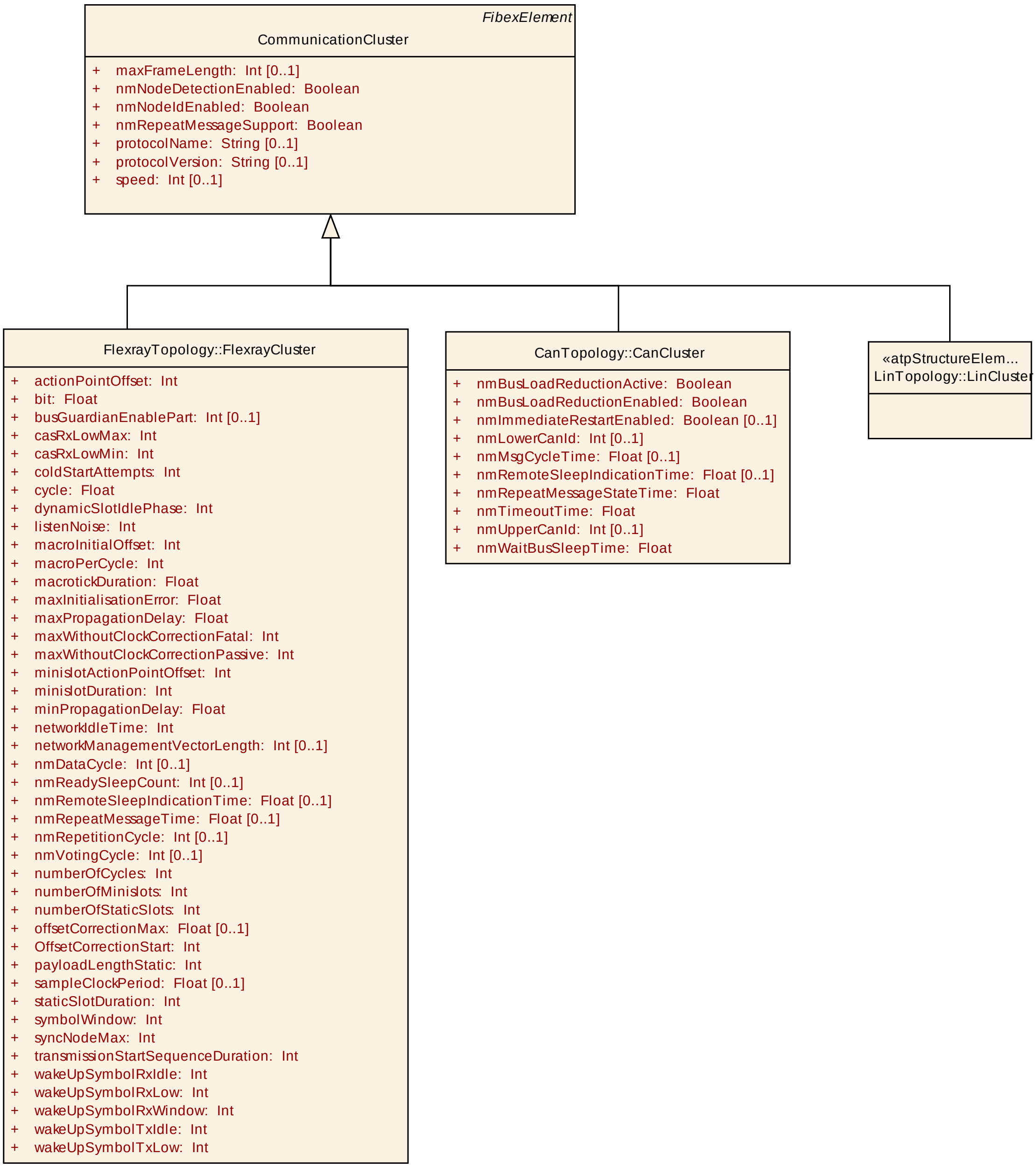

Figure 2.3: Specialized `CommunicationCluster` attributes (TopologyAttributeRefinement)

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).

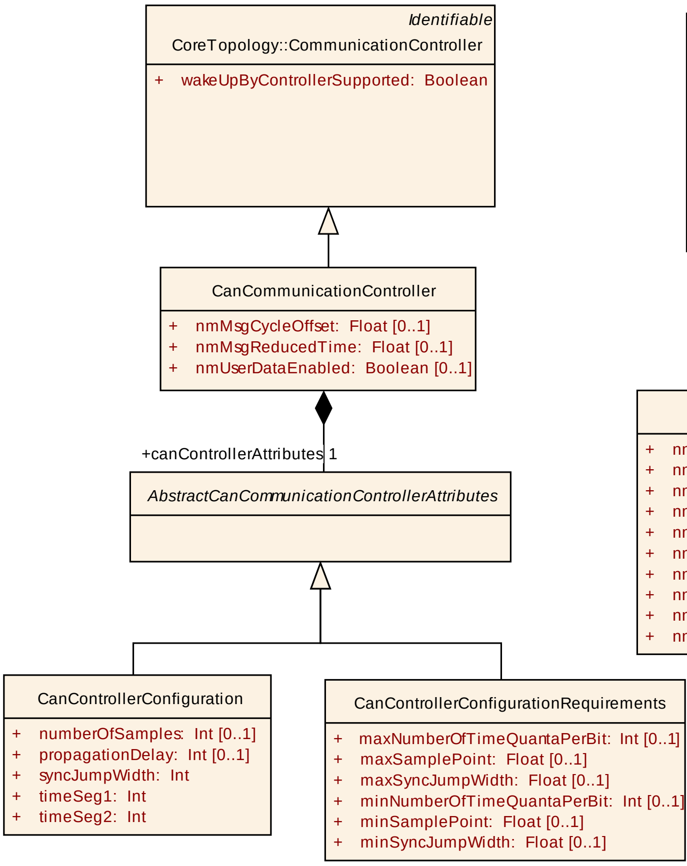

Figure 2.4: Can bus elements (Fibex4Can_Topology)

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	⟨⟨atpObject⟩⟩ CanCluster			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Class Desc.	CAN specific attributes			
Base Class(es)	CommunicationCluster			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
nmBus LoadReduction Active	Boolean	1	aggregation	It determines if bus load reduction for the respective NM channel is active or not. True: active False: inactive
nmBus LoadReduction Enabled	Boolean	1	aggregation	switch for enabling busload reduction support.
nmImmediateRestart Enabled	Boolean	0..1	aggregation	Enables the asynchronous transmission of a CanNm PDU upon bus-communication request in Prepare-Bus-Sleep mode.
nmLower CanId	Integer	0..1	aggregation	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.
nmMsg CycleTime	Float	0..1	aggregation	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.
nmRemote SleepIndicationTime	Float	0..1	aggregation	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.
nmRepeat Message StateTime	Float	1	aggregation	It defines how long the NM shall stay in the Repeat Message State (in seconds)
nmTimeout Time	Float	1	aggregation	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.
nmUpper CanId	Integer	0..1	aggregation	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.

nmWait BusSleep Time	Float	1	aggregation	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 2.6: 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	⟨⟨**atpObject**⟩⟩ **CanCommunicationController**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Class Desc.	CAN bus specific communication port attributes.			
Base Class(es)	CommunicationController			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
canController Attributes	Abstract CanCommunication Controller Attributes	1	aggregation	CAN Bit Timing configuration
nmMsgCycleOffset	Float	0..1	aggregation	Node specific time offset in the periodic transmission node. It determines the start delay of the transmission. Specified in seconds.
nmMsg Reduced Time	Float	0..1	aggregation	Node specific bus cycle time in the periodic transmission mode with bus load reduction. Specified in seconds.
nmUser DataEnabled	Boolean	0..1	aggregation	Switch for enabling user data support.

Table 2.7: CanCommunicationController

Class	⟨⟨**atpObject**⟩⟩ **AbstractCanCommunicationControllerAttributes (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Class Desc.	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 Class(es)	ARObject			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**

Table 2.8: AbstractCanCommunicationControllerAttributes

Class	⟨⟨**atpObject**⟩⟩ **CanControllerConfiguration**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Class Desc.	This element is used for the specification of the exact CAN Bit Timing configuration parameter values.			
Base Class(es)	AbstractCanCommunicationControllerAttributes			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
numberOf Samples	Integer	0..1	aggregation	Number of samples. Possible values are 1 or 3.

propagation Delay	Integer	0..1	aggregation	The propagation time segment in quanta.
syncJump Width	Integer	1	aggregation	The number of quanta in the Synchronization Jump Width, SJW. The (Re-)Synchronization Jump Width (SJW) defines how far a resynchronization may move the Sample Point inside the limits defined by the Phase Buffer Segments to compensate for edge phase errors.
timeSeg1	Integer	1	aggregation	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	aggregation	The number of quanta after the sampling point: timeSeg2 = Phase_Seg2

Table 2.9: CanControllerConfiguration

Class	⟨⟨**atpObject**⟩⟩ **CanControllerConfigurationRequirements**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Class Desc.	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 Class(es)	AbstractCanCommunicationControllerAttributes			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
maxNum-berOfTime QuantaPer Bit	Integer	0..1	aggregation	Maximum number of time quanta in the bit time.
maxSam-plePoint	Float	0..1	aggregation	The max. value of the sample point as a percentage of the total bit time.
maxSync JumpWidth	Float	0..1	aggregation	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.
minNum-berOfTime QuantaPer Bit	Integer	0..1	aggregation	Minimum number of time quanta in the bit time.
minSample Point	Float	0..1	aggregation	The min. value of the sample point as a percentage of the total bit time.

minSync JumpWidth	Float	0..1	aggregation	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 2.10: CanControllerConfigurationRequirements

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).

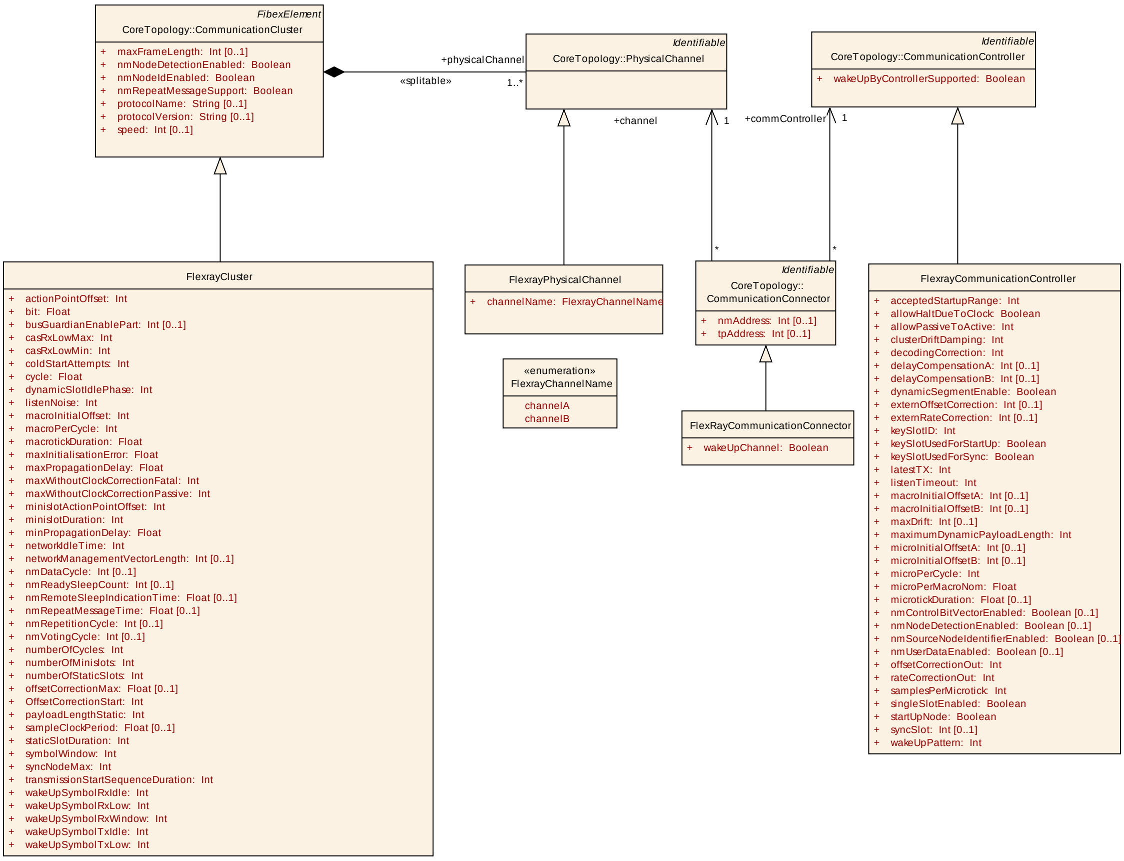

Figure 2.5: FlexRay cluster elements (Fibex4FlexRay_Topology)

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	⟨⟨**atpObject**⟩⟩ **FlexrayCluster**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::FlexrayTopology			
Class Desc.	FlexRay specific attributes to the physicalCluster			
Base Class(es)	CommunicationCluster			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***

Offset Correction Start	Integer	1	aggregation	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
actionPoint Offset	Integer	1	aggregation	The offset of the action point in networks
bit	Float	1	aggregation	Nominal bit time (= 1 / fx:SPEED). gdBit = cSamplesPerBit * gdSampleClockPeriod. Unit: seconds (gdBit)
bus Guardian EnablePart	Integer	0..1	aggregation	Bus Guardian Inter Slot Gap (ISG) part that follows a guarded schedule element. Unit macroticks
casRxLow Max	Integer	1	aggregation	Upper limit of the Collision Avoidance Symbol (CAS) acceptance window. Unit:bitDuration
casRxLow Min	Integer	1	aggregation	Lower limit of the Collision Avoidance Symbol (CAS) acceptance window. Unit:bitDuration
coldStart Attempts	Integer	1	aggregation	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	aggregation	Length of the cycle. Unit: seconds
dynamic SlotIdle Phase	Integer	1	aggregation	The duration of the dynamic slot idle phase in minislots.
listenNoise	Integer	1	aggregation	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
macro InitialOffset	Integer	1	aggregation	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
macroPer Cycle	Integer	1	aggregation	The number of macroticks in a communication cycle
macrotick Duration	Float	1	aggregation	Duration of the cluster wide nominal macrotick, expressed in seconds
maxInitiali-sationError	Float	1	aggregation	The maximum error that a node may have after initialization. Unit: seconds
maxProp-agation Delay	Float	1	aggregation	Maximum propagation delay of a Cluster (in seconds).

maxWith-outClock Correction Fatal	Integer	1	aggregation	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.
maxWith-outClock Correction Passive	Integer	1	aggregation	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.
minProp-agation Delay	Float	1	aggregation	Minimum propagation delay of a Cluster (in seconds).
minislotAc-tionPoint Offset	Integer	1	aggregation	The Offset of the action point within a minislot. Unit: macroticks
minislot Duration	Integer	1	aggregation	The duration of a minislot (dynamic segment). Unit: macroticks.
network IdleTime	Integer	1	aggregation	The duration of the network idle time in macroticks
network Man-agement Vector Length	Integer	0..1	aggregation	Length of the Network Management vector on a cluster. Unit: Bytes
nmData Cycle	Integer	0..1	aggregation	Number of FlexRay Communication Cycles needed to transmit the Nm Data PDUs of all FlexRay Nm Ecus of this FlexRayNmCluster.
nmReady Sleep Count	Integer	0..1	aggregation	Numbers of repetitions in the ready sleep state before NM switches to bus sleep mode. On a value of ”1”, the NM-State Machine will leave the Ready Sleep State after one NM Repetition Cycle with no ”keep awake” votes.
nmRemote SleepIndi-cationTime	Float	0..1	aggregation	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.
nmRepeat Message Time	Float	0..1	aggregation	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	aggregation	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.
nmVoting Cycle	Integer	0..1	aggregation	Number of FlexRay CommunicationCycles needed to transmit the Nm vote of Pdus of all FlexRay NmEcus of this FlexRayNmCluster.
numberOf Cycles	Integer	1	aggregation	Total number of cycles until a temporal transmission pattern is repeated. The CycleCounter of an AbsolutelyScheduledTiming is evaluated against this parameter.
numberOf Minislots	Integer	1	aggregation	number of Minislots in the dynamic segment.
numberOf StaticSlots	Integer	1	aggregation	The number of static slots in the static segment.
offsetCorrectionMax	Float	0..1	aggregation	Cluster global magnitude of the maximum permissible offset correction value Unit: seconds (gOffsetCorrectionMax)
payload Length Static	Integer	1	aggregation	Globally configured payload length of a static frame. Unit: 16-bit WORDS.
sample ClockPeriod	Float	0..1	aggregation	Sample clock period. Unit: seconds
staticSlot Duration	Integer	1	aggregation	The duration of a slot in the static segment. Unit: macroticks
symbol Window	Integer	1	aggregation	The duration of the symbol window. Unit: macroticks
syncNode Max	Integer	1	aggregation	The maximum number of sync nodes allowed in the cluster
transmission StartSequence Duration	Integer	1	aggregation	Number of bits in the Transmission Start Sequence [gdBits].
wakeUp SymbolRx Idle	Integer	1	aggregation	Number of bits used by the node to test the duration of the idle portion of a received wake up symbol. Unit:bitDuration
wakeUp SymbolRx Low	Integer	1	aggregation	Number of bits used by the node to test the LOW portion of a received wake up symbol. Unit:bitDuration

wakeUp SymbolRx Window	Integer	1	aggregation	Number of bits used by a node to test the overall duration of a received wake up symbol. Unit: gdBit
wakeUp SymbolTx Idle	Integer	1	aggregation	Number of bits used by the node to transmit the idle part of a wake up symbol. Unit: gDbit
wakeUp SymbolTx Low	Integer	1	aggregation	Number of bits used by the node to transmit the LOW part of a wake up symbol. Unit:bitDuration

Table 2.11: FlexrayCluster

2.3.2.2 FlexRay Communication Controller

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

Class	⟨⟨**atpObject**⟩⟩ **FlexrayCommunicationController**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::FlexrayTopology			
Class Desc.	FlexRay bus specific communication port attributes.			
Base Class(es)	CommunicationController			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
accepted Startup Range	Integer	1	aggregation	Expanded range of measured clock deviation allowed for startup frames during integration. Unit:microtick
allowHalt DueTo Clock	Boolean	1	aggregation	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).
allowPas-siveTo Active	Integer	1	aggregation	Number of consecutive even/odd cycle pairs that must have valid clock correction terms before the Communication Controller will be allowed to transition from the POC:normal passive state to POC:normal active state. If set to 0, the Communication Controller is not allowed to transition from POC:norm
clusterDrift Damping	Integer	1	aggregation	The cluster drift damping factor used in clock synchronization rate correction in microticks

decoding Correction	Integer	1	aggregation	Value used by the receiver to calculate the difference between primary time reference point and secondary time reference point. Unit: Microticks (pDecodingCorrection)
delayCom-pensationA	Integer	0..1	aggregation	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.
delayCom-pensationB	Integer	0..1	aggregation	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.
dynamic Segment Enable	Boolean	1	aggregation	Boolean flag that configures the Bus Guardian Schedule Monitoring Service to expect transmissions within the dynamic segment.
extern OffsetCor-rection	Integer	0..1	aggregation	Fixed amount added or subtracted to the calculated offset correction term to facilitate external offset correction, expressed in node-local microticks.
externRate Correction	Integer	0..1	aggregation	Fixed amount added or subtracted to the calculated rate correction term to facilitate external rate correction, expressed in node-local microticks.
keySlotID	Integer	1	aggregation	ID of the slot used to transmit the startup frame, sync frame, or designated single slot frame.
keySlot UsedFor StartUp	Boolean	1	aggregation	Flag indicating whether the Key Slot is used to transmit a startup frame.
keySlot UsedFor Sync	Boolean	1	aggregation	Flag indicating whether the Key Slot is used to transmit a sync frame.
latestTX	Integer	1	aggregation	The number of the last minislot in which a transmission can start in the dynamic segment for the respective node
listenTime-out	Integer	1	aggregation	Upper limit for the start up listen timeout and wake up listen timeout. Unit: Microticks
macroIni-tialOffsetA	Integer	0..1	aggregation	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	aggregation	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.
maxDrift	Integer	0..1	aggregation	Maximum drift offset in microticks between two nodes that operate with unsynchronized clocks over one communication cycle.
maximum Dynamic Payload Length	Integer	1	aggregation	Maximum payload length for the dynamic channel of a frame in 16 bit WORDS.
microInitial OffsetA	Integer	0..1	aggregation	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.
microInitial OffsetB	Integer	0..1	aggregation	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.
microPer Cycle	Integer	1	aggregation	The nominal number of microticks in a communication cycle
microPer MacroNom	Float	1	aggregation	Number of microticks per nominal macrotick that all implementations must support.
microtick Duration	Float	0..1	aggregation	Duration of a microtick. This attribute can be derived from samplePerMicrotick and gdSampleClockPeriod. Unit: seconds
nmControl BitVector Enabled	Boolean	0..1	aggregation	Enables control bit vector support.
nmNode Detection Enabled	Boolean	0..1	aggregation	Enables the Request Repeat Message Request support. Only valid if nmNodeIdEnabled is set to true.

nmSource Node Identifier Enabled	Boolean	0..1	aggregation	Switch for enabling SourceNodeIdentifier support.
nmUser DataEn-abled	Boolean	0..1	aggregation	Switch for enabling user data support.
offsetCor-rectionOut	Integer	1	aggregation	Magnitude of the maximum permissible offset correction value. Unit:microtick (pOffsetCorrectionOut)
rateCor-rectionOut	Integer	1	aggregation	Magnitude of the maximum permissible rate correction value. Unit:Microticks (pRateCorrectionOut)
samples PerMi-crotick	Integer	1	aggregation	Number of samples per microtick
singleSlot Enabled	Boolean	1	aggregation	Flag indicating whether or not the node shall enter single slot mode following startup.
startUp Node	Boolean	1	aggregation	Indicates that the node is a startup node (startup frame configured; connected to gChannels)
syncSlot	Integer	0..1	aggregation	The number of the static slot in which a sync frame shall be sent, if a sync frame shall be sent
wakeUp Pattern	Integer	1	aggregation	Number of repetitions of the Tx-wakeup symbol to be sent during the CC_WakeupSend state of this Node in the cluster

Table 2.12: FlexrayCommunicationController

2.3.2.3 FlexRay Communication Connector

`FlexrayCommunicationConnector` adds the FlexRay specific attributes to the `CommunicationConnector`.

Class	⟨⟨**atpObject**⟩⟩ **FlexRayCommunicationConnector**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::FlexrayTopology			
Class Desc.	FlexRay specific attributes to the CommunicationConnector			
Base Class(es)	CommunicationConnector			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
wakeUp Channel	Boolean	1	aggregation	Referenced channel used by the node to send a wakeup pattern. (pWakeupChannel)

Table 2.13: FlexRayCommunicationConnector

2.3.2.4 FlexRay Physical Channel

`FlexrayPhysicalChannel` adds the FlexRay specific attributes to the `PhysicalChannel`.

Class	⟨⟨**atpObject**⟩⟩ **FlexrayPhysicalChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::FlexrayTopology			
Class Desc.	FlexRay specific attributes to the physicalChannel			
Base Class(es)	PhysicalChannel			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
channel Name	Flexray Channel Name	1	aggregation	Name of the channel (Channel A or Channel B).

Table 2.14: FlexrayPhysicalChannel

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

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`.

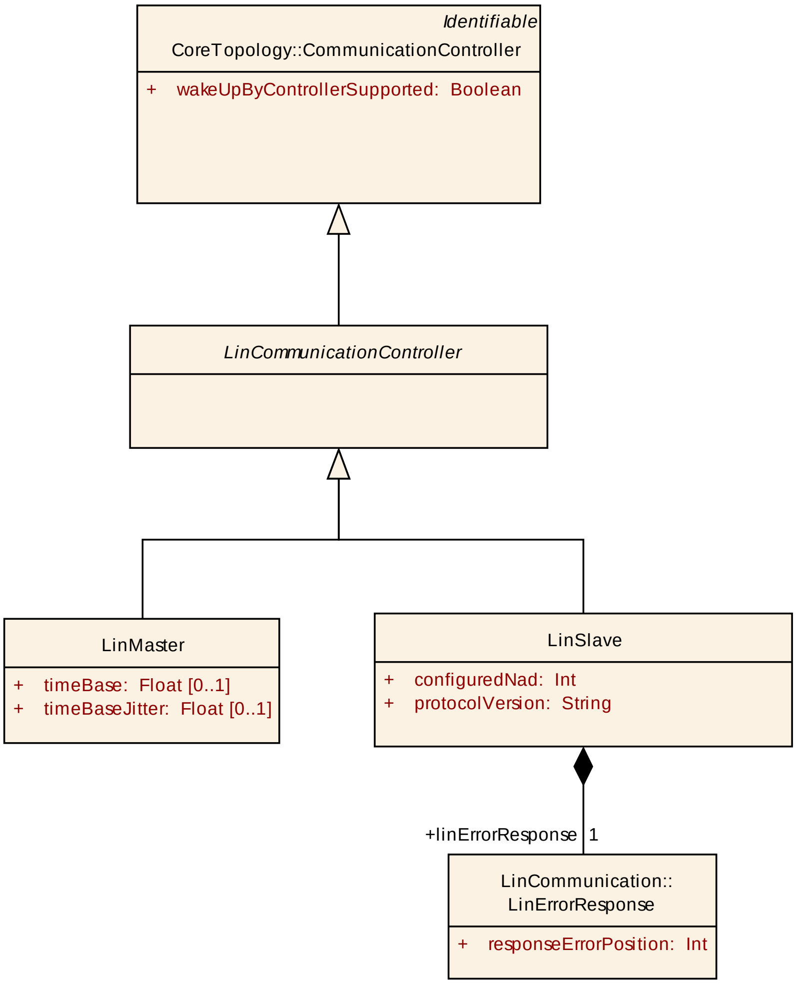

Figure 2.6: Specialized `LINCommunicationController` attributes (Fibex4Lin_Topology)

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	⟨⟨atpStructureElement⟩⟩ LinCluster			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Class Desc.	LIN specific attributes			
Base Class(es)	CommunicationCluster			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
scheduleTable	LinScheduleTable	*	aggregation	Schedule tables organize the Timings of the frames for LIN.

Table 2.15: 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	⟨⟨atpObject⟩⟩ LinCommunicationController (abstract)			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Class Desc.	LIN bus specific communication port instance attributes.			
Base Class(es)	CommunicationController			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**

Table 2.16: 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	⟨⟨atpObject⟩⟩ LinMaster			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Class Desc.	Describing the properties of the refering ecu as a LIN master.			
Base Class(es)	LinCommunicationController			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**

timeBase	Float	0..1	aggregation	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.
timeBase Jitter	Float	0..1	aggregation	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 2.17: LinMaster

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	⟨⟨**atpObject**⟩⟩ **LinSlave**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Class Desc.	Describing the properties of the refering ecu as a LIN slave.			
Base Class(es)	LinCommunicationController			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
configured Nad	Integer	1	aggregation	To distinguish LIN slaves that are used twice or more within the same cluster.
linError Response	LinError Response	1	aggregation	Each slave node shall publish one response error in one of its transmitted unconditional frames.
protocol Version	String	1	aggregation	Version specifier for a communication protocol.

Table 2.18: LinSlave

Class	⟨⟨**atpObject**⟩⟩ **LinErrorResponse**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Class Desc.	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 Class(es)	ARObject			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
frameTriggering	LinFrame Triggering	1	reference	Reference to an unconditional frame that transmits the response error. The referenced LinFrameTriggering shall contain a reference to an unconditionalFrame.
response ErrorPosition	Integer	1	aggregation	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 2.19: 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).

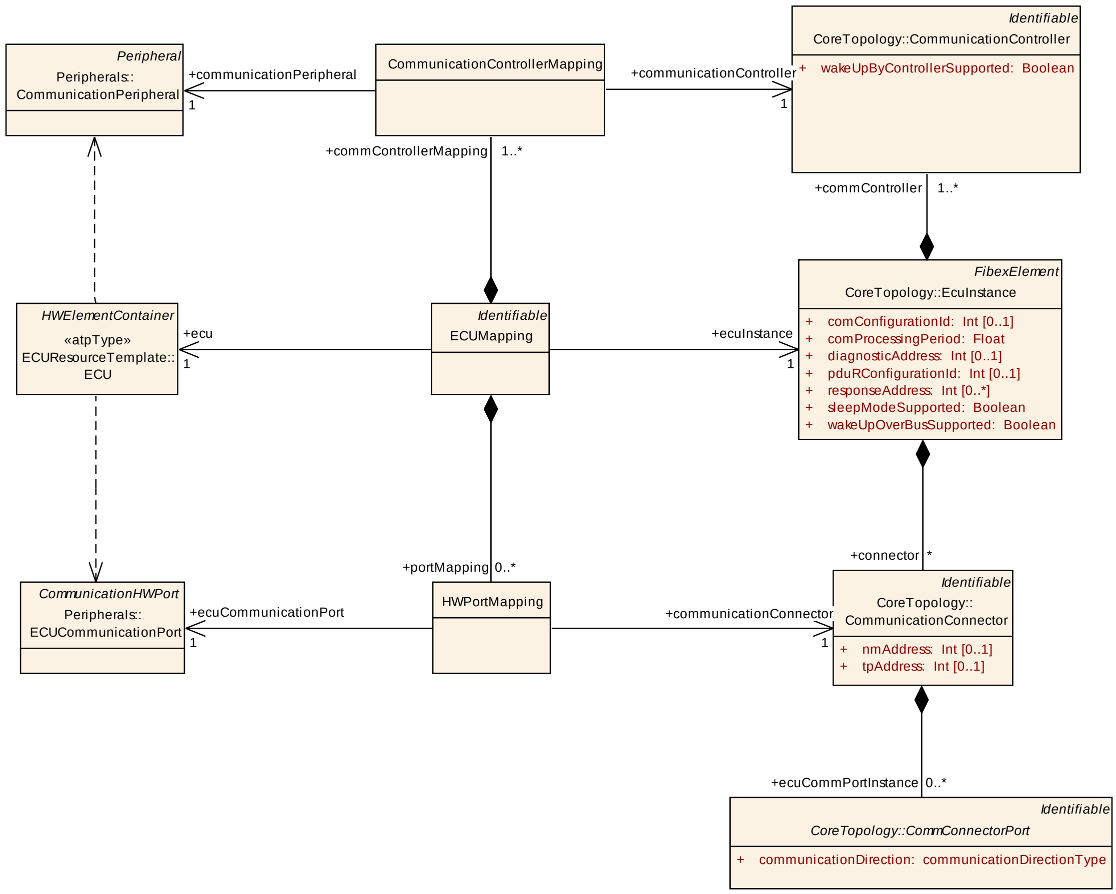

Figure 2.7: Mapping of topology description elements in the System Template onto hardware elements defined in the ECU Resource Template (ECUResourceMapping)

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	⟨⟨atpObject⟩⟩ **ECUMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::ECUResourceMapping			
Class Desc.	ECUMapping allows to assign an ECU hardware type (defined in the ECU Resource Template) to an ECUInstance used in a physical topology.			
Base Class(es)	Identifiable			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
comm Controller Mapping	Comm unication Controller Mapping	1..*	aggregation	The ECUMapping contains the mapping of all CommunicationControllers of the ECU.
ecu	ECU	1	reference	Reference to the Ecu description in the ECU Resource Template
ecuIn-stance	EcuIn-stance	1	reference	Reference to the EcuInstance in the System Template
portMap-ping	HWPort Mapping	*	aggregation	The ECUMapping contains the mapping of all HW Ports of the ECU.

Table 2.20: 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	⟨⟨**atpObject**⟩⟩ **CommunicationControllerMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::ECUResourceMapping			
Class Desc.	CommunicationControllerMapping specifies the CommunicationPeripheral hardware (defined in the ECU Resource Template) to realize the specified CommunicationController in a physical topology.			
Base Class(es)	ARObject			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
communicationController	CommunicationController	1	reference	Reference to the CommunicationController in the System Template
communicationPeripheral	CommunicationPeripheral	1	reference	

Table 2.21: 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	⟨⟨**atpObject**⟩⟩ **HWPortMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::ECUResourceMapping			
Class Desc.	HWPortMapping specifies the ECUCommunicationPort hardware (defined in the ECU Resource Template) to realize the specified CommunicationConnector in a physical topology.			
Base Class(es)	ARObject			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
communicationConnector	CommunicationConnector	1	reference	Reference to the CommunicationConnector in the System Template
ecuCommunicationPort	ECUCommunicationPort	1	reference	Reference to the Peripheral in the ECU Resource Template

Table 2.22: 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.

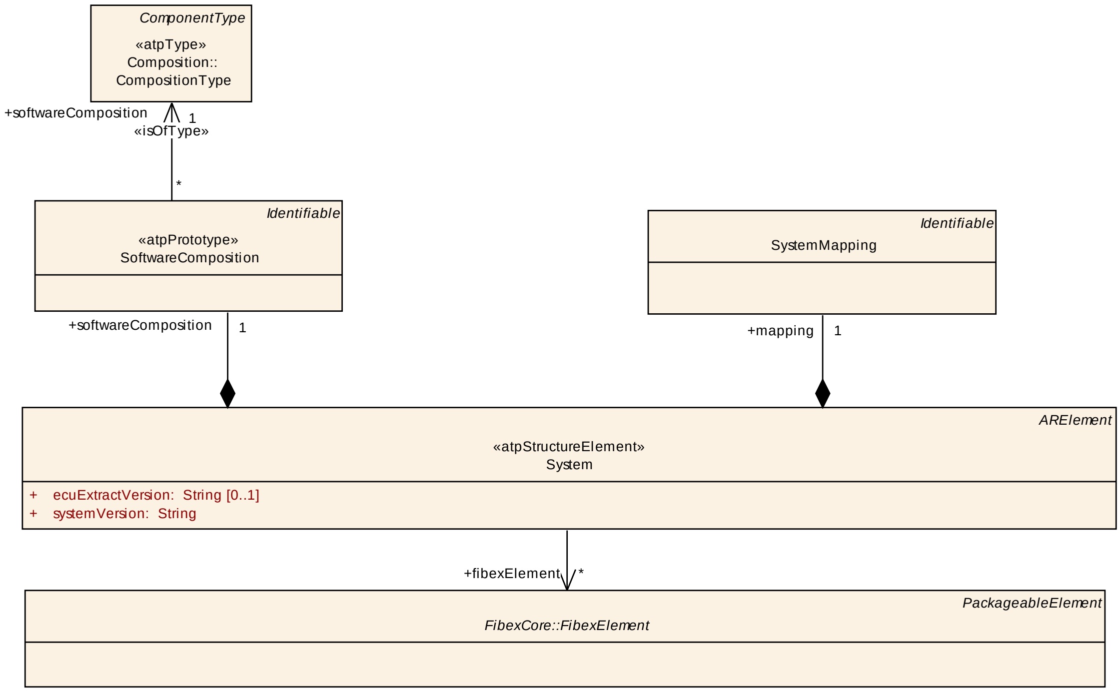

Figure 3.1: Inclusion of a (top-level) Software Composition into an AUTOSAR system (SystemTemplate)

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. 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`, `PortInterfaces`, `DataElementPrototypes`, `InternalBehavior` etc.

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	⟨⟨**atpPrototype**⟩⟩ **SoftwareComposition**			
Package	M2::AUTOSARTemplates::SystemTemplate			
Class Desc.	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 Class(es)	Identifiable			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
software Composi-tion	Composition Type	1	reference to type	We assume that there is exactly one top-level composition that includes all Component instances of the system

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.

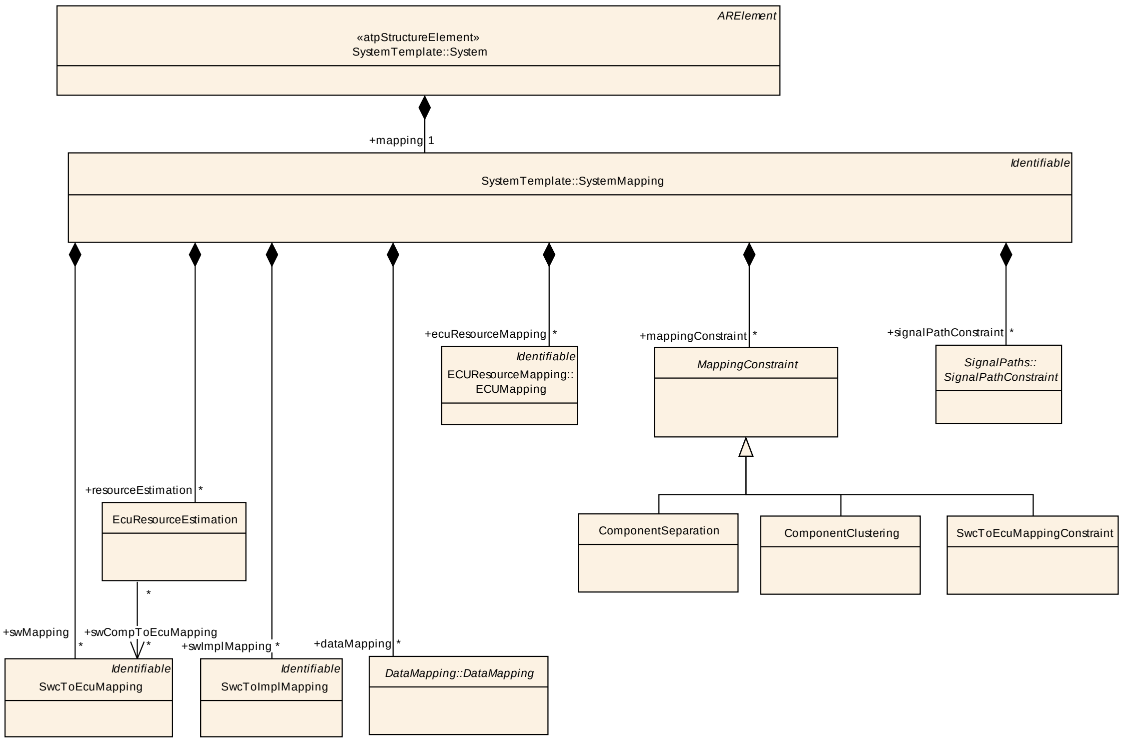

Figure 4.1: Mapping Overview (Mapping)

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	⟨⟨**atpObject**⟩⟩ **SystemMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate			
Class Desc.	The system mapping aggregates all mapping aspects (mapping of SW components to ECUs, mapping of data elements to signals, and mapping constraints).			
Base Class(es)	Identifiable			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
dataMapping	DataMapping	*	aggregation	The data mappings defined.
ecuResource Mapping	ECUMapping	*	aggregation	
mapping Constraint	Mapping Constraint	*	aggregation	Constraints that limit the mapping freedom for the mapping of SW components to ECUs.
resource Estimation	EcuResource Estimation	*	aggregation	Resource estimations for this set of mappings, zero or one per ECU instance.
signalPath Constraint	SignalPath Constraint	*	aggregation	Constraints that limit the mapping freedom for the mapping of data elements to signals.
swImpl Mapping	SwcToImpl Mapping	*	aggregation	The mappings of AtomicSoftwareComponent Instances to Implementations.
swMapping	SwcToEcu Mapping	*	aggregation	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.

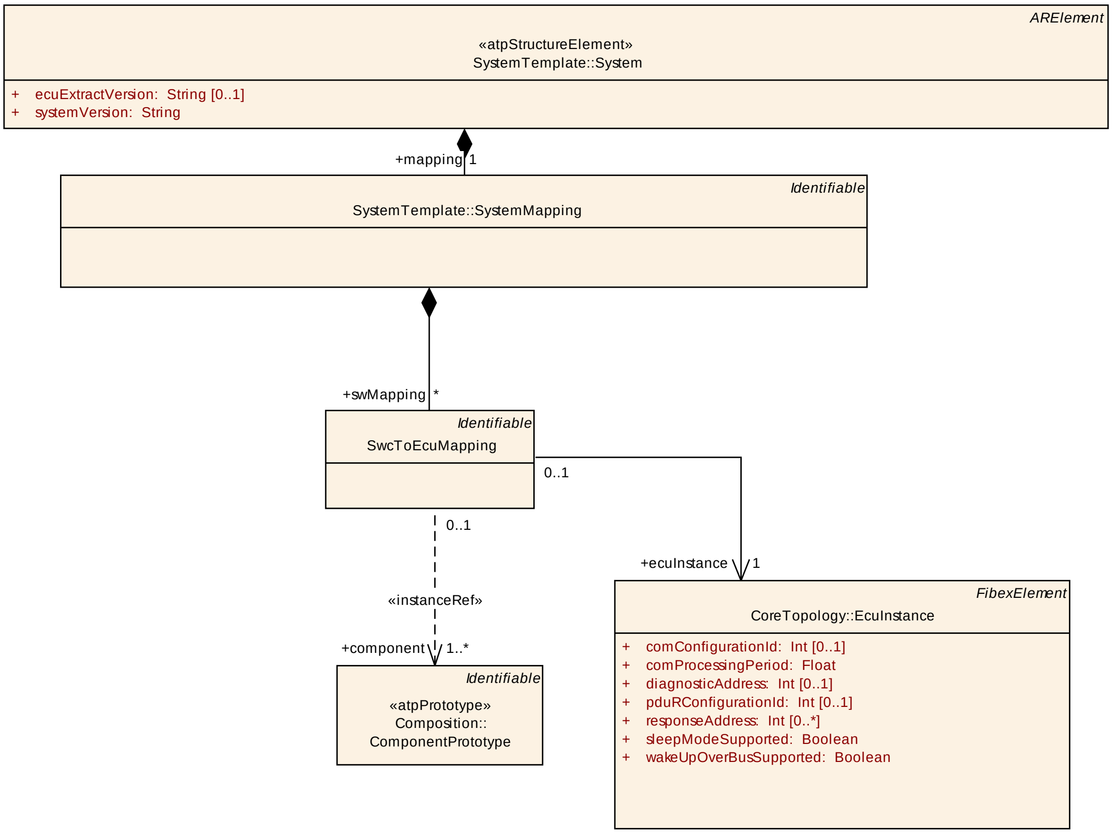

Figure 4.2: SW component to ECU mapping (SwcToEcuMapping)

The following table describes the `SwcToEcuMapping` in detail.

Class	**⟨⟨atpObject⟩⟩ SwcToEcuMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Class Desc.	Map software components to a specific ECU Instance.			
Base Class(es)	Identifiable			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
component	Component Prototype	1..*	instanceRef	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.
ecuIn-stance	EcuIn-stance	1	reference	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.

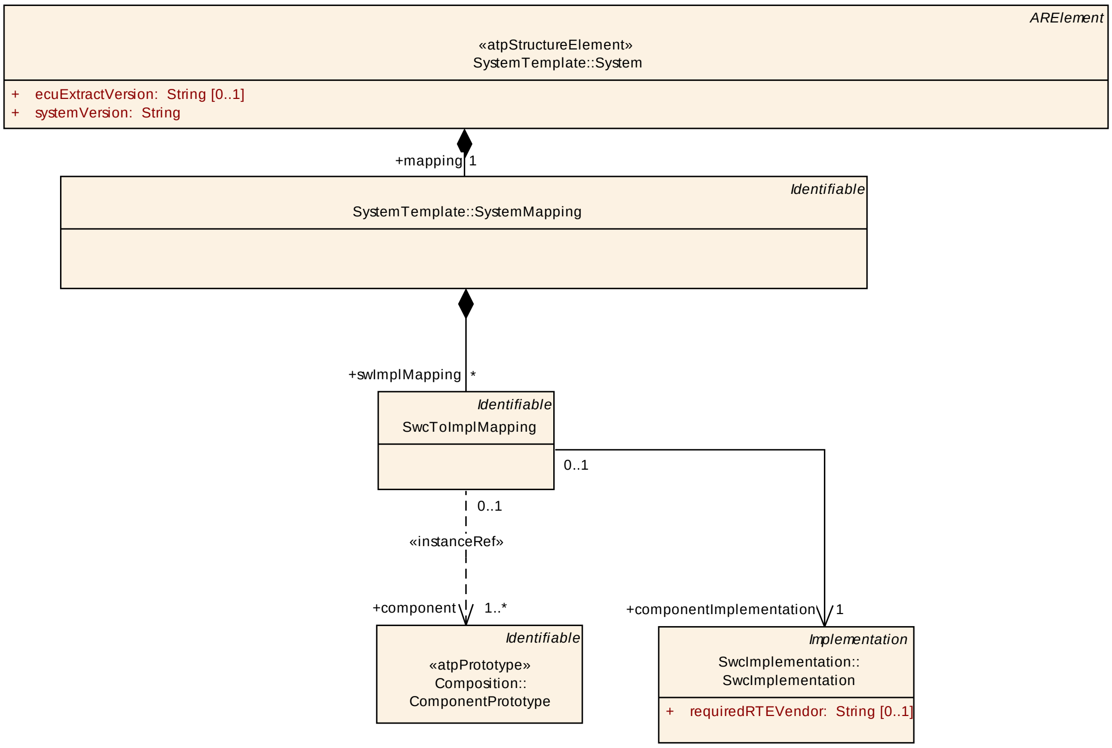

Figure 4.3: SW Component to Implementation mapping (SwcToImplMapping)

The following table contains the detailed description of SwcToImplMapping:

Class	⟨⟨atpObject⟩⟩ SwcToImplMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Class Desc.	Map instances of an AtomicSoftwareComponentType to a specific Implementation.			
Base Class(es)	Identifiable			
Attribute	Datatype	Mul.	Link Type	Description
component	Component Prototype	1..*	instanceRef	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.
component Implementation	SwcImplementation	1	reference	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	⟨⟨atpObject⟩⟩ MappingConstraint (abstract)			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Class Desc.	Different constraints that may be used to limit the mapping of SW components to ECUs.			
Base Class(es)	ARObject			
Attribute	Datatype	Mul.	Link Type	Description

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`.

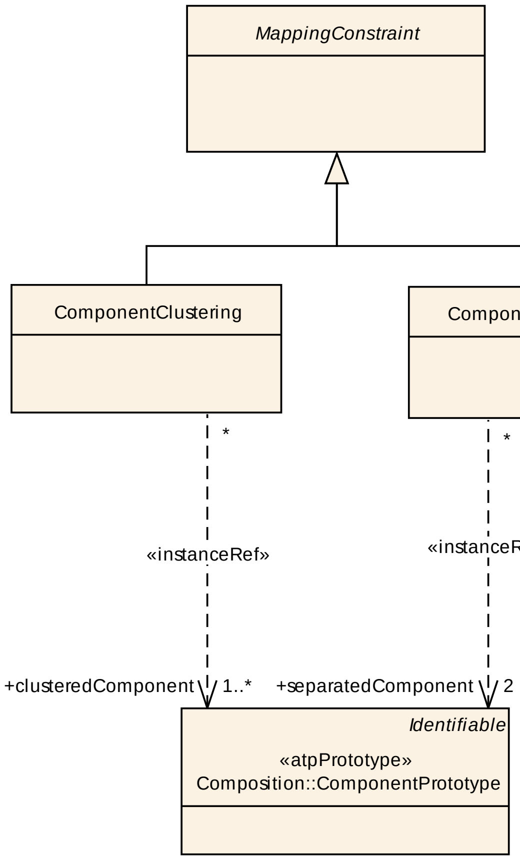

Figure 4.4: Details on ComponentClustering and ComponentSeparation (SwcClustering)

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	⟨⟨**atpObject**⟩⟩ **ComponentClustering**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Class Desc.	Constraint that forces the mapping of all referenced SW component instances to the same ECU			
Base Class(es)	MappingConstraint			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
clustered Compo-nent	Component Prototype	1..*	instanceRef	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.

[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.

Class	⟨⟨**atpObject**⟩⟩ **ComponentSeparation**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Class Desc.	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 Class(es)	MappingConstraint			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
separated Compo-nent	Component Prototype	2	instanceRef	The two components that have to be mapped to different ECUs

Table 4.6: ComponentSeparation

4.1.3.3 SwcToEcuMappingConstraint

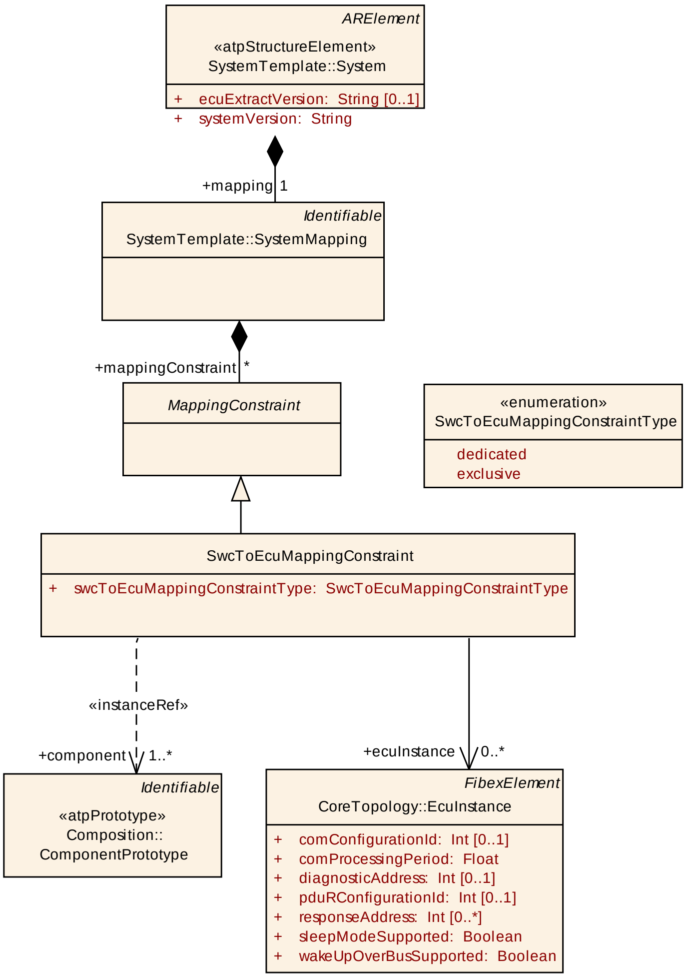

Figure 4.5: Dedicated and exclusive Mapping of SWC to ECUs

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

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

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

Class	⟨⟨**atpObject**⟩⟩ **SwcToEcuMappingConstraint**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Class Desc.	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 Class(es)	MappingConstraint			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
component	Component Prototype	1..*	instanceRef	
ecuIn-stance	EcuIn-stance	*	reference	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.
swcToEcu Mapping Constraint Type	SwcToEcu Mapping Constraint Type	1	aggregation	This attribute determines if dedicated or exclusive mapping is used.

Table 4.7: SwcToEcuMappingConstraint

Enumeration	**SwcToEcuMappingConstraintType**
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping
Enum Desc.	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.

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`.

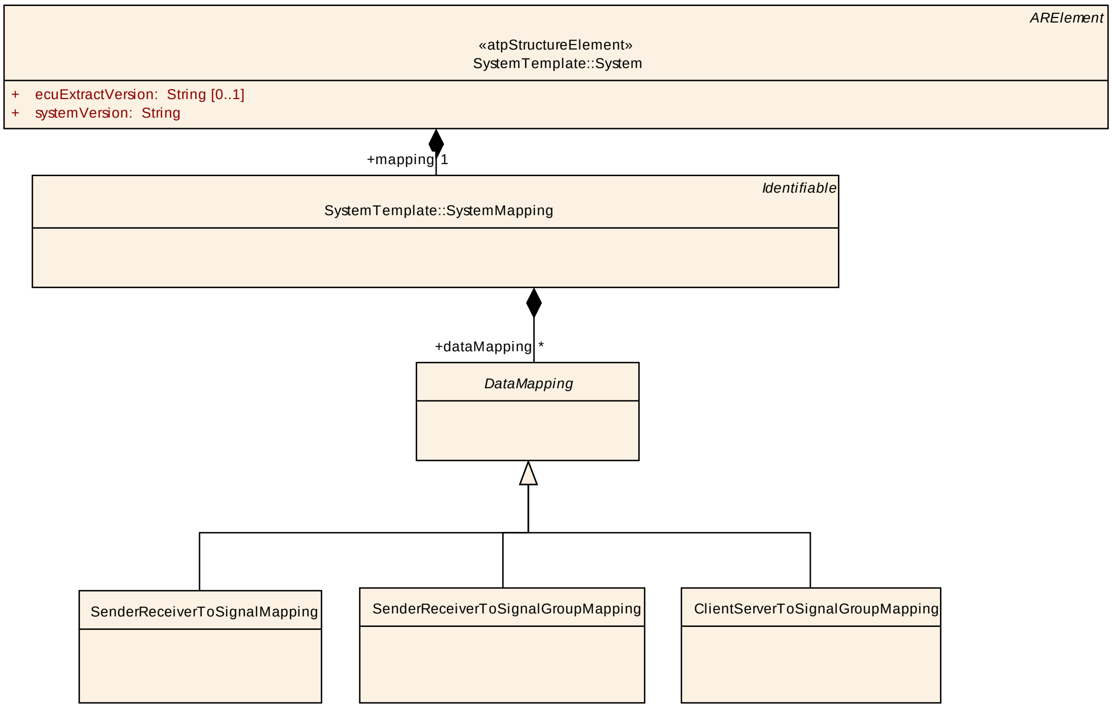

Figure 4.6: Overview: Data Mapping Description (DataMappingOverview)

`SystemSignals` represent `DataElementPrototypes` and `OperationPrototypes` in the communication description. The `SystemSignals` are unique per System and can be defined independently of frames and communication clusters. 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	⟨⟨**atpObject**⟩⟩ **SystemSignal**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	AbstractSignal			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
dataType	Primitive Type	0..1	reference	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	Value Specification	0..1	reference	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	aggregation	Size of the signal in bits.

Table 4.8: SystemSignal

In case that a `DataElementPrototype` is transferred over the network a `SystemSignal` is being defined representing the `DataElementPrototype` on the network.

SystemSignals are unique in the sense that the same SystemSignal represents the same DataElementPrototype system wide.

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

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

2) 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.

3) 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 4.7).

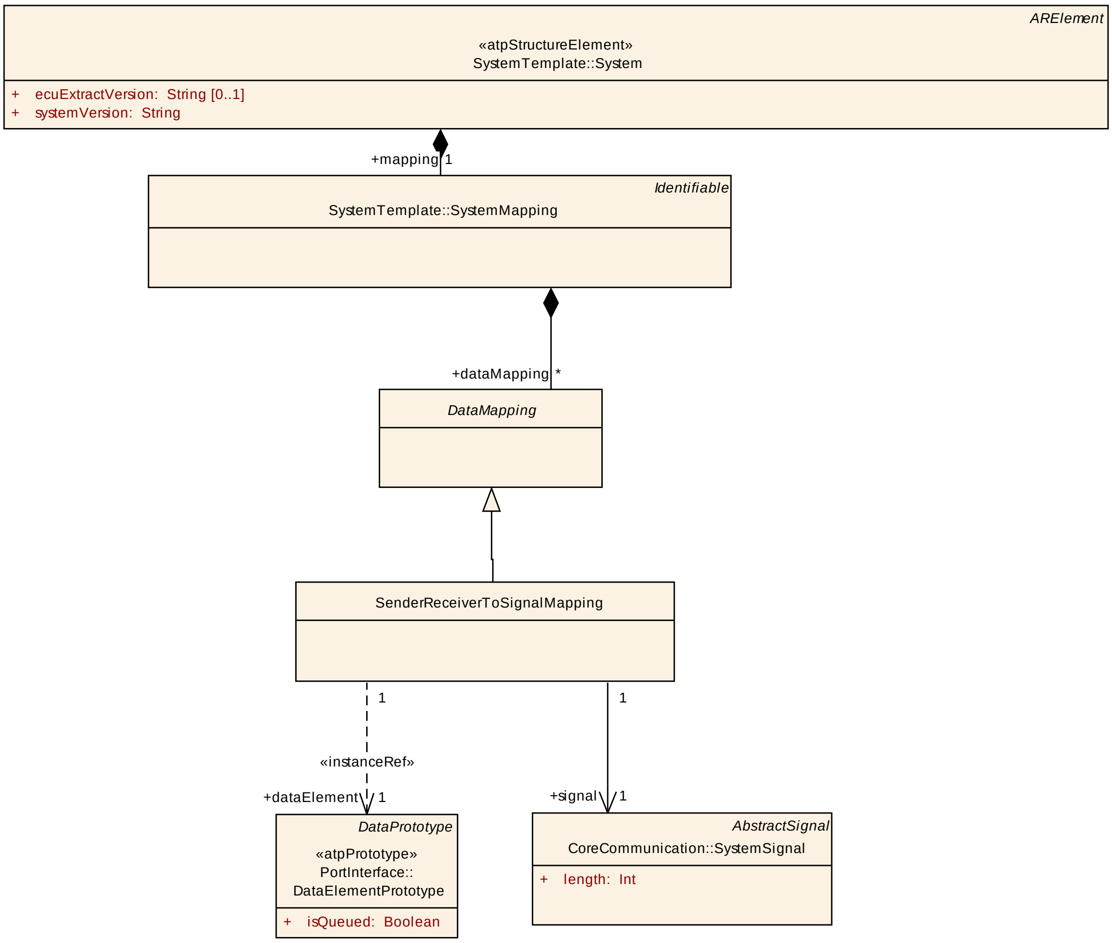

Figure 4.7: Mapping of data elements with primitive datatypes (SenderRecPrimitiveTypeMapping)

Class	⟨⟨**atpObject**⟩⟩ **SenderReceiverToSignalMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	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 Class(es)	DataMapping			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
dataElement	DataElement Prototype	1	instanceRef	Reference to the data element, which ought to be sent over the Communication bus. This DataElement is described in the Software Component Template.
signal	System Signal	1	reference	Reference to the system signal used to carry the data element.

Table 4.9: 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 atomically. To achieve this, the "signal group" mechanisms shall be utilized. The complex data type must be decomposed into single signals. As this set of single signals has to be treated as atomic, it is placed in a "signal group".

Thus, each PrimitiveType will be one SystemSignal in the System Description. For a CompositeType several SystemSignals will be used. The relationship between the SystemSignals and the DataElementPrototypes is provided in the SenderReceiverToSignalGroupMapping (see Figure 4.8).

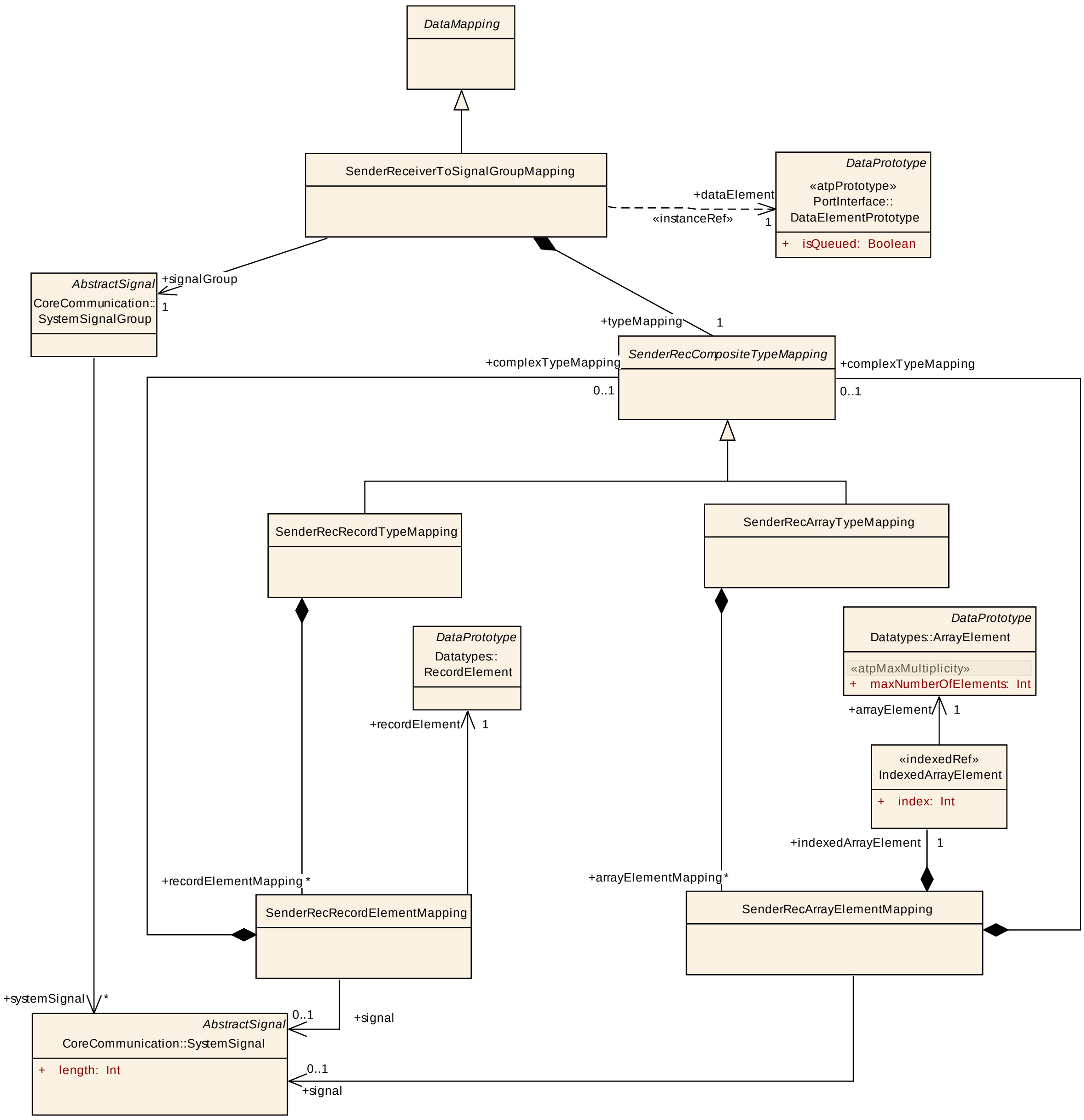

Figure 4.8: Mapping of data elements with composite datatypes (SenderRecComposite-TypeMapping)

Class	⟨⟨**atpObject**⟩⟩ **SenderReceiverToSignalGroupMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	Mapping of a sender receiver communication data element with a composite datatype to a signal group.			
Base Class(es)	DataMapping			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
dataElement	DataElementPrototype	1	instanceRef	Reference to the data element, which ought to be sent over the Communication bus.
signalGroup	SystemSignalGroup	1	reference	Reference to the signal group, which contain all primitive datatypes of the composite type
typeMapping	SenderRecCompositeTypeMapping	1	aggregation	

Table 4.10: SenderReceiverToSignalGroupMapping

Class	⟨⟨**atpObject**⟩⟩ **SenderRecCompositeTypeMapping (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***

Table 4.11: SenderRecCompositeTypeMapping

Class	⟨⟨atpObject⟩⟩ SenderRecArrayTypeMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	If the compositeType is an Array, the ”ArrayTypeMapping” will be used.			
Base Class(es)	SenderRecCompositeTypeMapping			
Attribute	Datatype	Mul.	Link Type	Description
arrayElement Mapping	Sender RecArray Element Mapping	*	aggregation	Each ArrayElement must be mapped on a SystemSignal.

Table 4.12: SenderRecArrayTypeMapping

Class	⟨⟨atpObject⟩⟩ SenderRecRecordTypeMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	If the compositeType is a Record, the ”RecordTypeMapping” will be used.			
Base Class(es)	SenderRecCompositeTypeMapping			
Attribute	Datatype	Mul.	Link Type	Description
record Element Mapping	Sender RecRecord Element Mapping	*	aggregation	Each RecordElement must be mapped on a SystemSignal.

Table 4.13: SenderRecRecordTypeMapping

Class	⟨⟨atpObject⟩⟩ SenderRecRecordElementMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	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 Class(es)	ARObject			
Attribute	Datatype	Mul.	Link Type	Description
complex TypeMapping	Sender RecCompositeType Mapping	0..1	aggregation	This aggregation will be used if the element is composite.
record Element	Record Element	1	reference	Reference to a RecordElement in the context of the dataElement or in the context of a composite element.
signal	System Signal	0..1	reference	Reference to the system signal used to carry the primitive RecordElement.

Table 4.14: SenderRecRecordElementMapping

Class	⟨⟨atpObject⟩⟩ SenderRecArrayElementMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	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 Class(es)	ARObject			
Attribute	Datatype	Mul.	Link Type	Description
complex TypeMapping	Sender RecCompositeType Mapping	0..1	aggregation	This aggregation will be used if the element is composite.
indexed ArrayElement	Indexed Array Element	1	aggregation	Reference to an indexed array element in the context of the dataElement or in the context of a composite element.
signal	System Signal	0..1	reference	Reference to the system signal used to carry the primitive ArrayElement.

Table 4.15: 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 4.9).

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.

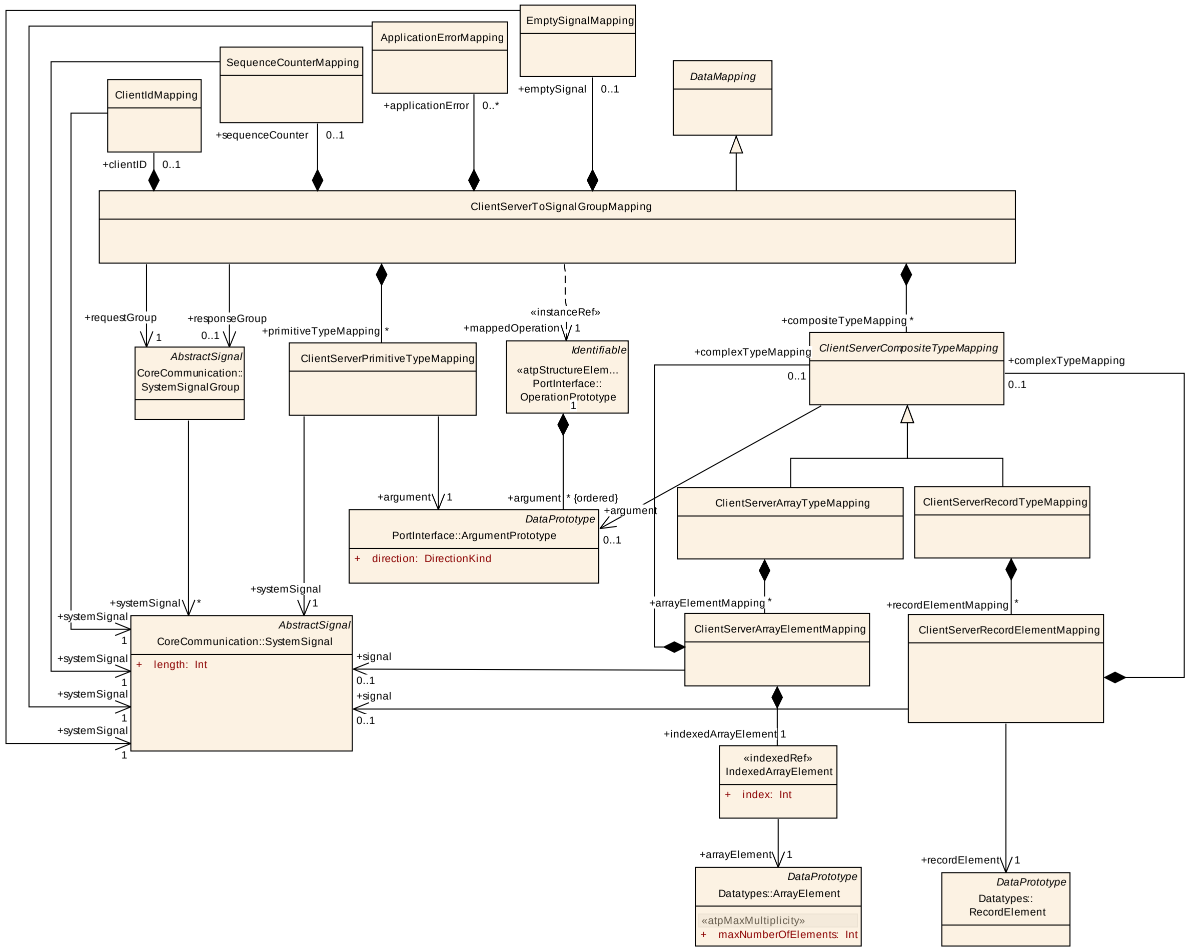

Figure 4.9: Operation Mapping (ClientServerOperationMapping)

Class	⟨⟨atpObject⟩⟩ ClientServerToSignalGroupMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	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 Class(es)	DataMapping			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
applicationError	ApplicationErrorMapping	*	aggregation	In client server communication, the server may return any value within the application error range.
clientID	ClientIdMapping	0..1	aggregation	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	*	aggregation	Mapping of arguments with composite datatypes.
emptySignal	EmptySignalMapping	0..1	aggregation	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	instanceRef	Reference to the operation whose arguments should be transmitted via the communication bus.
primitiveTypeMapping	ClientServerPrimitiveTypeMapping	*	aggregation	Mapping of an argument with a primitive datatype to a signal.
requestGroup	SystemSignalGroup	1	reference	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	reference	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	aggregation	The purpose of sequence counters is to map a response to the correct request of a known client.

Table 4.16: ClientServerToSignalGroupMapping

Class	⟨⟨**atpObject**⟩⟩ **ClientIdMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
system Signal	System Signal	1	reference	Reference to the SystemSignal with the ClientID.

Table 4.17: ClientIdMapping

Class	⟨⟨**atpObject**⟩⟩ **SequenceCounterMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	The purpose of sequence counters is to map a response to the correct request of a known client.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
system Signal	System Signal	1	reference	Reference to the SystemSignal with the SequenceCounter.

Table 4.18: SequenceCounterMapping

Class	⟨⟨**atpObject**⟩⟩ **ApplicationErrorMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	In client server communication, the server may return any value within the application error range.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
system Signal	System Signal	1	reference	Reference to the SystemSignal with the ApplicationError.

Table 4.19: ApplicationErrorMapping

Class	⟨⟨**atpObject**⟩⟩ **EmptySignalMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
system Signal	System Signal	1	reference	Reference to a SystemSignal with "signalLength" = 0 and an UpdateBit.

Table 4.20: EmptySignalMapping

Class	⟨⟨**atpObject**⟩⟩ **ClientServerPrimitiveTypeMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	Mapping of an argument with a primitive datatype to a signal.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
argument	Argument Prototype	1	reference	Reference to an argument in the context of the mappedOperation.
system Signal	System Signal	1	reference	Reference to the system signal used to carry the argument

Table 4.21: ClientServerPrimitiveTypeMapping

Class	⟨⟨**atpObject**⟩⟩ **ClientServerCompositeTypeMapping (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
argument	Argument Prototype	0..1	reference	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 4.22: ClientServerCompositeTypeMapping

Class	⟨⟨**atpObject**⟩⟩ **ClientServerArrayTypeMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	If the compositeType is an Array, the "ArrayTypeMapping" will be used.			
Base Class(es)	ClientServerCompositeTypeMapping			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
arrayElement Mapping	Client Server ArrayElement Mapping	*	aggregation	Each ArrayElement must be mapped on a SystemSignal.

Table 4.23: ClientServerArrayTypeMapping

Class	⟨⟨**atpObject**⟩⟩ **ClientServerRecordTypeMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	If the compositeType is a Record, the "RecordTypeMapping" will be used.			
Base Class(es)	ClientServerCompositeTypeMapping			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
record Element Mapping	Client Server Record Element Mapping	*	aggregation	Each RecordElement must be mapped on a SystemSignal.

Table 4.24: ClientServerRecordTypeMapping

Class	⟨⟨**atpObject**⟩⟩ **ClientServerArrayElementMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
complex TypeMapping	Client Server CompositeType Mapping	0..1	aggregation	This aggregation will be used if the element is composite.
indexed ArrayElement	Indexed Array Element	1	aggregation	Reference to an indexed array element in the context of the mappedOperation or in the context of a composite element.
signal	System Signal	0..1	reference	Reference to the system signal used to carry the primitive ArrayElement.

Table 4.25: ClientServerArrayElementMapping

Class	⟨⟨**atpObject**⟩⟩ **ClientServerRecordElementMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
complex TypeMapping	Client Server CompositeType Mapping	0..1	aggregation	This aggregation will be used if the element is composite.
record Element	Record Element	1	reference	Reference to a RecordElement in the context of the mappedOperation or in the context of a composite element.
signal	System Signal	0..1	reference	Reference to the system signal used to carry the primitive RecordElement.

Table 4.26: 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 4.10 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.

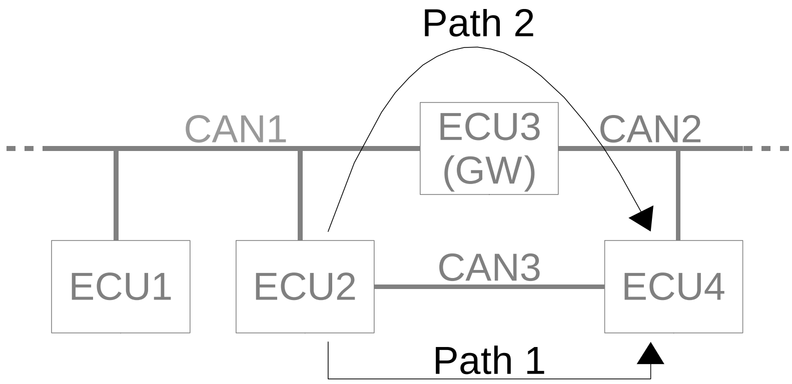

Figure 4.10: Example for a Communication Path

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 4.11):

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.

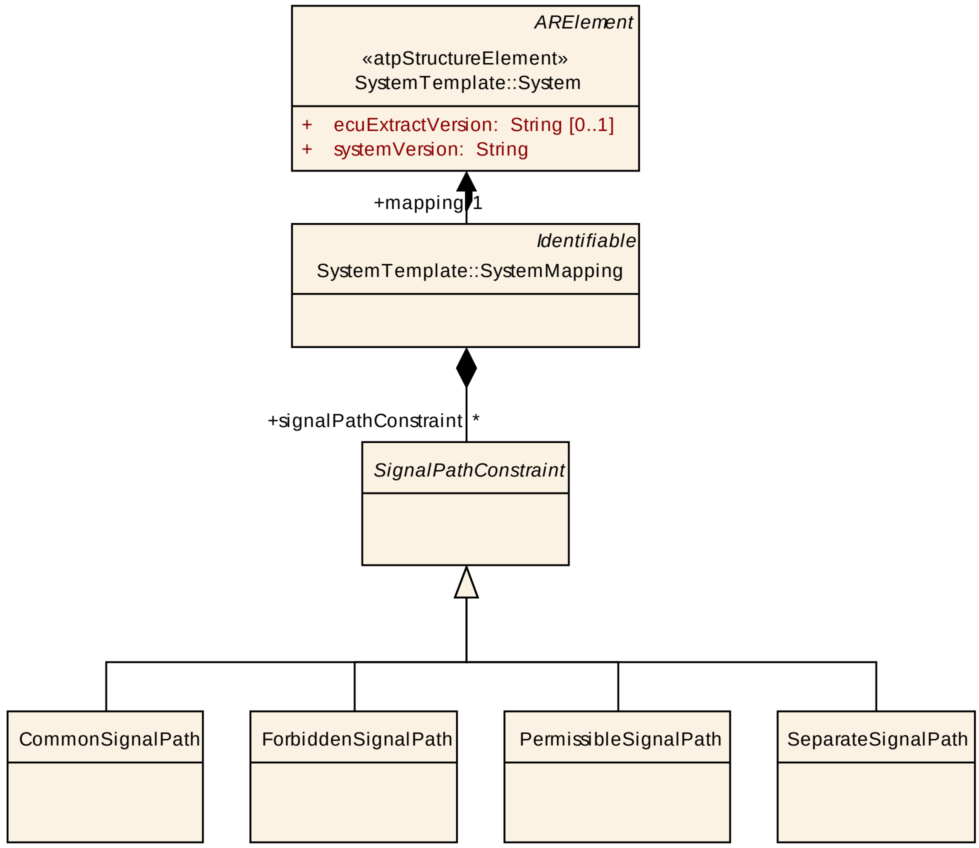

Figure 4.11: Communication Path Description (SignalPathConstraints)

4.2.2.1 CommonSignalPath

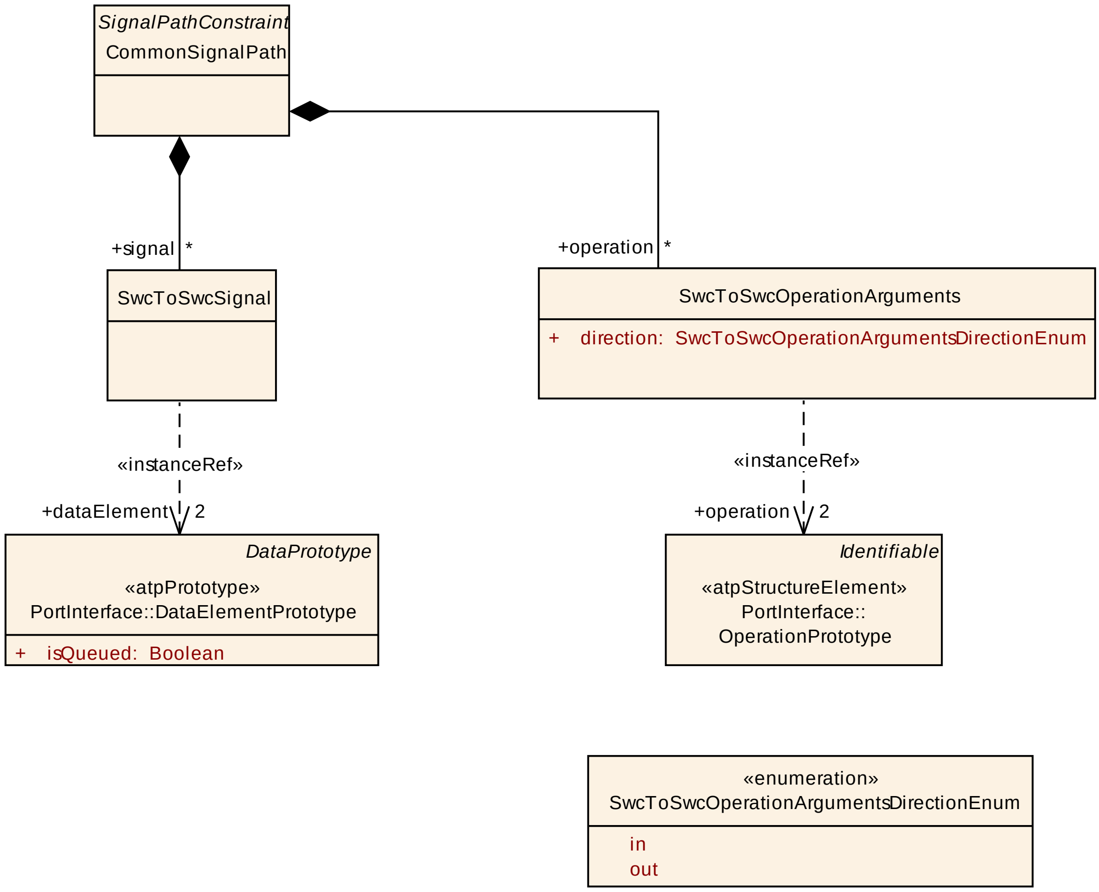

Figure 4.12: Description of signals that must take the same way in the topology (CommonSignalPath)

Class	⟨⟨**atpObject**⟩⟩ **CommonSignalPath**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths			
Class Desc.	The CommonSignalPath describes that two or more SwcToSwcSignals and/or SwcToSwcOperationArguments must take the same way (Signal Path) in the topology.			
Base Class(es)	SignalPathConstraint			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
operation	SwcToSwc Operation Arguments	*	aggregation	
signal	SwcToSwc Signal	*	aggregation	The SwcToSwcSignals that must take the same way (Signal Path) in the topology.

Table 4.27: CommonSignalPath

Class	⟨⟨**atpObject**⟩⟩ **SwcToSwcSignal**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
dataElement	DataElement Prototype	2	instanceRef	Reference to a data element on the PPort and to the same data element on the RPort.

Table 4.28: SwcToSwcSignal

Class	⟨⟨**atpObject**⟩⟩ **SwcToSwcOperationArguments**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
direction	SwcToSwc Operation Arguments Direction Enum	1	aggregation	direction addressed by this SwcToSwcClientServerOperation element.
operation	Operation Prototype	2	instanceRef	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 4.29: SwcToSwcOperationArguments

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

4.2.2.2 ForbiddenSignalPath

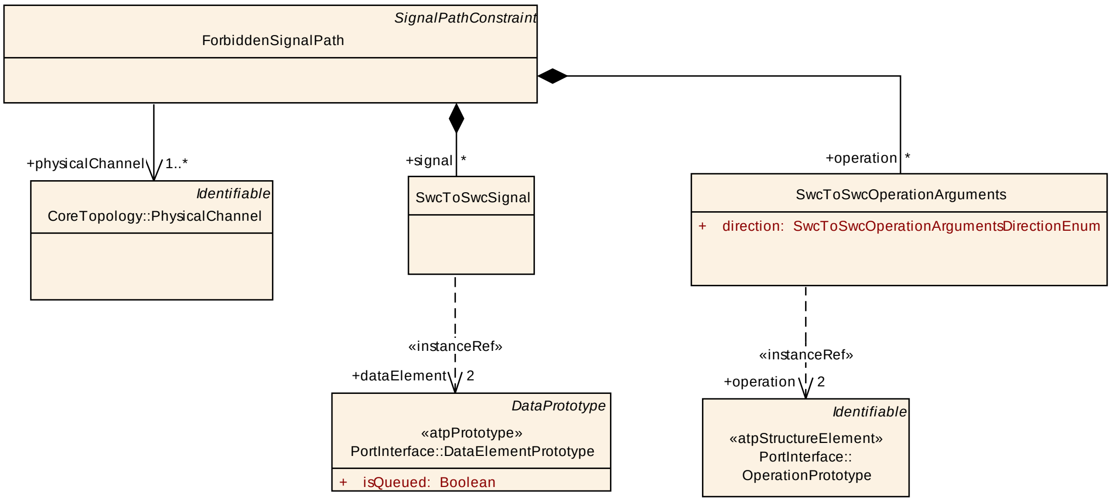

Figure 4.13: Description of the signal path that a signal must not take in the topology (ForbiddenSignalPath)

Class	⟨⟨**atpObject**⟩⟩ **ForbiddenSignalPath**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths			
Class Desc.	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 Class(es)	SignalPathConstraint			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
operation	SwcToSwc Operation Arguments	*	aggregation	Reference to the operation arguments of one operation which must not take the predefined way in the topology.
physical Channel	Physical Channel	1..*	reference	The SwcToSwcSignal must not be transmitted on one of these physical channels.
signal	SwcToSwc Signal	*	aggregation	The data element which must not take the predefined way in the topology.

Table 4.30: ForbiddenSignalPath

4.2.2.3 PermissibleSignalPath

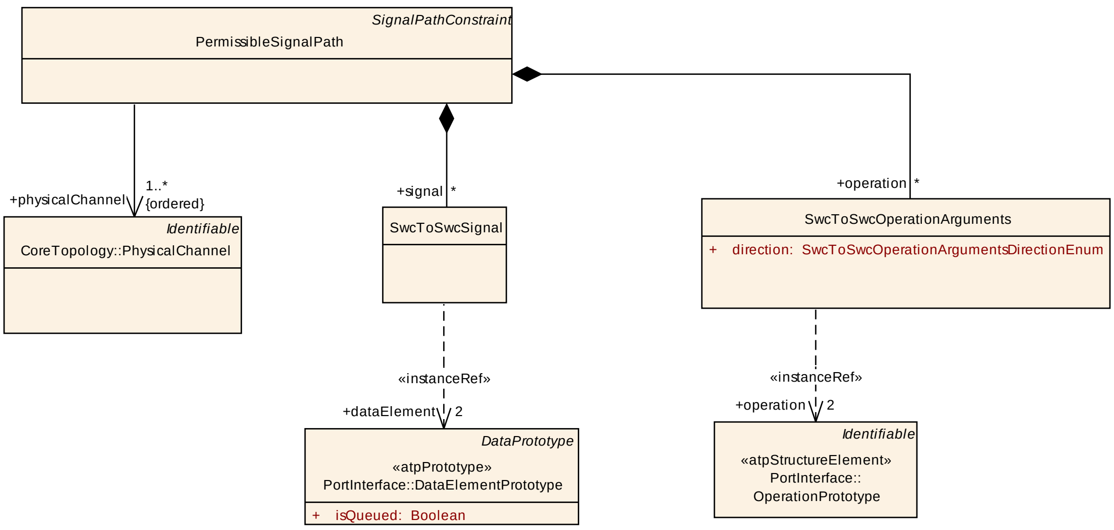

Figure 4.14: Description of the signal path that a signal must take in the topology (PermissibleSignalPath)

Class	⟨⟨**atpObject**⟩⟩ **PermissibleSignalPath**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths			
Class Desc.	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 Class(es)	SignalPathConstraint			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
operation	SwcToSwc Operation Arguments	*	aggregation	The arguments of an operation that can take the predefined way in the topology.
physical Channel (ordered)	Physical Channel	1..*	reference	The SwcToSwcSignal can be transmitted on one of these physical channels.
signal	SwcToSwc Signal	*	aggregation	The data element which can take the predefined way in the topology.

Table 4.31: PermissibleSignalPath

4.2.2.4 SeparateSignalPath

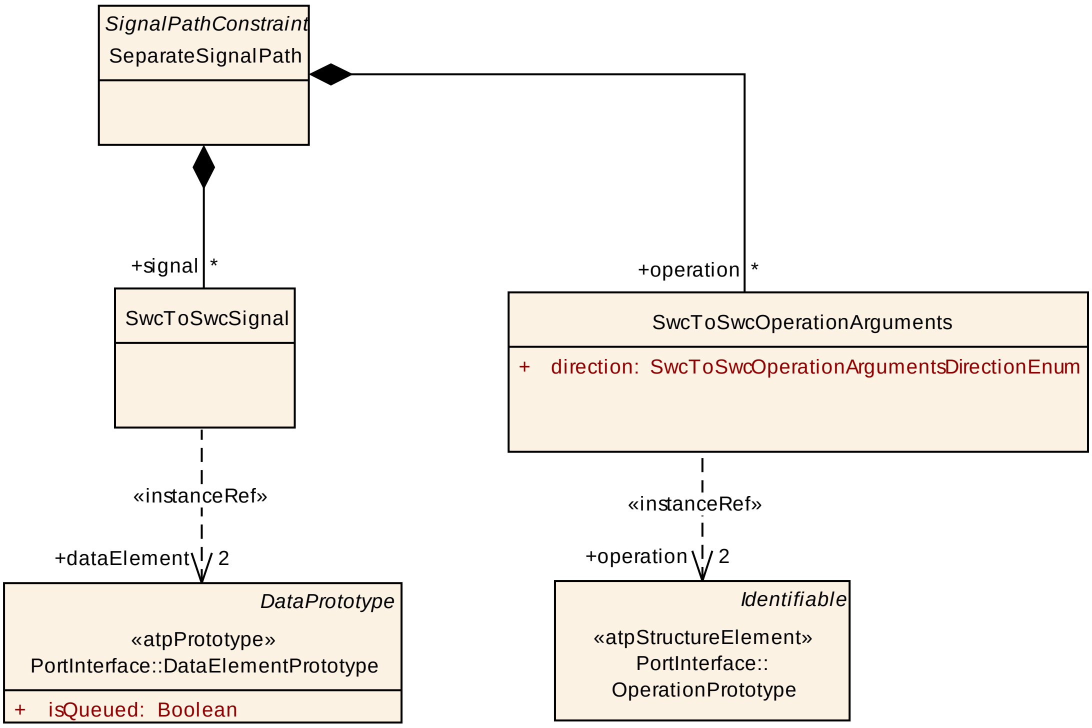

Figure 4.15: Description of signals that must not take the same way in the topology (SeparateSignalPath)

Class	⟨⟨**atpObject**⟩⟩ **SeparateSignalPath**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths			
Class Desc.	The SeparateSignalPath describes that two SwcToSwcSignals and/or SwcToSwcOperationArguments must not take the same way (Signal Path) in the topology (e.g. Redundancy).			
Base Class(es)	SignalPathConstraint			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
operation	SwcToSwc Operation Arguments	*	aggregation	The SwcToSwcOperationArguments that must not take the same way (Signal Path) in the topology.
signal	SwcToSwc Signal	*	aggregation	The SwcToSwcSignals that must not take the same way (Signal Path) in the topology.

Table 4.32: 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 4.16 shows the meta-model for resource estimations for RTE and basic SW.

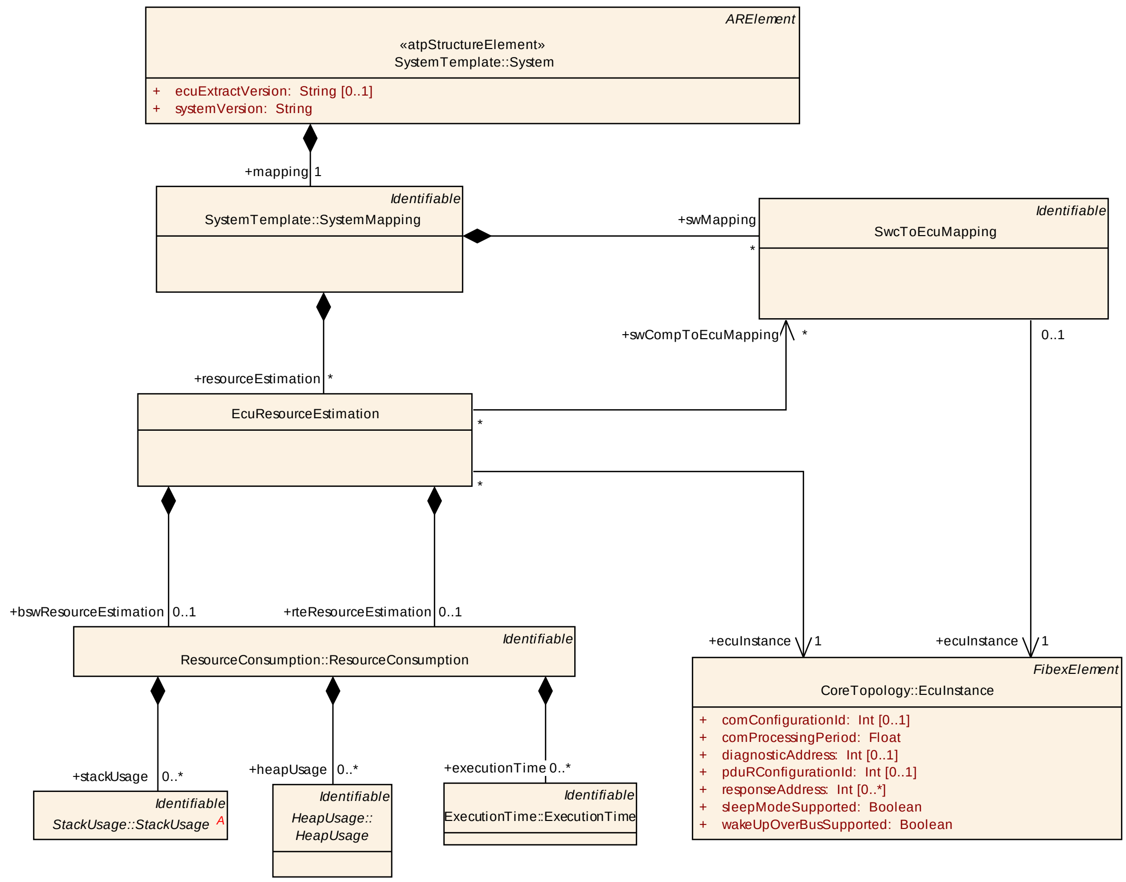

Figure 4.16: ECU resource estimations (ResourceEstimation)

Class	⟨⟨**atpObject**⟩⟩ **EcuResourceEstimation**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Class Desc.	Resource estimations for RTE and BSW of a single ECU instance.			
Base Class(es)	ARObject			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
bswResourceEstimation	ResourceConsumption	0..1	aggregation	Estimation for the resource consumption of the basic software.
ecuInstance	EcuInstance	1	reference	Reference to the ECU this estimation is done for.
rteResourceEstimation	ResourceConsumption	0..1	aggregation	Estimation for the resource consumption of the run time environment.

swCompToEcuMapping	SwcToEcuMapping	*	reference	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 4.33: EcuResourceEstimation

Class	⟨⟨**atpObject**⟩⟩ **ResourceConsumption**			
Package	M2::AUTOSARTemplates::CommonStructure::ResourceConsumption			
Class Desc.	Description of consumed resources by one implementation of a software.			
Base Class(es)	Identifiable			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
executionTime	ExecutionTime	*	aggregation	Collection of the execution time descriptions for the runnable entities of this implementation.
heapUsage	HeapUsage	*	aggregation	Collection of the heap memory allocated by this implementation.
objectFileSection	MemorySection	1..*	aggregation	Provides additional information to the sections of the object-file containing the implementation of the SW-Component
stackUsage	StackUsage	*	aggregation	Collection of the stack memory usage for each runnable entity of this implementation.

Table 4.34: ResourceConsumption

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

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.

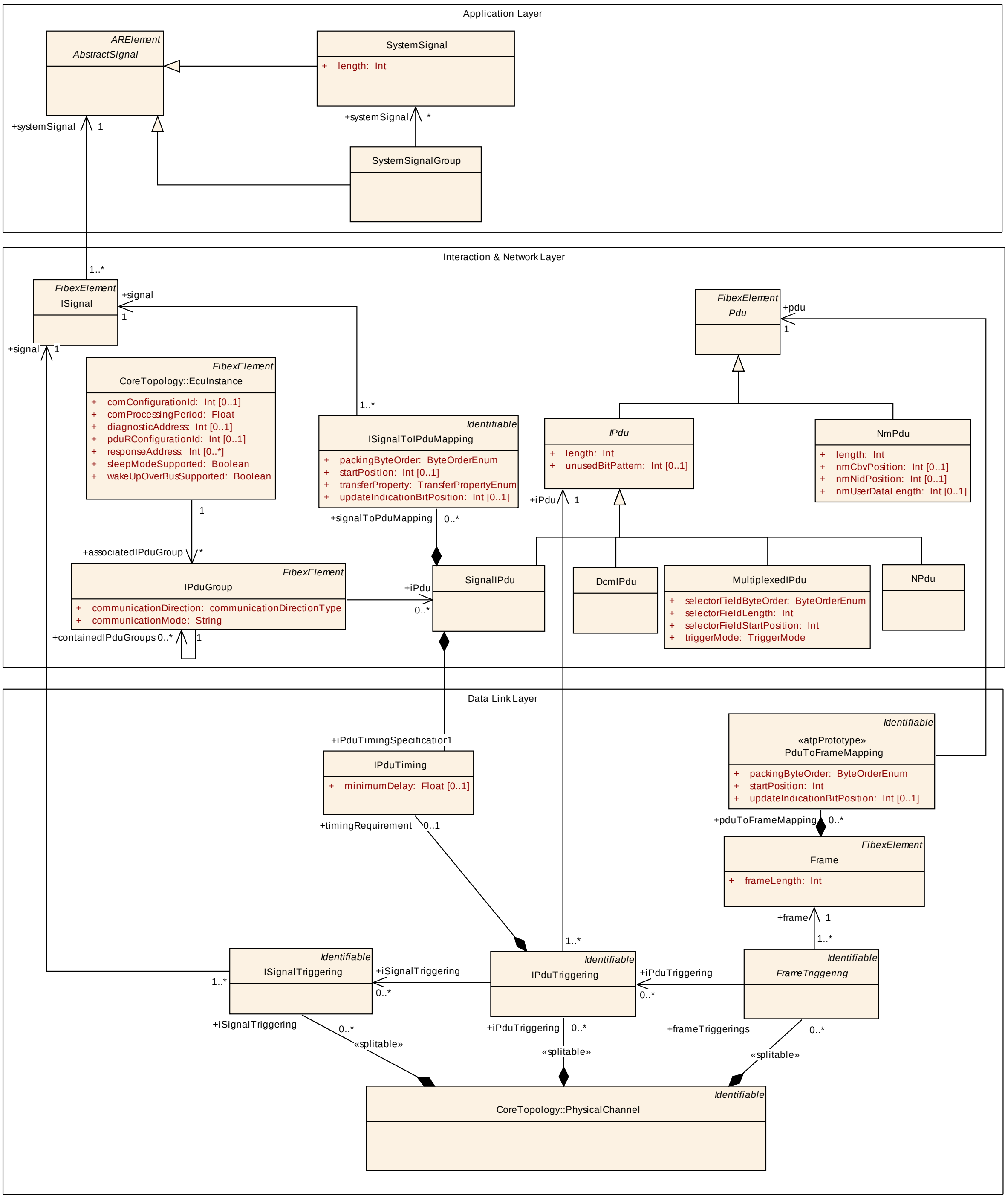

Figure 5.1: Communication Overview (FibexCore: Communication)

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.

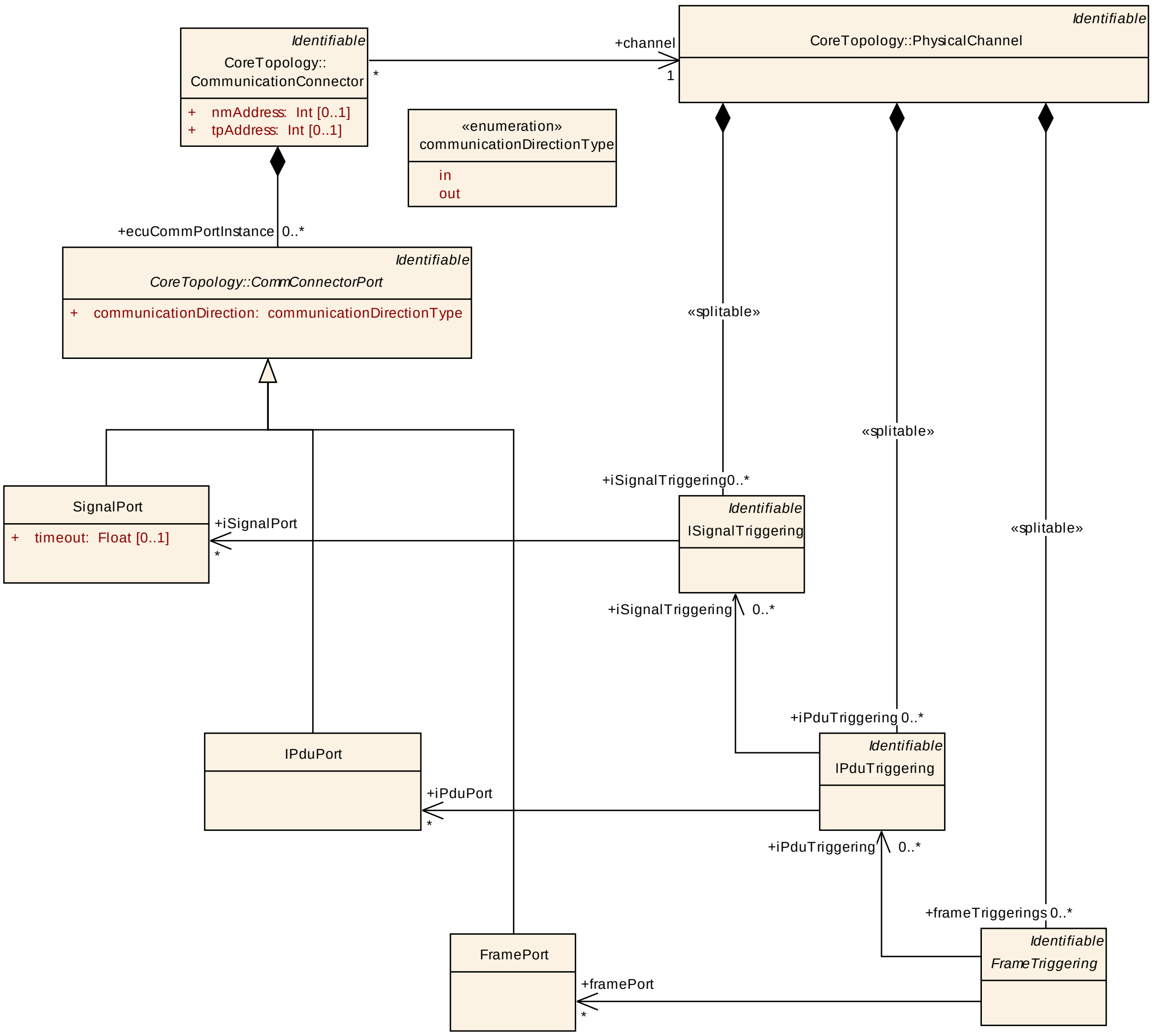

Figure 5.2: Communication Matrix (FibexCore: CommunicationMatrix)

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.

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

Class	⟨⟨**atpObject**⟩⟩ **CommConnectorPort (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Class Desc.	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 Class(es)	Identifiable			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
communicationDirection	communicationDirectionType	1	aggregation	communication Direction of the Connector Port (input or output Port).

Table 5.1: CommConnectorPort

Class	⟨⟨**atpObject**⟩⟩ **FramePort**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	Connectors reception or send port on the referenced channel referenced by a FrameTriggering.			
Base Class(es)	CommConnectorPort			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***

Table 5.2: FramePort

Class	⟨⟨**atpObject**⟩⟩ **IPduPort**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	Connectors reception or send port on the referenced channel referenced by an IPduTriggering.			
Base Class(es)	CommConnectorPort			
Attribute	*Datatype*	*Mul.*	*Link Type*	*Description*

Table 5.3: IPduPort

Class	⟨⟨**atpObject**⟩⟩ **SignalPort**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	Connectors reception or send port on the referenced channel referenced by an ISignalTriggering.			
Base Class(es)	CommConnectorPort			
Attribute	*Datatype*	*Mul.*	*Link Type*	*Description*
timeout	Float	0..1	aggregation	Optional timeout value in seconds for the reception of the ISignal. In case the System Description doesn't use a complete Software Component Description (VFB View). This supports the inclusion of legacy system signals. If a full DataMapping exist for the SystemSignal this information may be available from a configured ReceiverComSpec, in this case the timeout value in ReceiverComSpec override this optional timeout specification.

Table 5.4: SignalPort

5.1.1 Port elements in ECU Extract

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

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

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

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

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

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

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

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

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

5.2 ISignals

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

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

To support the ”signal fan-out” `ISignals` 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`.

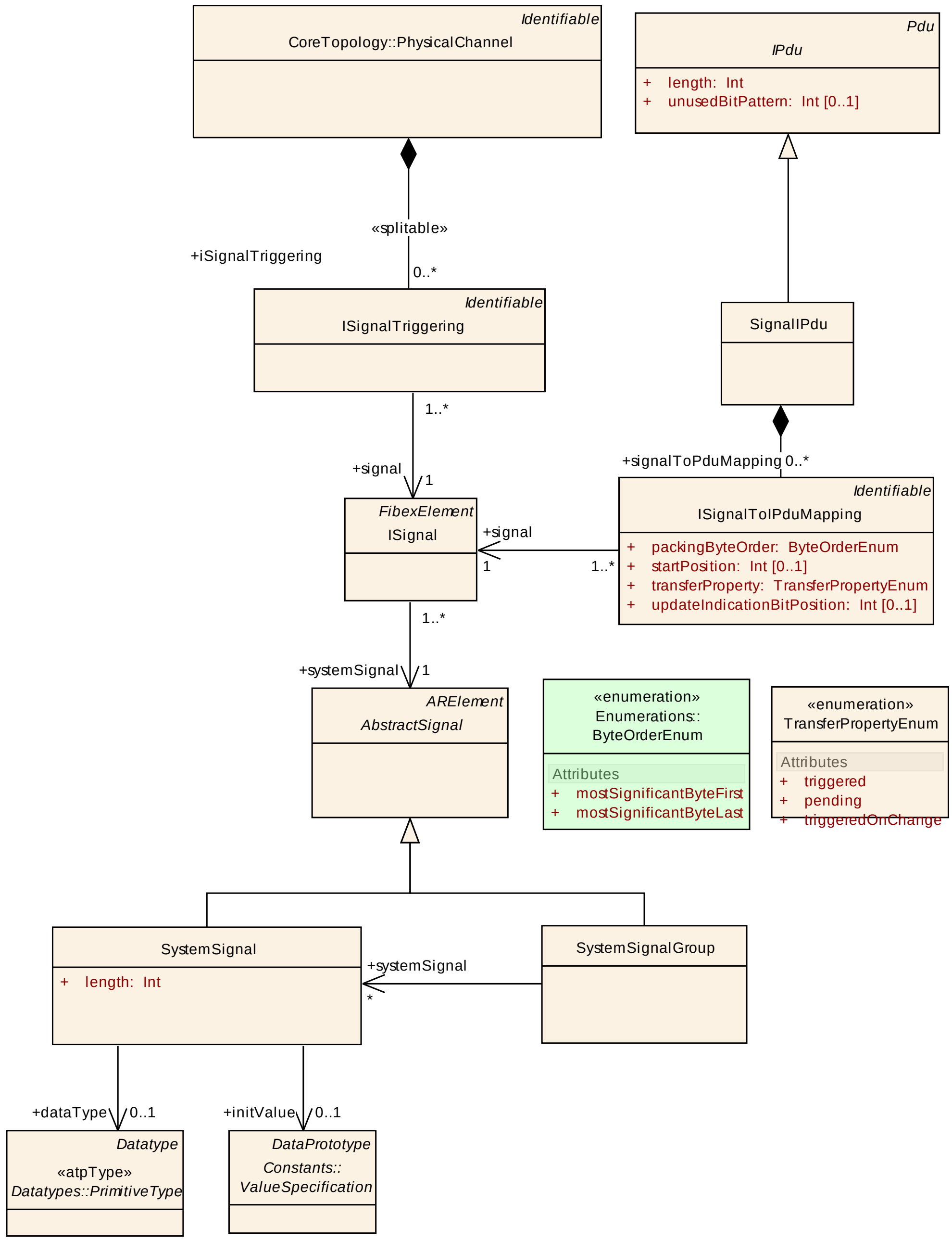

Figure 5.3: ISignals and the mapping into IPdus (FibexCore: SignalOverview)

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	⟨⟨**atpObject**⟩⟩ **ISignal**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	Signal of the Interaction Layer. The RTE supports a "signal fan-out" where the same System Signal is sent in different SignalIPdus to multiple receivers. The System Signal is unique per System. To support the RTE "signal fan-out" each SignalIPdu contains ISignals. If the same System Signal is to be mapped into several SignalIPdus there is one ISignal needed for each ISignalToIPduMapping. ISignals describe the Interface between the Precompile configured RTE and the potentially Postbuild configured Com Stack (see ECUC Parameter Mapping). In 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 Class(es)	FibexElement			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
system Signal	Abstract Signal	1	reference	Reference to the System Signal that is supposed to be transmitted in the ISignal.

Table 5.5: ISignal

Class	⟨⟨**atpObject**⟩⟩ **SystemSignalGroup**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	AbstractSignal			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
system Signal	System Signal	*	reference	Reference to a set of signals that must always be kept together.

Table 5.6: SystemSignalGroup

Class	⟨⟨**atpObject**⟩⟩ **ISignalToIPduMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 I-PDU fan-out. Every ISignal can only be mapped into one SignalIPdu. Several ISignalToIPduMappings to the same ISignal are only relevant when the ECU handles the signal gateway. If a mapping for the SystemSignalGroup is defined, only the UpdateIndicationBitPosition is relevant, and the startPosition shall be ignored.			
Base Class(es)	Identifiable			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
packing ByteOrder	ByteOrder Enum	1	aggregation	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	reference	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.

startPosition	Integer	0..1	aggregation	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). Bits within the IPdu are counted as follows (see the OSEK COM v3.0.3 specification) : Bit 0 corresponds to Byte 0 Bit 0 Bit 1 corresponds to Byte 0 Bit 1 ..... Bit 8 corresponds to Byte 1 Bit 0 etc. 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. Note also that the absolute position of the signal in the SignalIPdu is then determined by the definition of the packingByteOrder attribute of the signal. If a mapping for the SystemSignalGroup is defined, only the UpdateIndicationBitPosition is relevant, and the startPosition shall be ignored.

transfer Property	Transfer Property Enum	1	aggregation	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. The immediate transmission of the IPdu is caused even if only one Signal of an IPdu has the transferProperty triggered or triggeredOnChange 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.
update Indication BitPosition	Integer	0..1	aggregation	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. The updateIndicationBitPosition is determined by the definition of the packingByteOrder attribute. If Big Endian is specified, the updateIndicationBitPosition indicates the bit position of the most significant bit in the ISignalIPdu. If Little Endian is specified, the updateIndicationBitPosition indicates the bit position of the least significant bit in the ISignalIPdu.

Table 5.7: ISignalToIPduMapping

Enumeration	**TransferPropertyEnum**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::Core Communication
Enum Desc.	Transfer Properties of a Signal.
Literal	***Description***
triggered	The signal in the assigned IPdu is updated and a request for the IPdu's transmission is made.
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.
triggeredOn Change	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.

The following example (Figure 5.4) explains the attribute `packingByteOrder` in more detail. The `packingByteOrder` attribute defines the way byte frontiers are crossed when mapping data elements to I-PDUs. The example shows how a nine bit data element fills a I-PDU bit by bit (starting from signal bit 0). It starts somewhere in Byte n and if the end of the byte is reached, there are two choices to continue. The two options are to go ahead from byte n to byte n+1 (Little Endian) or to go backwards from byte n to byte $n - 1$ (Big Endian). The startPosition is now defined depending on the endianess: if little endian, the position of the data element is bit 0; if big endian, the position of the data element is bit 9. More details can be found in FIBEX [7].

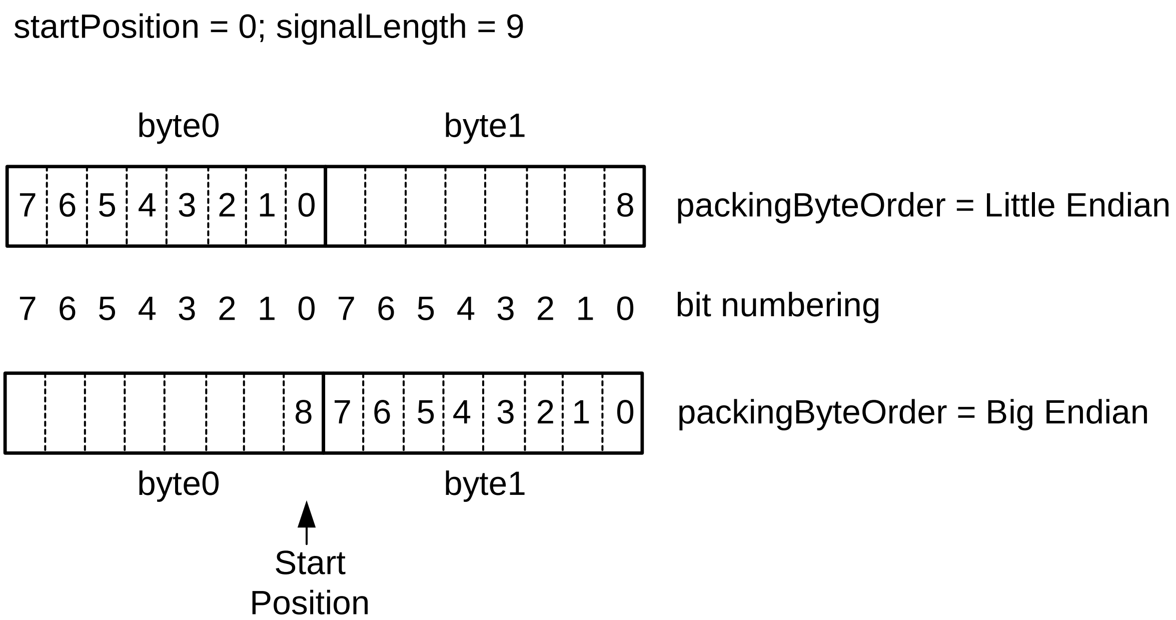

Figure 5.4: PackingByteOrder Example

5.3 PDUs - I-Pdus, N-Pdus and NmPdus

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. [2] This information can be derived from the System Template.

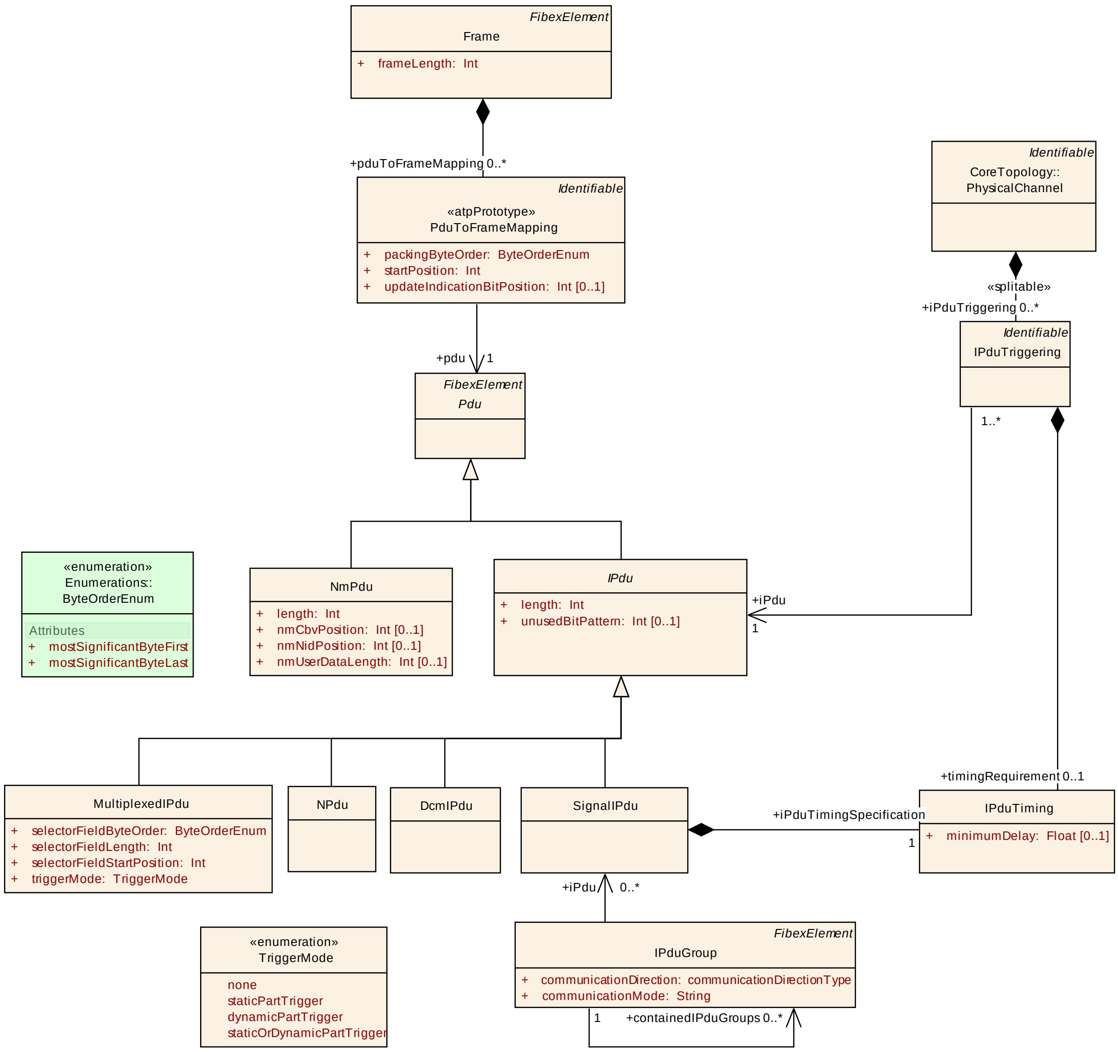

Figure 5.5: Pdus and the mapping into Frames (FibexCore: PDUOverview)

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

A timing description IPduTiming can be aggregated directly by the 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 ele-

[2] 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.

ment 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	⟨⟨**atpObject**⟩⟩ **Pdu (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	Collection of all Pdus that can be routed through a bus interface.			
Base Class(es)	FibexElement			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***

Table 5.8: Pdu

Class	⟨⟨**atpObject**⟩⟩ **IPdu (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	Pdu			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
length	Integer	1	aggregation	The size of the IPDU in bits.
unusedBit Pattern	Integer	0..1	aggregation	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 5.9: IPdu

Class	⟨⟨**atpObject**⟩⟩ **SignalIPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	IPdu			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
iPduTiming Specifica-tion	IPduTim-ing	1	aggregation	Timing specification for Com IPdus (Transmission Modes).

signalTo PduMap-ping	ISignal ToIPdu Mapping	*	aggregation	Definition of SignalToIPduMappings included in the SignalIPdu.

Table 5.10: SignalIPdu

Class	⟨⟨**atpObject**⟩⟩ **NPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	IPdu			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***

Table 5.11: NPdu

Class	⟨⟨**atpObject**⟩⟩ **NmPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	Network Management I-Pdu			
Base Class(es)	Pdu			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
length	Integer	1	aggregation	The size of the NmPDU in bits.
nmCbv Position	Integer	0..1	aggregation	Defines the position of the control bit vector within the NM PDU (Bitpositon).
nmNid Position	Integer	0..1	aggregation	Defines the bitposition of the source node identifier within the NM PDU.
nmUser Data Length	Integer	0..1	aggregation	Defines the length in Bytes of the user data contained in the NM PDU.

Table 5.12: NmPdu

Class	⟨⟨**atpObject**⟩⟩ **DcmIPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	Represents the I-PDU's handled by Dcm.			
Base Class(es)	IPdu			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**

Table 5.13: DcmIPdu

Class	⟨⟨**atpObject**⟩⟩ **IPduGroup**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	FibexElement			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
communicationDirection	communicationDirectionType	1	aggregation	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	aggregation	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	*	reference	An I-PDU group can be included in other I-Pdu groups.
iPdu	SignalIPdu	*	reference	Reference to a set of SignalIPdus, which are contained in the I-Pdu Group.

Table 5.14: 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.

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

Class	⟨⟨**atpObject**⟩⟩ **PduToFrameMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	Identifiable			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
packing ByteOrder	ByteOrder Enum	1	aggregation	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	reference	Reference to a I-Pdu, N-Pdu or NmPdu that is transmitted in the Frame.
startPosi-tion	Integer	1	aggregation	This parameter is necessary to describe the byteposition of a Pdu within a Frame. Note that the absolute position of the Pdu in the Frame is determined by the definition of the packingByteOrder attribute. If Big Endian is specified, the start position indicates the bit position of the most significant bit in the Frame. If Little Endian is specified, the start position indicates the bit position of the least significant bit in the Frame.

update Indication BitPosition	Integer	0..1	aggregation	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. The updateIndicationBitPosition is determined by the definition of the packingByteOrder attribute. If Big Endian is specified, the updateIndicationBitPosition indicates the bit position of the most significant bit in the Frame. If Little Endian is specified, the updateIndicationBitPosition indicates the bit position of the least significant bit in the Frame.

Table 5.15: PduToFrameMapping

Class	⟨⟨**atpObject**⟩⟩ **IPduTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
cyclicTim-ing	Cyclic Timing	0..1	aggregation	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	EventControlledTiming	0..1	aggregation	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	aggregation	Minimum Delay in seconds between successive transmissions of this I-PDU, independent of the Transmission Mode.
requestControlledTiming	RequestControlledTiming	0..1	aggregation	A RequestControlled Timing can be defined as a Timing Requirement. Timing Requirements are aggregated by the IPduTriggering/IPduTiming element.
transmissionModeDeclaration	TransmissionModeDeclaration	0..1	aggregation	AUTOSAR COM allows configuring statically two different transmission modes for each I-PDU (True and False). The Transmission Mode Selector evaluates the conditions for a subset of signals and decides the transmission mode. It is possible to switch between the transmission modes during runtime.

Table 5.16: IPduTiming

Class	⟨⟨**atpObject**⟩⟩ **IPduTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	Identifiable			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
iPdu	IPdu	1	reference	Reference to the Ipdu for which the I-Pdu triggering is defined. One I-Pdu can be triggered on different channels.
iPduPort	IPduPort	*	reference	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.
iSignal Triggering	ISignal Triggering	*	reference	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.
timingRe-quirement	IPduTim-ing	0..1	aggregation	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 5.17: IPduTriggering

5.4 Frames

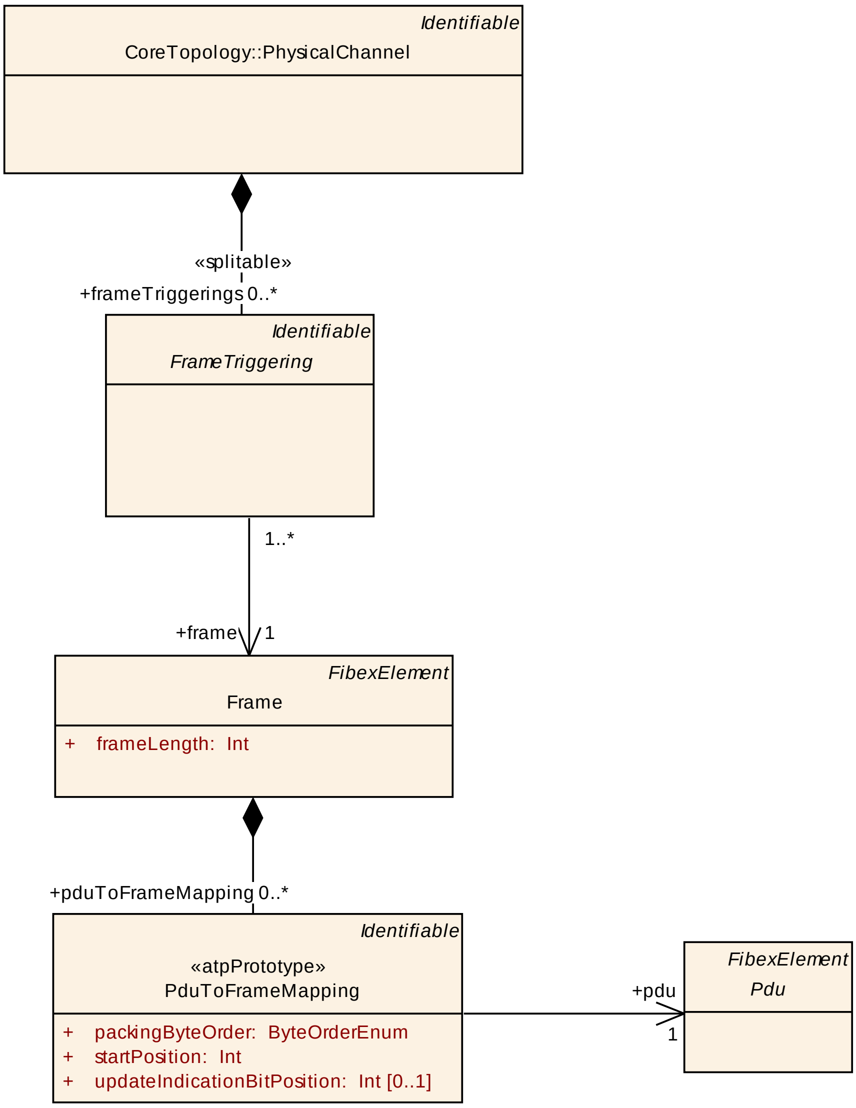

Figure 5.6: Frame Overview (FibexCore: FrameOverview)

`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	⟨⟨**atpObject**⟩⟩ **Frame**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	Data frame which is sent over a communication medium. This element describes the pure Layout of a frame sent on a channel.			
Base Class(es)	FibexElement			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
frame Length	Integer	1	aggregation	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).
pduTo Frame Mapping	PduTo Frame Mapping	*	aggregation	A frames layout as a sequence of Pdus.

Table 5.18: Frame

Class	⟨⟨**atpObject**⟩⟩ **FrameTriggering (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	Identifiable			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
frame	Frame	1	reference	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.
framePort	FramePort	*	reference	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.
iPduTrig-gering	IPduTrig-gering	*	reference	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 5.19: 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.

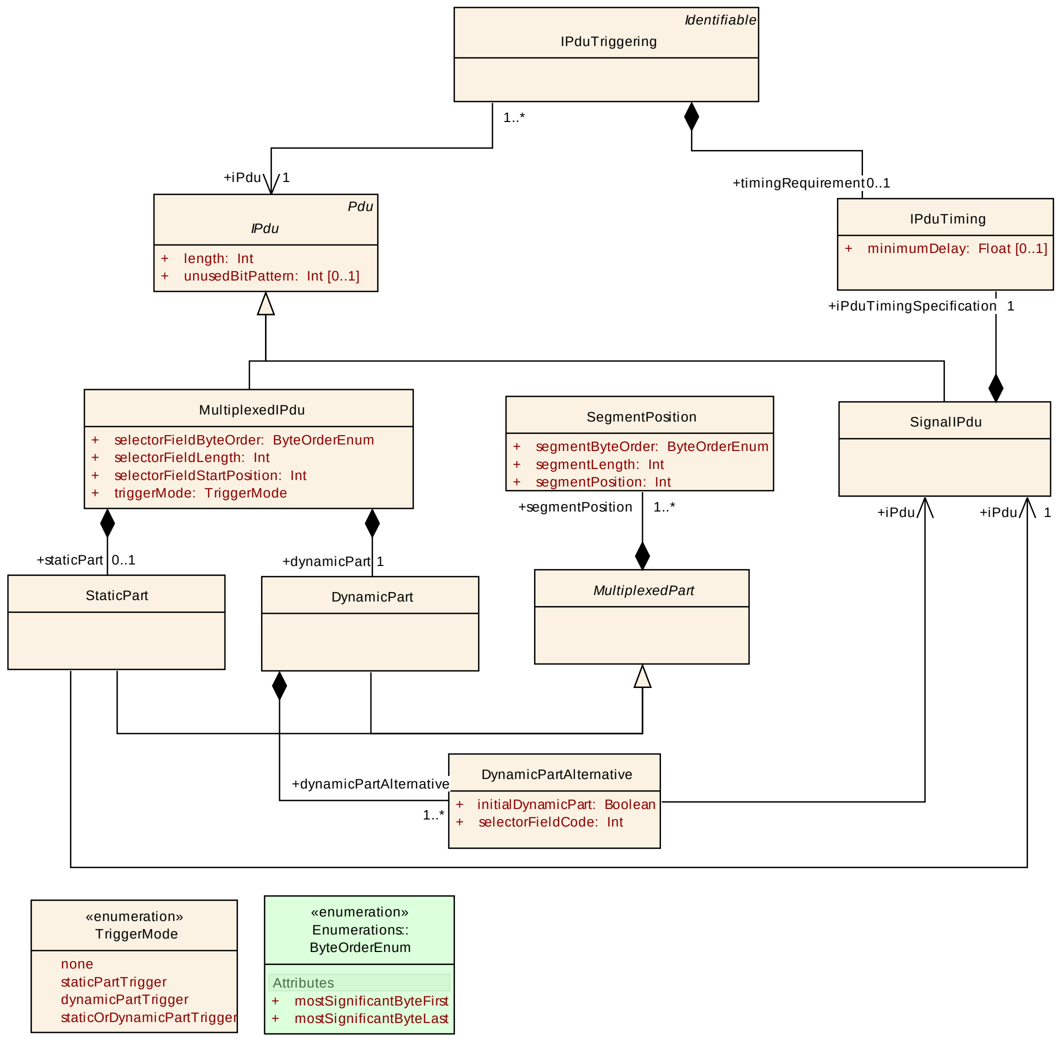

Figure 5.7: I-Pdu Multiplexer (FibexCore: IPDUMultiplexerOverview)

Class	⟨⟨atpObject⟩⟩ **MultiplexedIPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	IPdu			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
dynamic Part	Dynamic Part	1	aggregation	According to the value of the selector field some parts of the IPdu have a different layout.
selector FieldByte Order	ByteOrder Enum	1	aggregation	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.
selector Field Length	Integer	1	aggregation	The size in bits of the selector field shall be configurable in a range of one bit and eight bits.
selector FieldStart Position	Integer	1	aggregation	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	aggregation	The static part of the multiplexed IPdu is the same regardless of the selector field. The static part is optional.
trigger Mode	Trigger Mode	1	aggregation	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 5.20: MultiplexedIPdu

Enumeration	**TriggerMode**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::Core Communication
Enum Desc.	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**
none	IPduM does not trigger transmission because of receiving anything of this IPdu in case of TriggerTransmit.
staticPart Trigger	IPduM sends a transmission request to the PduR if a static part is received.
dynamicPart Trigger	IPduM sends a transmission request to the PduR if a dynamic part is received.
staticOrDy-namicPart Trigger	IPduM sends a transmission request to the PduR if a static or dynamic part is received.

Class	⟨⟨**atpObject**⟩⟩ **StaticPart**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	MultiplexedPart			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
iPdu	SignalIPdu	1	reference	Reference to a Com IPdu which is routed to the IPduM module and is combined to a multiplexedPdu.

Table 5.21: StaticPart

Class	⟨⟨**atpObject**⟩⟩ **DynamicPart**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	MultiplexedPart			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
dynamic PartAlter-native	Dynamic PartAlter-native	1..*	aggregation	Com IPdu alternatives that are transmitted in the Dynamic Part of the MultiplexedIPdu.

Table 5.22: DynamicPart

Class	⟨⟨**atpObject**⟩⟩ **DynamicPartAlternative**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	ARObject			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**

iPdu	SignalIPdu	1	reference	Reference to a Com IPdu which is routed to the IPduM module and is combined to a multiplexedPdu.
initialDy-namicPart	Boolean	1	aggregation	Dynamic part that shall be used to initialize this multiplexed IPdu. Constraint: Only one "DynamicPartAlternative" in a "DynamicPart" shall be the initialDynamicPart.
selector FieldCode	Integer	1	aggregation	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 5.23: DynamicPartAlternative

Class	⟨⟨**atpObject**⟩⟩ **MultiplexedPart (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	The StaticPart and the DynamicPart have common properties. Both can be separated in multiple segments within the multiplexed PDU.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
segment Position	Segment Position	1..*	aggregation	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 5.24: MultiplexedPart

Class	⟨⟨**atpObject**⟩⟩ **SegmentPosition**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	ARObject			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
segment ByteOrder	ByteOrder Enum	1	aggregation	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.
segment Length	Integer	1	aggregation	Data Length of the segment in bits.
segment Position	Integer	1	aggregation	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 5.25: SegmentPosition

Figure 5.8 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 5.8 the dynamic Part is segmented into two parts.

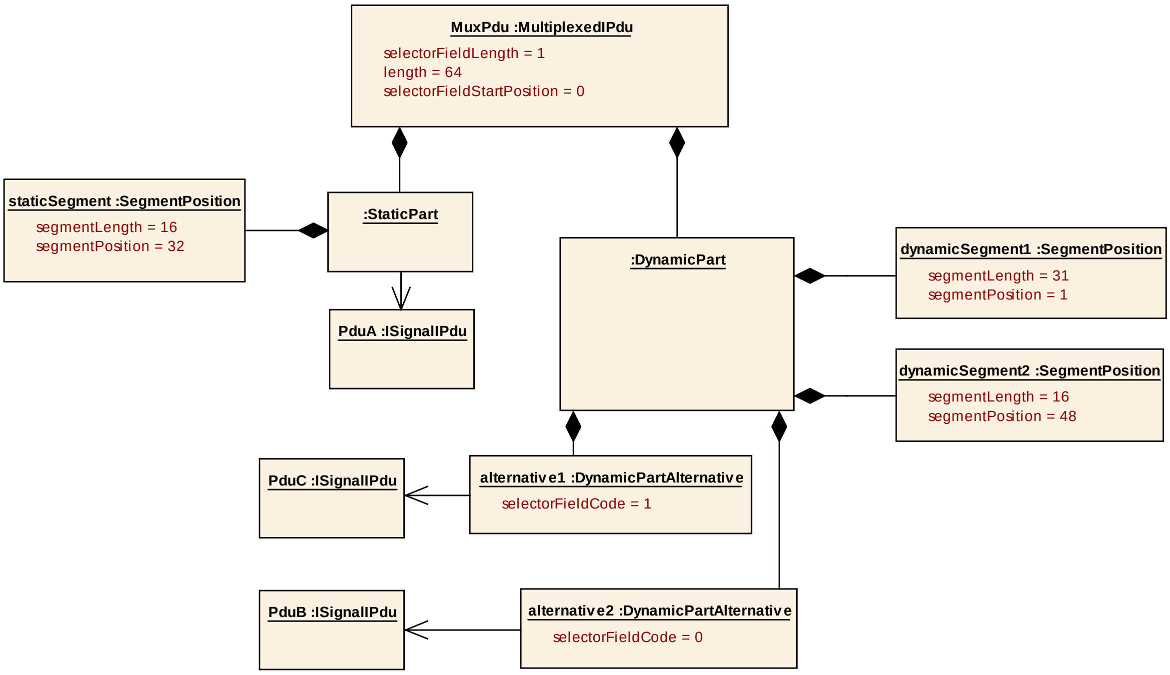

Figure 5.8: I-Pdu Multiplexer Example

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.

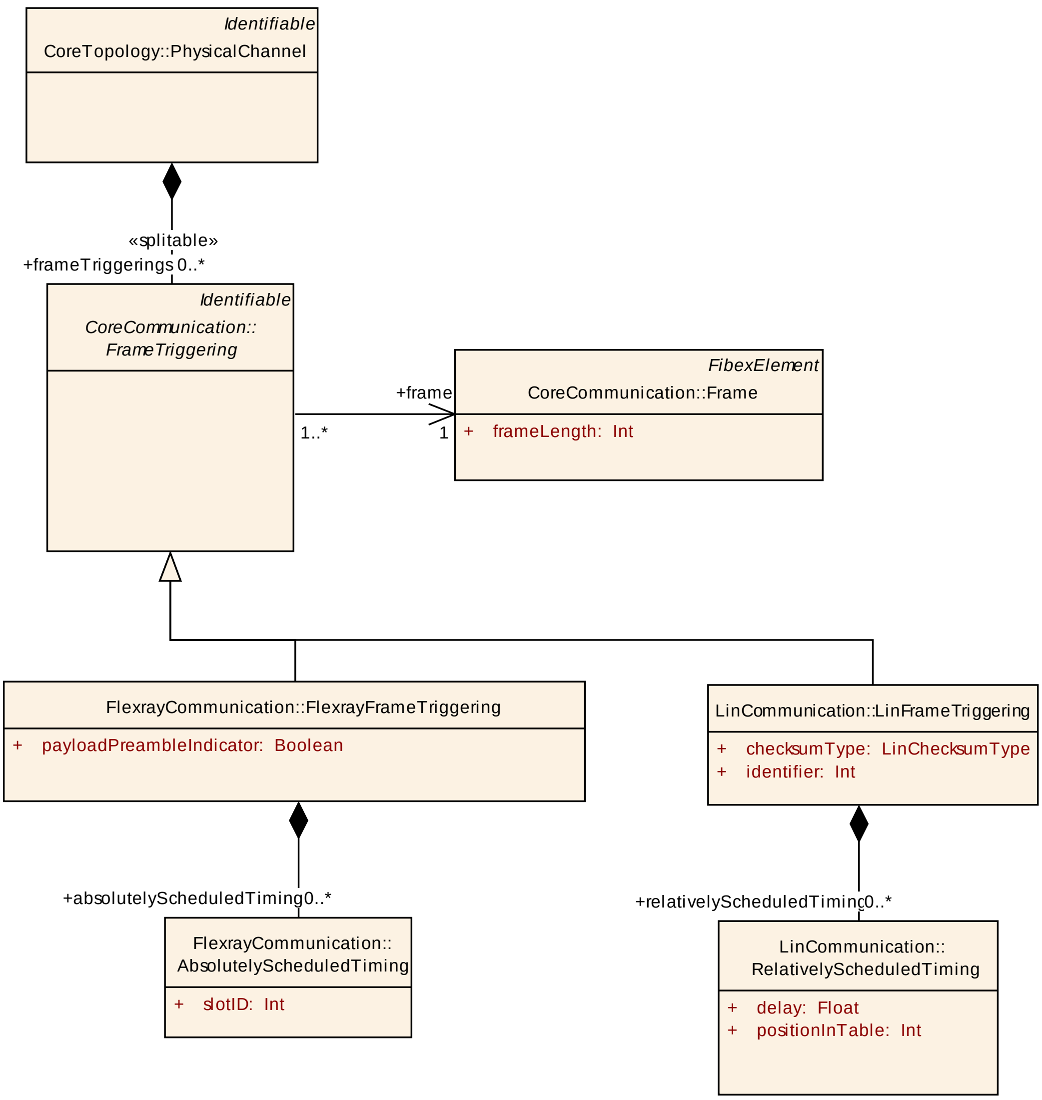

Figure 5.9: Frame Triggering

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.

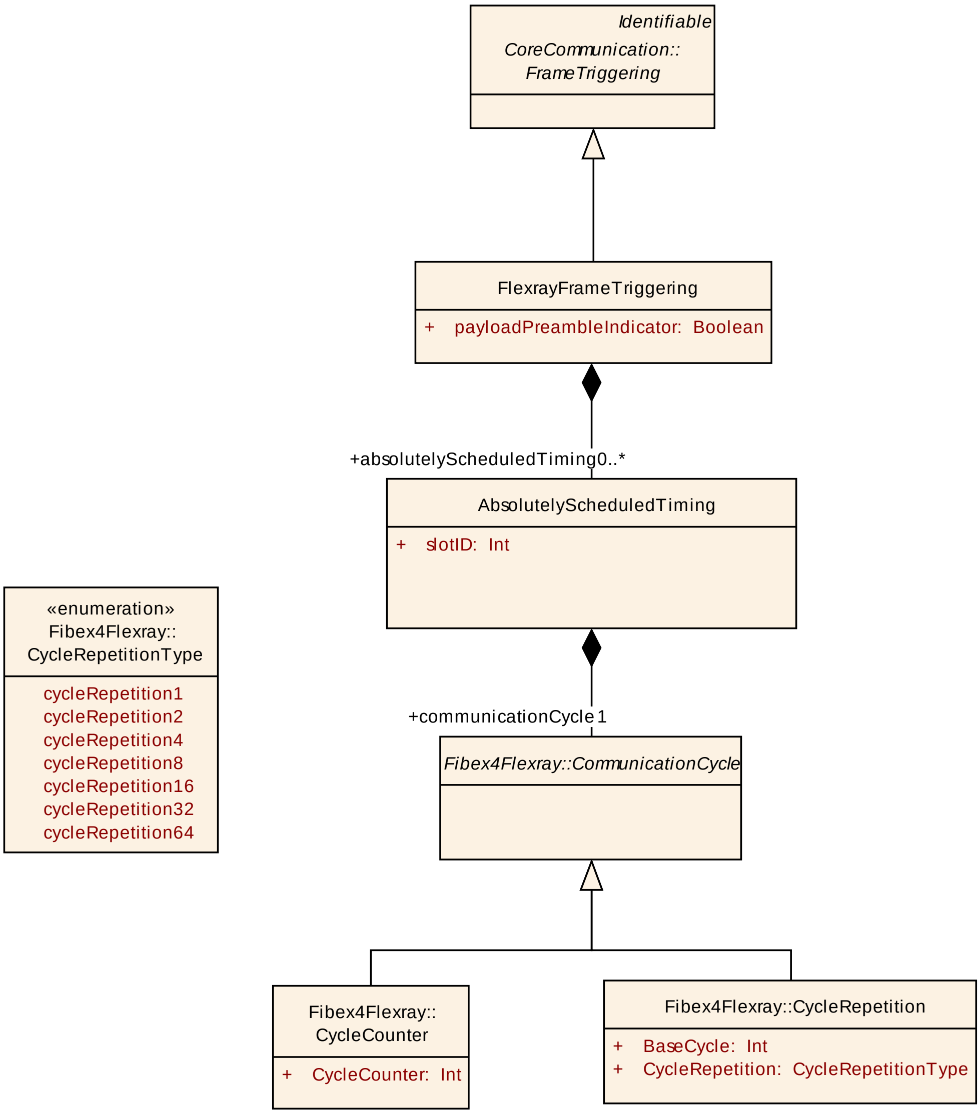

Figure 5.10: Absolutely Scheduled Timing (Fibex4FlexRay:AbsolutelyScheduledTiming)

Class	⟨⟨atpObject⟩⟩ **FlexrayFrameTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::Flexray Communication			
Class Desc.	FlexRay specific attributes to the FrameTriggering			
Base Class(es)	FrameTriggering			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
absolutely Scheduled Timing	Absolutely Scheduled Timing	*	aggregation	Specification of a sending behaviour where the exact time for the frames transmission is guaranteed.
payload Preamble Indicator	Boolean	1	aggregation	Switching the Payload Preamble bit.

Table 5.26: FlexrayFrameTriggering

Class	⟨⟨atpObject⟩⟩ **AbsolutelyScheduledTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::Flexray Communication			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
communic ationCycle	Communic ationCycle	1	aggregation	The communication cycle where the frame is sent.
slotID	Integer	1	aggregation	In the static part the SlotID defines the slot in which the frame is transmitted. The SlotID also determines, in combination with FlexrayCluster::numberOfStaticSlots, whether the frame is sent in static or dynamic segment. In the dynamic part, the slot id is equivalent to a priority. Lower dynamic slot ids are all sent until the end of the dynamic segment. Higher numbers, which were ignored that time, have to wait one cycle and then must try again. minValue: 1 maxValue: 2047

Table 5.27: AbsolutelyScheduledTiming

Class	⟨⟨atpObject⟩⟩ **CommunicationCycle (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray			
Class Desc.	The communication cycle where the frame is sent.			
Base Class(es)	ARObject			
Attribute	*Datatype*	*Mul.*	*Link Type*	*Description*

Table 5.28: CommunicationCycle

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

Class	⟨⟨atpObject⟩⟩ **CycleCounter**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray			
Class Desc.	The communication cycle where the frame is send is described by the attribute "cycleCounter".			
Base Class(es)	CommunicationCycle			
Attribute	*Datatype*	*Mul.*	*Link Type*	*Description*
Cycle Counter	Integer	1	aggregation	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 5.29: CycleCounter

Class	⟨⟨**atpObject**⟩⟩ **CycleRepetition**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray			
Class Desc.	The communication cycle where the frame is send is described by the attributes baseCycle and cycleRepetition.			
Base Class(es)	CommunicationCycle			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
BaseCycle	Integer	1	aggregation	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.
CycleRep-etition	Cycle Repetition Type	1	aggregation	The number of communication cycles (after the first cycle) whenever the frame described by this timing is sent again.

Table 5.30: CycleRepetition

Enumeration	**CycleRepetitionType**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray
Enum Desc.	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**
cycleRepeti-tion64	cycleRepetition value="64"
cycleRepeti-tion1	cycleRepetition value="1"
cycleRepeti-tion2	cycleRepetition value="2"
cycleRepeti-tion4	cycleRepetition value="4"
cycleRepeti-tion8	cycleRepetition value="8"
cycleRepeti-tion16	cycleRepetition value="16"
cycleRepeti-tion32	cycleRepetition value="32"

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.

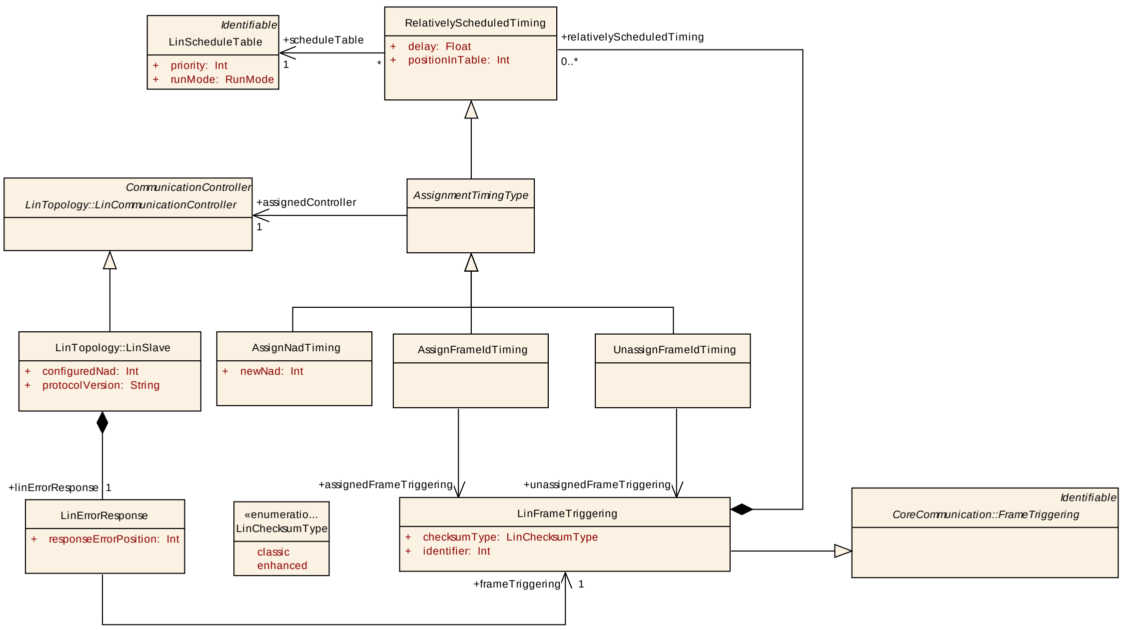

Figure 5.11: Relatively Scheduled and Assignment Timing (Fibex4Lin:AssignmentTiming)

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	**⟨⟨atpObject⟩⟩ LinFrameTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Class Desc.	Lin specific attributes to the FrameTriggering			
Base Class(es)	FrameTriggering			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
checksum Type	LinCheck-sumType	1	aggregation	Type of checksum that the frame is using.
identifier	Integer	1	aggregation	To describe a frames identifier on the communication system, usualy with a fixed identifierValue.
relatively Scheduled Timing	Relatively Scheduled Timing	*	aggregation	Specification of a sending behaviour where the transmission order is predefined.

Table 5.31: LinFrameTriggering

Enumeration	LinChecksumType
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication
Enum Desc.	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.

Class	⟨⟨atpObject⟩⟩ RelativelyScheduledTiming			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Class Desc.	Specification of a sending behavior where the transmission order is predefined, e.g. used on LIN buses			
Base Class(es)	ARObject			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
delay	Float	1	aggregation	Relative delay between this frame and the start of the successor frame in the schedule table in seconds.
positionIn Table	Integer	1	aggregation	Relative position of the frame described by this timing in the schedule table
schedule Table	LinSched-uleTable	1	reference	The master task transmits frame headers based on a schedule table. The master application may use different schedule tables and select among them.

Table 5.32: RelativelyScheduledTiming

Class	⟨⟨atpObject⟩⟩ LinScheduleTable			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Class Desc.	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 Class(es)	Identifiable			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
priority	Integer	1	aggregation	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	aggregation	The schedule table can be executed in two different modes.

Table 5.33: LinScheduleTable

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

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	**⟨⟨atpObject⟩⟩ AssignmentTimingType (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Class Desc.	A LIN specific extension of the common RelativelyScheduledTiming The extension describes the LIN specific assignment frames.			
Base Class(es)	RelativelyScheduledTiming			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
assigned Controller	LinCommunication Controller	1	reference	The LIN slaves controller who is target of this assignment.

Table 5.34: AssignmentTimingType

Class	⟨⟨atpObject⟩⟩ **UnassignFrameIdTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Class Desc.	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 Class(es)	AssignmentTimingType			
Attribute	*Datatype*	*Mul.*	*Link Type*	*Description*
unassigned FrameTriggering	LinFrame Triggering	1	reference	The frame whose identifier is reset by this assignment.

Table 5.35: UnassignFrameIdTiming

Class	⟨⟨atpObject⟩⟩ **AssignFrameIdTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Class Desc.	Schedule entry for an Assign Frame Id master request.			
Base Class(es)	AssignmentTimingType			
Attribute	*Datatype*	*Mul.*	*Link Type*	*Description*
assigned Frame Triggering	LinFrame Triggering	1	reference	The frame whose identifier is set by this assignment.

Table 5.36: AssignFrameIdTiming

Class	⟨⟨atpObject⟩⟩ **AssignNadTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Class Desc.	Schedule entry for an Assign NAD master request.			
Base Class(es)	AssignmentTimingType			
Attribute	*Datatype*	*Mul.*	*Link Type*	*Description*
newNad	Integer	1	aggregation	The newly assigned NAD value (valid range 1..126)

Table 5.37: 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`.

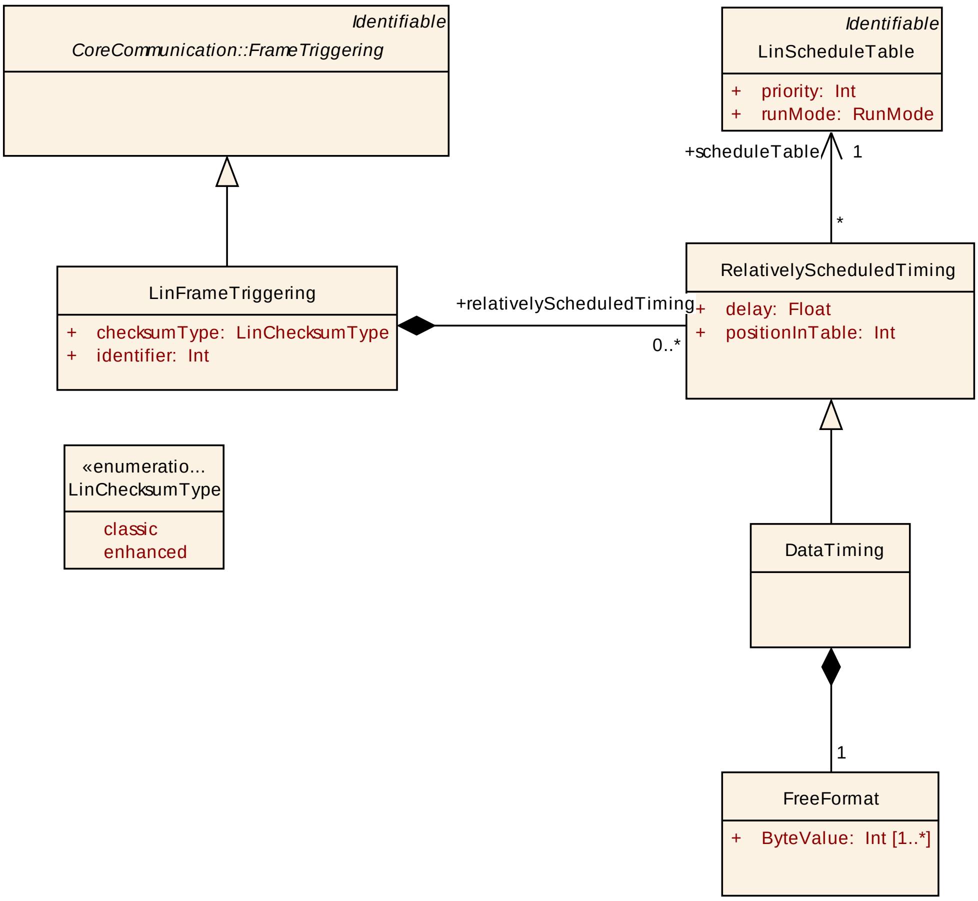

Figure 5.12: Free Format (Fibex4Lin:DataTiming)

Class	⟨⟨atpObject⟩⟩ DataTiming			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Class Desc.	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 Class(es)	RelativelyScheduledTiming			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
freeFormat	FreeFormat	1	aggregation	

Table 5.38: DataTiming

Class	⟨⟨atpObject⟩⟩ FreeFormat			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Class Desc.	Representing freely defined data.			
Base Class(es)	ARObject			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
ByteValue	Integer	1..*	aggregation	The integer Value of a freely defined data byte.

Table 5.39: 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.

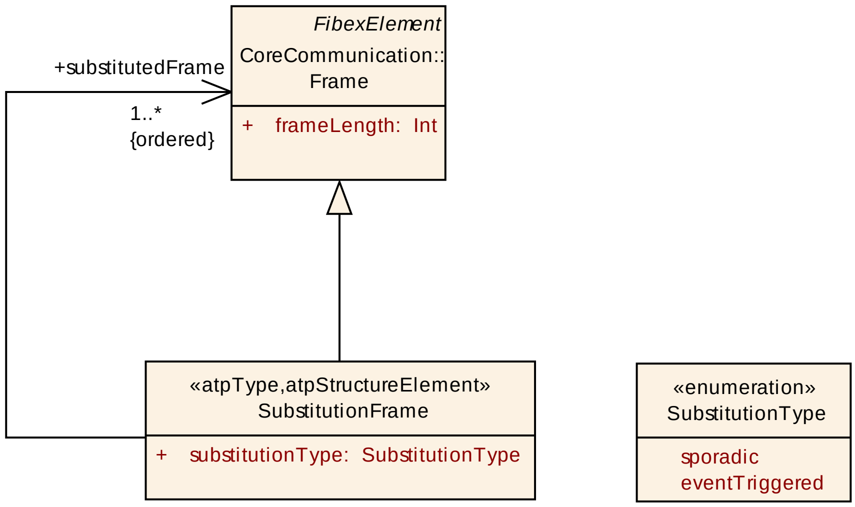

Figure 5.13: Substitution Frame (Fibex4Lin:SubstitutionFrame)

Class	⟨⟨**atpStructureElement**⟩⟩ **SubstitutionFrame**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Class Desc.	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 Class(es)	Frame			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
substituted Frame (ordered)	Frame	1..*	reference	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.
substitution Type	Substitution Type	1	aggregation	The type of substitution. Substitution frames can either be used for event triggered or for sporadic frames.

Table 5.40: SubstitutionFrame

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

5.9 Can specific description

This chapter describes additions to the CAN definition of Frames.

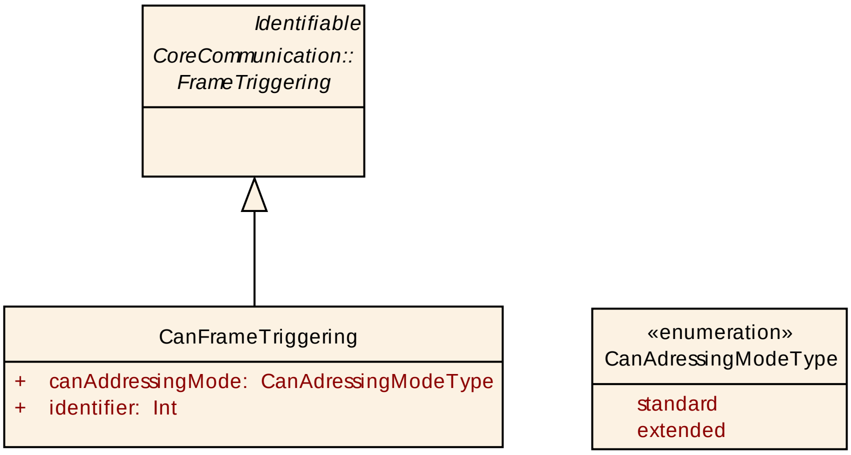

Figure 5.14: CanFrameTriggering (Fibex4Can:CanCommunication)

Class	⟨⟨**atpObject**⟩⟩ **CanFrameTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanCommunication			
Class Desc.	CAN specific attributes to the FrameTriggering			
Base Class(es)	FrameTriggering			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
canAd-dressing Mode	Can Adress-ingMode Type	1	aggregation	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	1	aggregation	To describe a frames identifier on the communication system, usualy with a fixed identifierValue.

Table 5.41: CanFrameTriggering

Enumeration	**CanAdressingModeType**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::Can Communication
Enum Desc.	Indicates whether standard or extended CAN identifiers are used
Literal	***Description***
standard	standard 11-bit-identifiers are used (CAN 2.0A)
extended	extended 29-bit-identifiers are used (CAN 2.0B)

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 5.42: COM Transmission Modes

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.

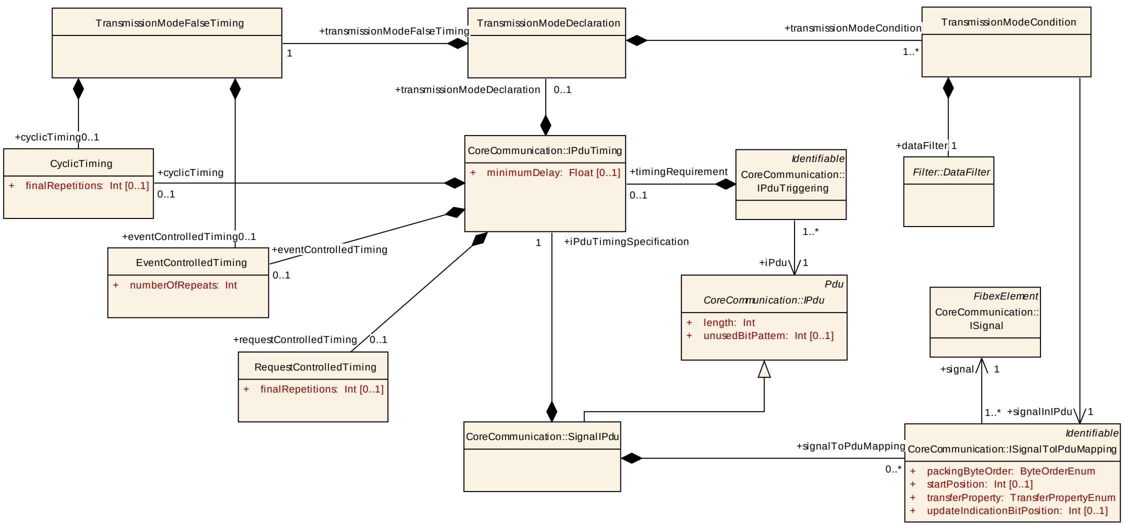

Figure 5.15: IPdu Timing

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 [17].

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 5.42).

Class	⟨⟨**atpObject**⟩⟩ **TransmissionModeDeclaration**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Class Desc.	AUTOSAR COM provides the possibility to define two different TRANSMISSION MODES (True and False) for each I-PDU.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
transmissionModeCondition	TransmissionModeCondition	1..*	aggregation	The Transmission Mode Selector evaluates the conditions for a subset of signals and decides which transmission mode should be used.
transmissionModeFalseTiming	TransmissionModeFalseTiming	1	aggregation	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 5.43: TransmissionModeDeclaration

Class	⟨⟨**atpObject**⟩⟩ **TransmissionModeCondition**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
dataFilter	DataFilter	1	aggregation	Possibilities to define conditions
signalInIPdu	ISignalToIPduMapping	1	reference	Reference to a signal to which a condition is attached.

Table 5.44: TransmissionModeCondition

Class	⟨⟨**atpObject**⟩⟩ **TransmissionModeFalseTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
cyclicTiming	Cyclic Timing	0..1	aggregation	Periodic Transmission Mode.
eventControlled Timing	Event Controlled Timing	0..1	aggregation	Direct Transmission Mode.

Table 5.45: TransmissionModeFalseTiming

Class	⟨⟨**atpObject**⟩⟩ **DataFilter (abstract)**			
Package	M2::AUTOSARTemplates::CommonStructure::Filter			
Class Desc.	Base class for data filters.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***

Table 5.46: DataFilter

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

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.

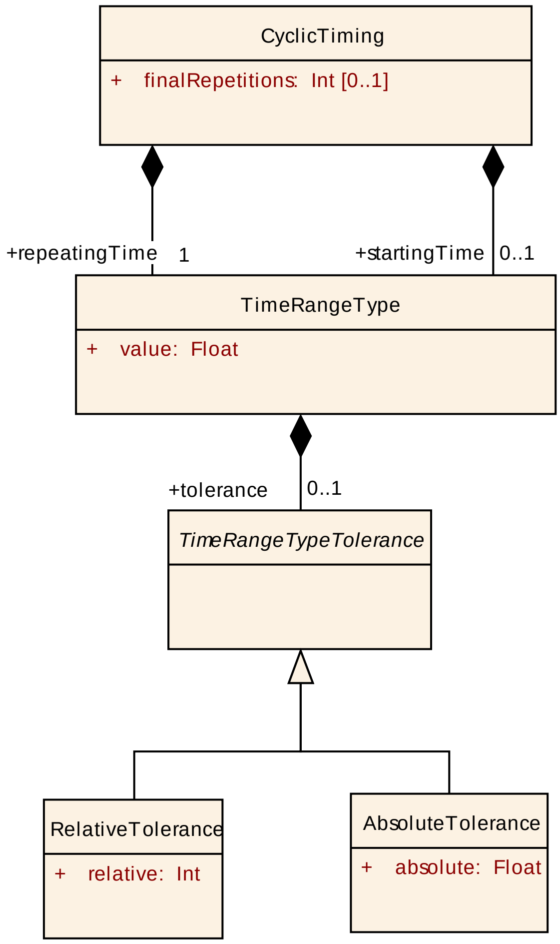

Figure 5.16: Cyclic Timing

Class	⟨⟨**atpObject**⟩⟩ **CyclicTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Class Desc.	Specification of a cyclic sending behavior.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
finalRepetitions	Integer	0..1	aggregation	Number of repetitions the pdu is sent from the moment the stop condition has been met
repeatingTime	TimeRangeType	1	aggregation	Specification of the repeating cycle.
startingTime	TimeRangeType	0..1	aggregation	Specification of the time that is needed before the pdu can be sent the first time.

Table 5.47: CyclicTiming

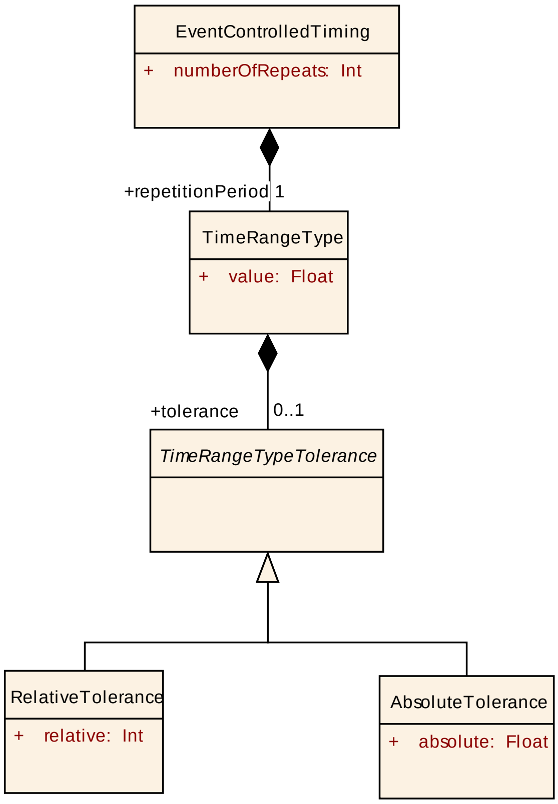

Figure 5.17: EventControlled Timing

Class	**⟨⟨atpObject⟩⟩ EventControlledTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
numberOf Repeats	Integer	1	aggregation	Defines the number of repetitions for the Direct/N-Times transmission mode and the event driven part of Mixed transmission mode.
repetition Period	Time Range Type	1	aggregation	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 5.48: EventControlledTiming

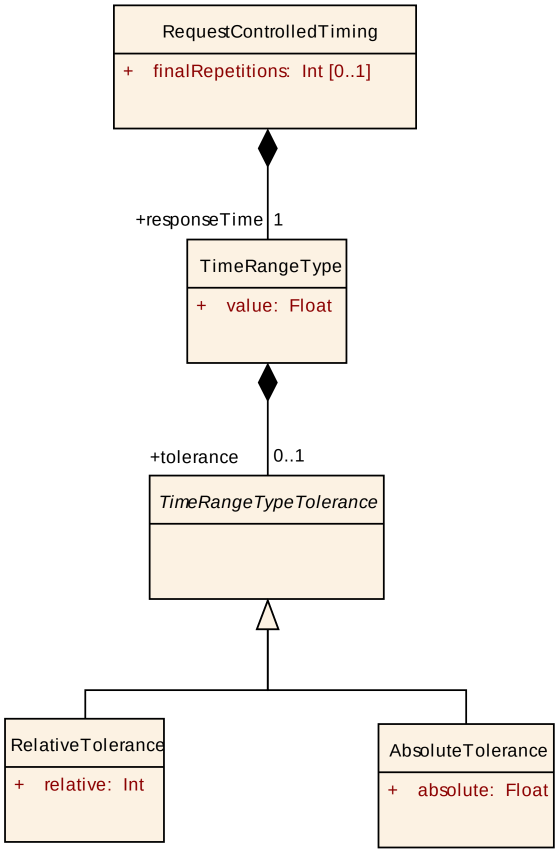

Figure 5.18: RequestControlled Timing

Class	⟨⟨atpObject⟩⟩ **RequestControlledTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Class Desc.	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 Class(es)	ARObject			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
finalRepetitions	Integer	0..1	aggregation	Number of repetitions the frame is sent for a single request
response Time	Time Range Type	1	aggregation	Specification of the time that is needed before the frame can be sent after the requests arrival

Table 5.49: RequestControlledTiming

Class	⟨⟨**atpObject**⟩⟩ **TimeRangeType**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Class Desc.	The timeRange can be specified with the value attribute. Optionally a tolerance can be defined.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
tolerance	Time Range TypeToler-ance	0..1	aggregation	
value	Float	1	aggregation	Average value of a date (in seconds)

Table 5.50: TimeRangeType

Class	⟨⟨**atpObject**⟩⟩ **RelativeTolerance**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Class Desc.	Maximum allowable deviation			
Base Class(es)	TimeRangeTypeTolerance			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
relative	Integer	1	aggregation	Maximum allowable deviation in percent

Table 5.51: RelativeTolerance

Class	⟨⟨**atpObject**⟩⟩ **AbsoluteTolerance**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Class Desc.	Maximum allowable deviation			
Base Class(es)	TimeRangeTypeTolerance			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
absolute	Float	1	aggregation	Maximum allowable deviation in duration (in seconds)

Table 5.52: 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`.

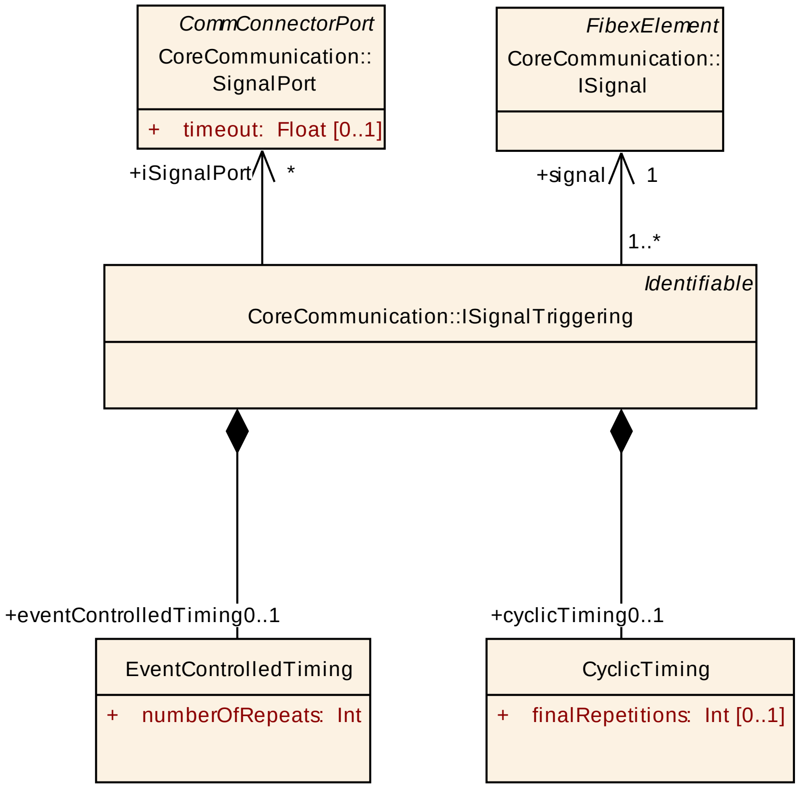

Figure 5.19: Signal Triggering

Class	⟨⟨**atpObject**⟩⟩ **ISignalTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	Identifiable			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
cyclicTiming	Cyclic Timing	0..1	aggregation	Specification of a cyclic sending behavior.
eventControlled Timing	Event Controlled Timing	0..1	aggregation	Specification of a event driven sending behavior.
iSignalPort	SignalPort	*	reference	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	reference	Reference to the ISignal for which the ISignalTriggering is defined.

Table 5.53: ISignalTriggering

5.12 Transport Layer

In AUTOSAR, the Transport Layer has two main purposes: The segmentation and reassembly of messages that are too long to fit into one frame on the underlying communication cluster, and the re-use of fixed frame identifiers for different message content. 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 adressed 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	⟨⟨**atpObject**⟩⟩ **TpChannel (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Class Desc.	A Transport Protocol channel.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***

Table 5.54: TpChannel

Class	⟨⟨**atpObject**⟩⟩ **NPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Class Desc.	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 Class(es)	IPdu			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***

Table 5.55: NPdu

5.12.1 FlexRay Transport Layer

The FlexRay Transport Layer supports multiple sessions, i.e. multiple segmented transfers can be handled at the same time. As each of these sessions requires individual state machines and thus additional resources, the same session (in FlexRay TP called `FlexRayTpChannel`) can be reused for an arbitrary number of `FlexRayTpConnections`.

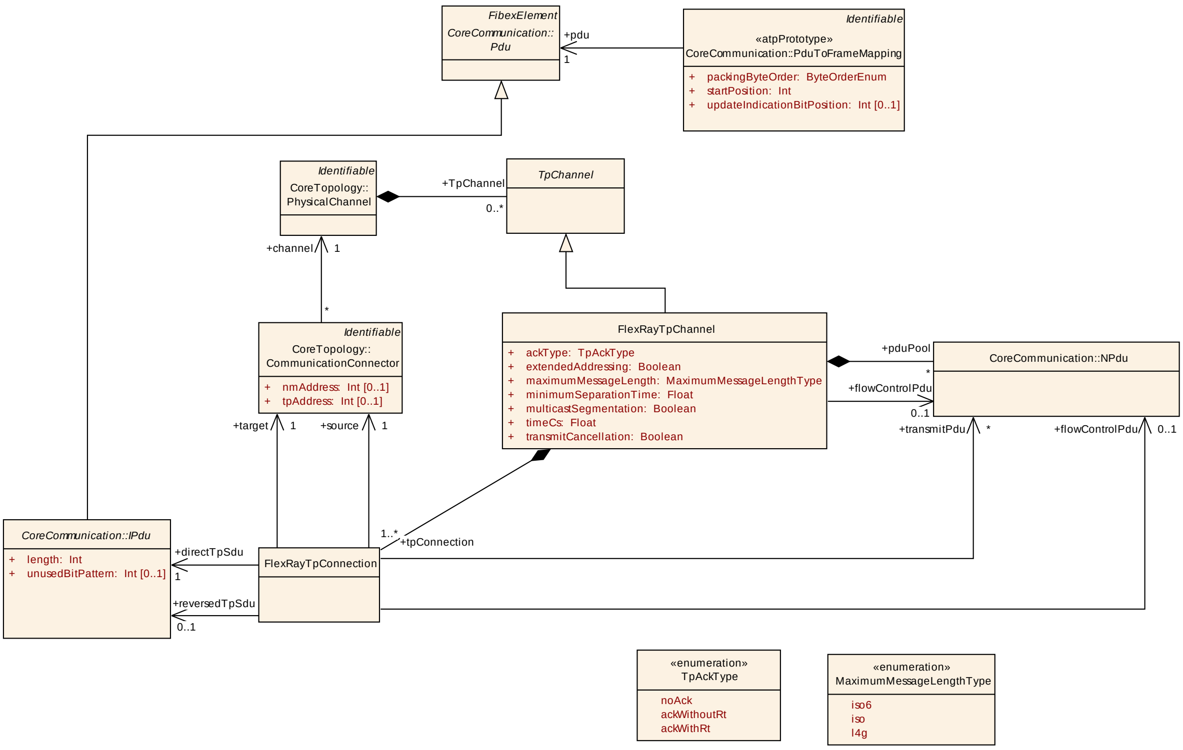

Figure 5.20: FlexRay Transport Layer Configuration (TransportProtocols: FlexRayTransportProtocol)

A `FlexRayTpChannel` provides a pool of `NPdus` which may are being used by the channel's `FlexRayTpConnections`: Each `FlexRayTpConnections` needs to specify at least on `NPdu` as transmit PDU; however, in order to achieve a higher band width the same connection may use more than one transmit `NPdu`.

As there is no concurrent transfer of connections within one channel, a flow control `NPdu` can be specified globally for the `FlexRayTpChannel`. In this case, all `FlexRayTpConnections` being realized by this channel use the same `NPdu` for Flow Control. However, this each `FlexRayTpConnections` may also define its own flow control `NPdu`.

`FlexRayTpConnections` are specifically used for communication between one source and one target device. These communication partners are specified using the `source` and `target` associations to `CommunicationControllers`, providing the diagnostic `tpAddress`.

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

Class	⟨⟨**atpObject**⟩⟩ **FlexRayTpChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Class Desc.	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 Class(es)	TpChannel			
Attribute	**Datatype**	**Mul.**	**Link Type**	**Description**
ackType	TpAckType	1	aggregation	Type of Acknowledgement.
extended Addressing	Boolean	1	aggregation	Adressing Type of this connection: true: Two Bytes false: One Byte
flowControl Pdu	NPdu	0..1	reference	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:
maximum Message Length	Maximum Message Length Type	1	aggregation	This specifies the maximum message length for the particular channel.
minimum Separation Time	Float	1	aggregation	This attribute defines the minimum amount of time (separation Time) between two succeeding CFs. Specified in seconds.
multicast Segmenta-tion	Boolean	1	aggregation	This attribute defines whether segmentation within a 1:n connection is allowed or not.
pduPool	NPdu	*	aggregation	A FlexRayTpChannel contains a pool of NPdus.

timeCs	Float	1	aggregation	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).
tpConnection	FlexRayTp Connection	1..*	aggregation	Group of connections that can be used in this channel.
transmit Cancellation	Boolean	1	aggregation	This attribute states whether Transmit Cancellation is supported on this channel.

Table 5.56: FlexRayTpChannel

Class	⟨⟨**atpObject**⟩⟩ **FlexRayTpConnection**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
directTp Sdu	IPdu	1	reference	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.

flowControlPdu	NPdu	0..1	reference	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.
reversedTpSdu	IPdu	0..1	reference	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	CommunicationConnector	1	reference	The source of the TP connection.
target	CommunicationConnector	1	reference	The target of the TP connection.

transmit Pdu	NPdu	*	reference	Reference to an NPdu (Single Frame, First Frame or Consecutive Frame). The Single Frame network protocol data unit (SF N_PDU) shall be sent out by the sending network entity and can be received by one or multiple receiving network entities. The Single Frame (SF N_PDU) shall be sent out to transfer a service data unit that can be transferred via a single service request to the data link layer. This network protocol data unit shall be sent to transfer unsegmented messages. The First Frame network protocol data unit (FF N_PDU) identifies the first network protocol data unit (N_PDU) of a segmented message transmitted by a network sending entity and received by a receiving network entity. The Consecutive Frame network protocol data unit (CF N_PDU) transfers segments (N_Data) of the service data unit message data (<MessageData>). All network protocol data units (N_PDUs) transmitted by the sending entity after the First Frame network protocol data unit (FF N_PDU) shall be encoded as Consecutive Frames network protocol data units (CF N_PDUs).

Table 5.57: FlexRayTpConnection

5.12.2 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 `CommunicationConnector`, providing the diagnostic `tpAddress`.

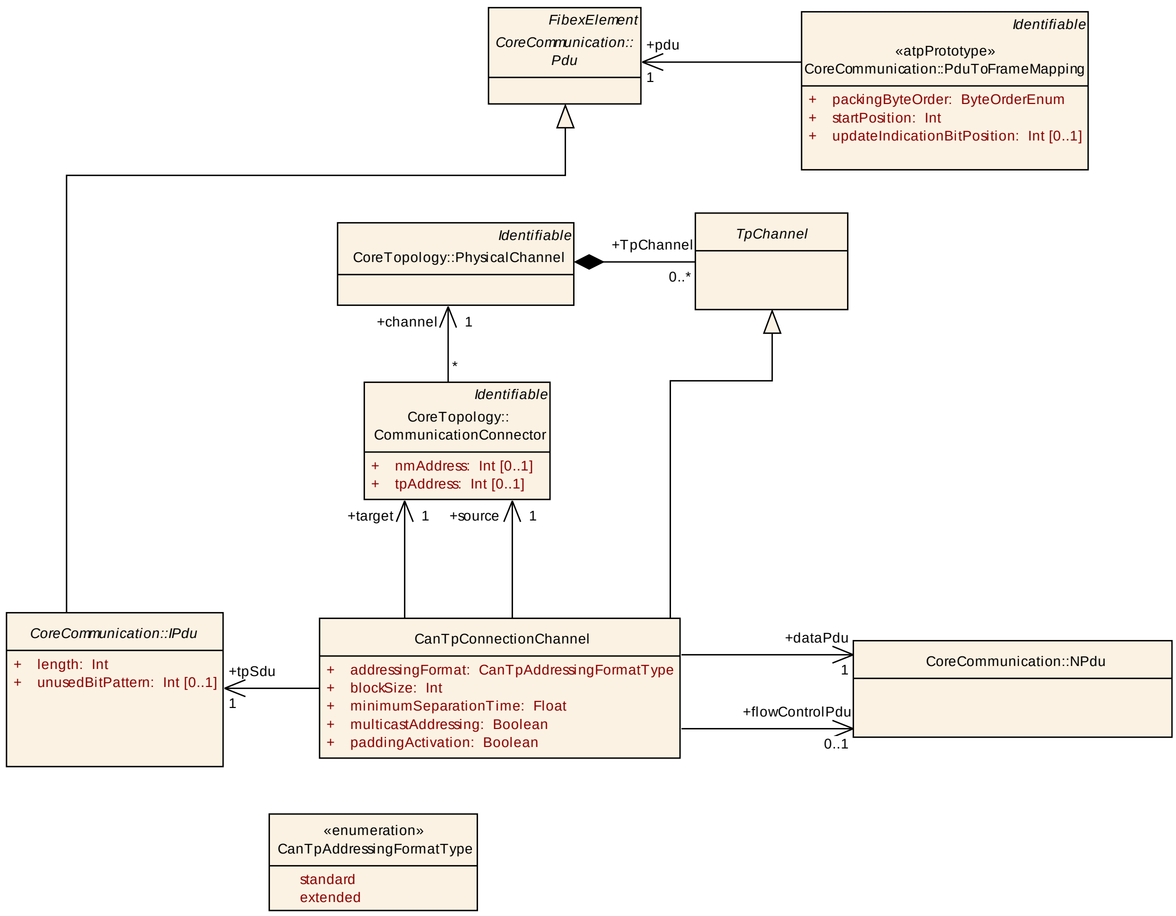

Figure 5.21: CAN Transport Layer Configuration (TransportProtocols: CanTransportProtocol)

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

Class	**⟨⟨atpObject⟩⟩ CanTpConnectionChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Class Desc.	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 Class(es)	TpChannel			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
addressing Format	CanTp Addressing Format Type	1	aggregation	Declares which communication addressing mode is supported.
blockSize	Integer	1	aggregation	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

dataPdu	NPdu	1	reference	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).
flowControl Pdu	NPdu	0..1	reference	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.
minimum Separation Time	Float	1	aggregation	This attribute defines the minimum amount of time (separation Time) between two succeeding CFs. Specified in seconds.

multicast Addressing	Boolean	1	aggregation	Specifies the communication type: true: 1:n communication (Functional) false: 1:1 communication (Physical)
padding Activation	Boolean	1	aggregation	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)
source	Comm unication Connector	1	reference	The source of the TP connection.
target	Comm unication Connector	1	reference	The target of the TP connection.
tpSdu	IPdu	1	reference	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 5.58: CanTpConnectionChannel

5.12.3 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 CommunicationConnector, providing the diagnostic tpAddress.

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

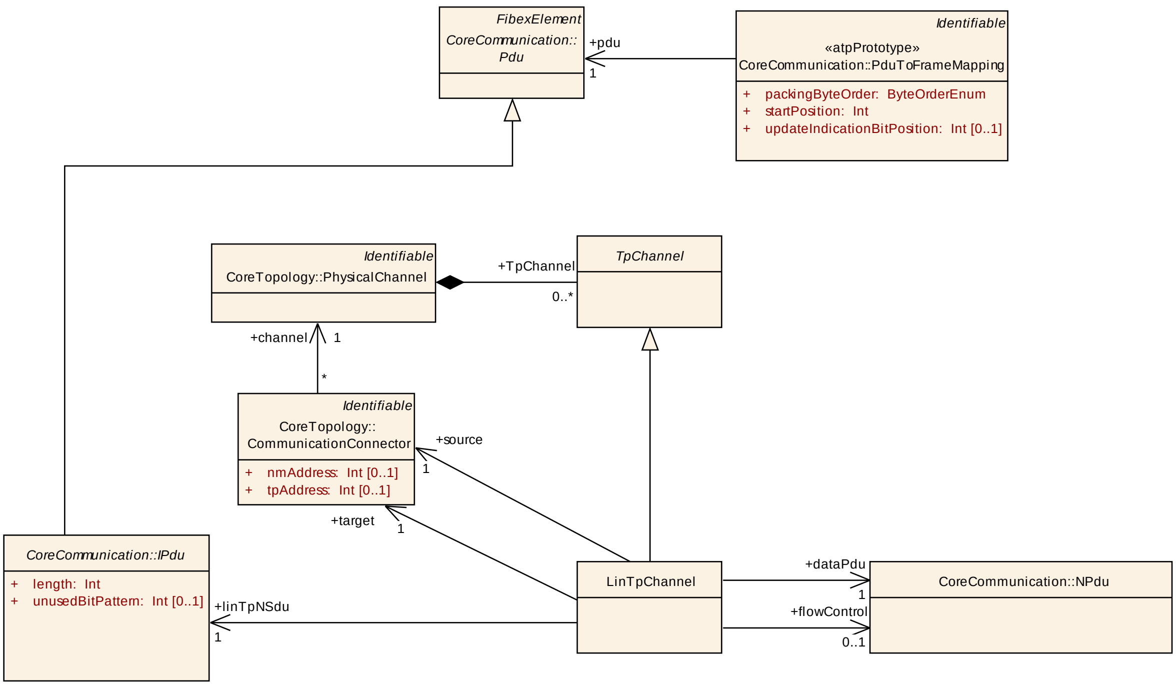

Figure 5.22: LIN Transport Layer Configuration (TransportProtocols: LinTransportProtocol)

Class	⟨⟨**atpObject**⟩⟩ **LinTpChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Class Desc.	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 Class(es)	TpChannel			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
dataPdu	NPdu	1	reference	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).

flowControl	NPdu	0..1	reference	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	reference	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.
source	Comm unication Connector	1	reference	The source of the TP connection.
target	Comm unication Connector	1	reference	The target of the TP connection.

Table 5.59: LinTpChannel

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 FrameTriggerings exist on the same cluster then the interface should handle the fan-out/in.

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 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.

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

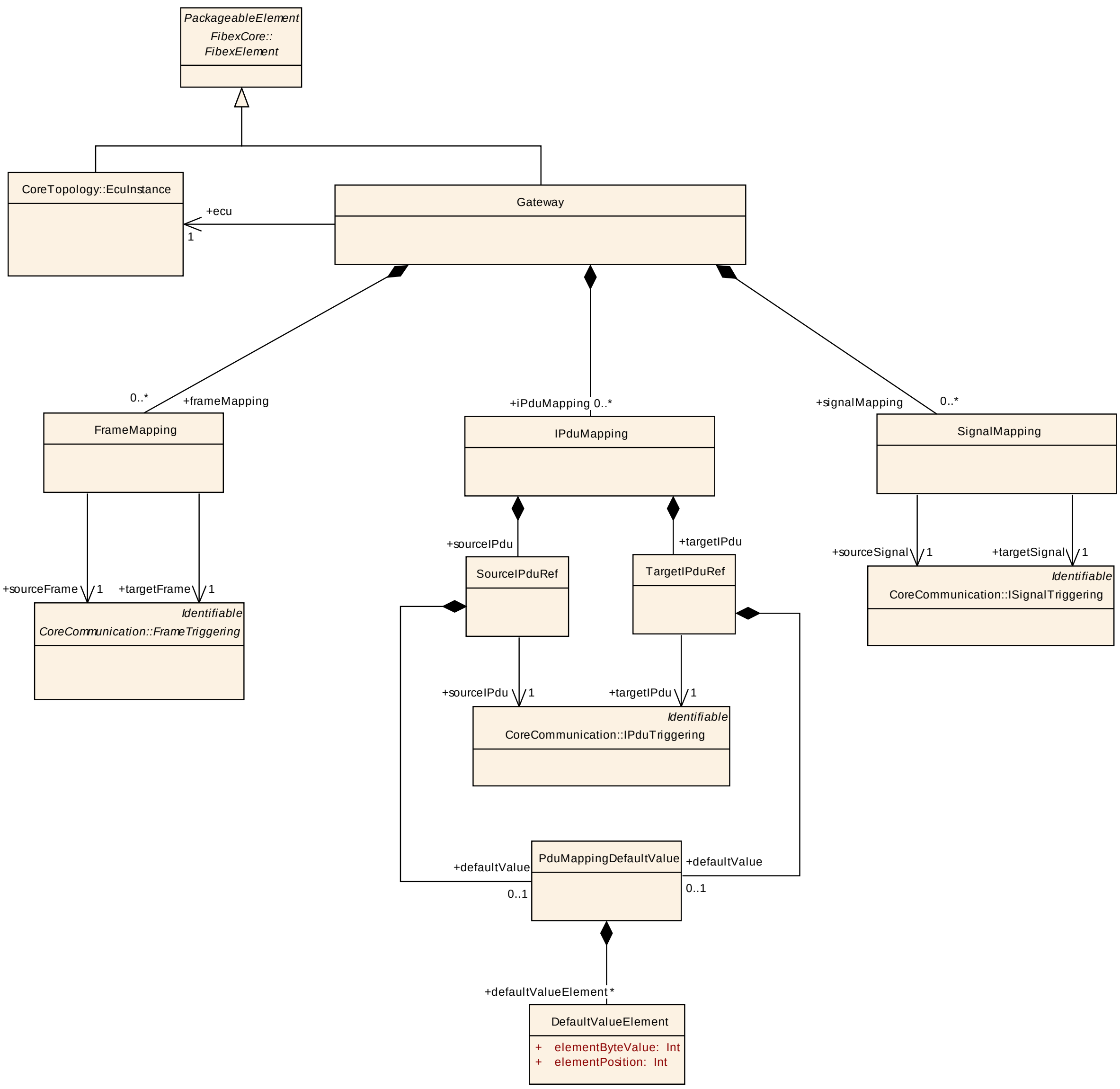

Figure 6.1: Communication Overview (Fibex4Multiplatform: Gateway)

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.

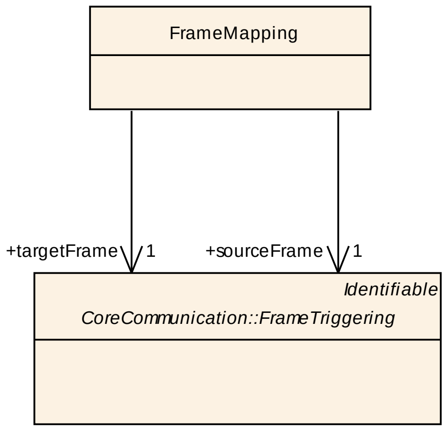

Figure 6.2: Frame Mapping (Fibex4Multiplatform: FrameMapping)

Class	⟨⟨**atpObject**⟩⟩ **FrameMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
source Frame	Frame Triggering	1	reference	Source destination of the referencing mapping.
target Frame	Frame Triggering	1	reference	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.

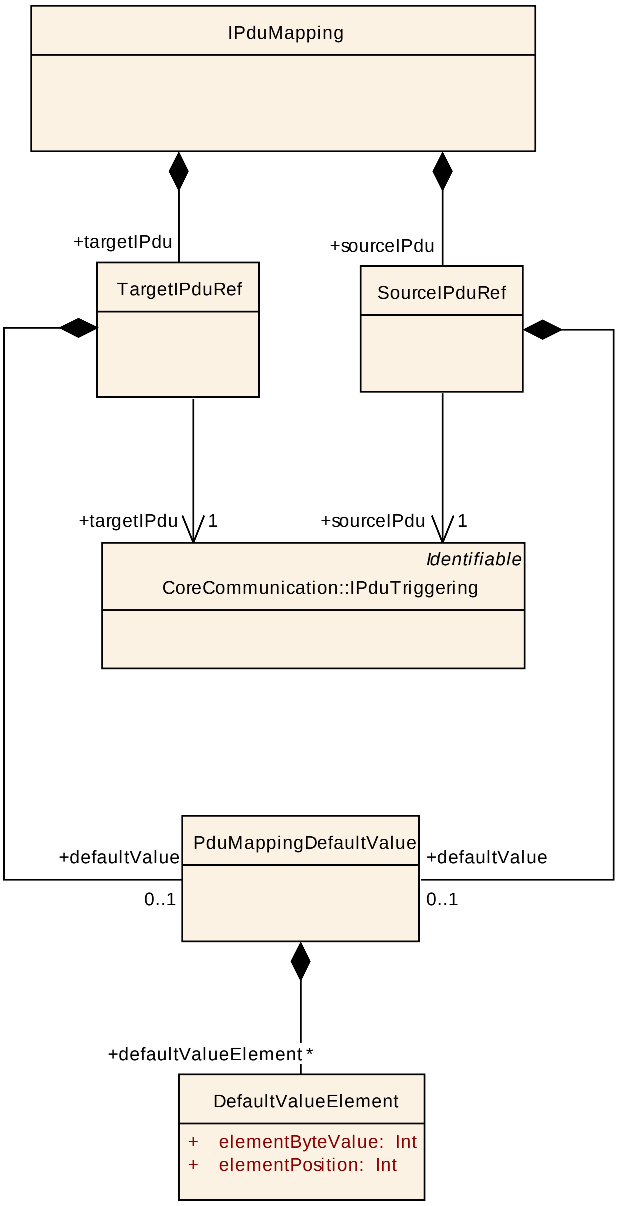

Figure 6.3: I-Pdu Mapping (Fibex4Multiplatform: IPduMapping)

Class	⟨⟨**atpObject**⟩⟩ **IPduMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Class Desc.	Arranges those IPdus that are transferred by the gateway from one channel to the other in pairs and defines the mapping between them.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
sourceIPdu	SourceIPduRef	1	aggregation	Source destination of the referencing mapping.
targetIPdu	TargetIPduRef	1	aggregation	Target destination of the referencing mapping.

Table 6.2: IPduMapping

Class	⟨⟨**atpObject**⟩⟩ **TargetIPduRef**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Class Desc.	Target destination of the referencing mapping.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
defaultValue	PduMappingDefaultValue	0..1	aggregation	If no I-Pdu has been received a default value will be distributed.
targetIPdu	IPduTriggering	1	reference	IPdu Reference

Table 6.3: TargetIPduRef

Class	⟨⟨**atpObject**⟩⟩ **SourceIPduRef**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Class Desc.	Source destination of the referencing mapping.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
defaultValue	PduMappingDefaultValue	0..1	aggregation	If no I-Pdu has been received a default value will be distributed.
sourceIPdu	IPduTriggering	1	reference	IPdu Reference

Table 6.4: SourceIPduRef

Class	⟨⟨atpObject⟩⟩ **PduMappingDefaultValue**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Class Desc.	Default Value which will be distributed if no I-Pdu has been received since last sending.			
Base Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
default ValueEle- ment	Default Value Element	*	aggregation	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	⟨⟨atpObject⟩⟩ **DefaultValueElement**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
element ByteValue	Integer	1	aggregation	The integer value of a freely defined data byte.
element Position	Integer	1	aggregation	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 that are transferred by the gateway from one channel to the other in pairs and defines the mapping between them in terms of some attributes describing the triggering behavior of the mapping. Each pair consists in a source and a target referencing to a `ISignalToPduMapping`.

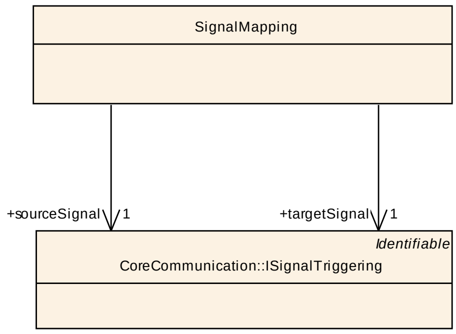

Figure 6.4: Signal Mapping (Fibex4Multiplatform: Signal Mapping)

Class	**⟨⟨atpObject⟩⟩ SignalMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Class Desc.	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 Class(es)	ARObject			
Attribute	***Datatype***	***Mul.***	***Link Type***	***Description***
source Signal	ISignal Triggering	1	reference	Source destination of the referencing mapping.
targetSig-nal	ISignal Triggering	1	reference	Target destination of the referencing mapping.

Table 6.7: SignalMapping

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 there is an ISignal included that references to this SystemSignalGroup
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 TransmissionModeFalseTiming 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.

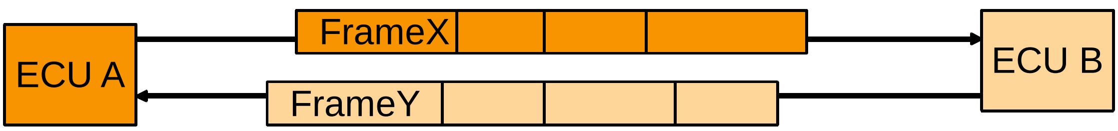

Figure 8.1: Example topology with two ECUs and two frames exchanged between them

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'.

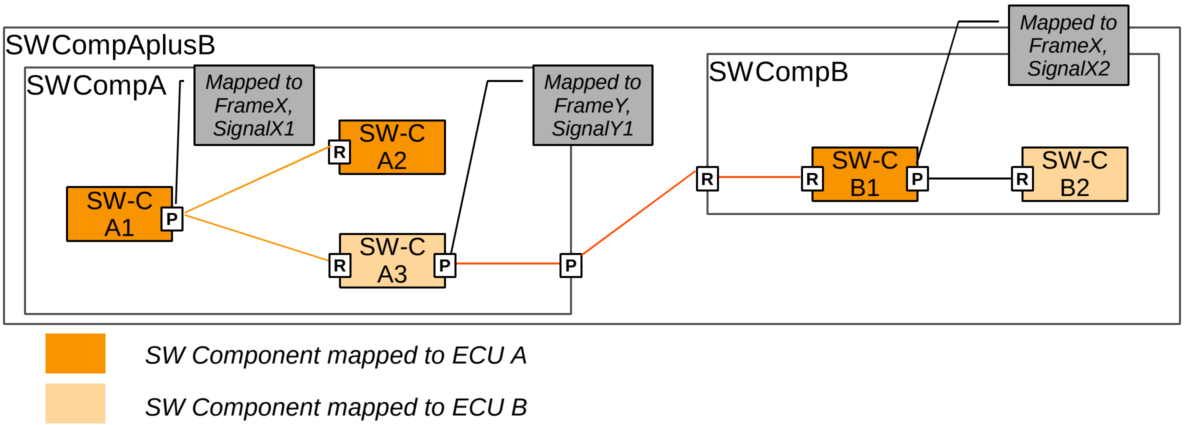

Figure 8.2: Example SW composition with mapping information

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.

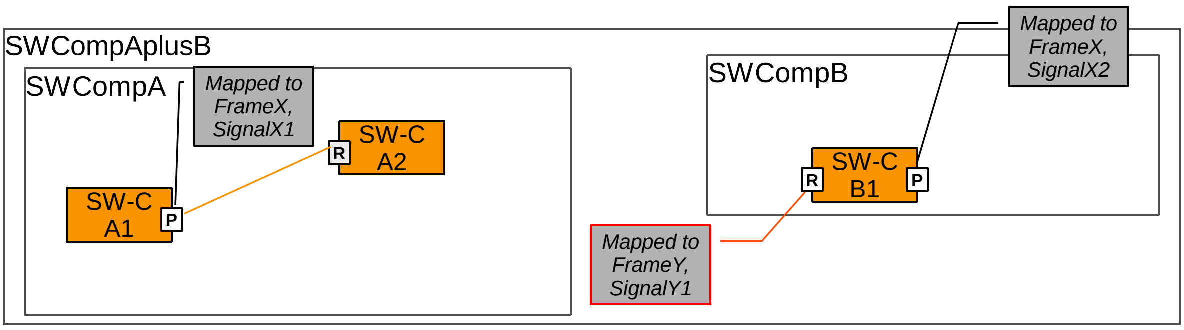

Figure 8.3: Example ECU extract for ECU A of above introduced composition

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'.

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/ComConfig	
BSW Parameter		**BSW Type**
ComConfig		PARAM-CONF-CONTAINER-DEF
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		INTEGER-PARAM-DEF
BSW Description		
This ID is returned by a call to Com_GetConfigurationId.		
M2 Template	**M2 Description**	
System Template	This ID is returned by a call to Com_GetConfigurationId()	
M2 Parameter		
CoreTopology::ECUInstance::ComConfigurationID		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComGwMapping	
BSW Parameter		**BSW Type**
ComGwMapping		PARAM-CONF-CONTAINER-DEF
BSW Description		
Each instance of this container defines one mapping of the integrated Signal Gateway.		
M2 Template	**M2 Description**	
System Template	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 ISignalToIPduMapping.	
M2 Parameter		
Fibex4Multiplatform::SignalMapping		
Mapping Rule		**Mapping Type**
create container if ECU contains a SignalMapping Gatweway		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestinationDescription	
BSW Parameter		**BSW Type**
ComGwDestinationDescription		PARAM-CONF-CONTAINER-DEF
BSW Description		
Description of a gateway destination. This container allows to define a gateway destination without the configuration of a complete COM signal. This allows to add / change gateway relations postbuild without the configuration of new signals.		
M2 Template	**M2 Description**	
System Template	Target destination of the referencing mapping.	
M2 Parameter		
Fibex4Multiplatform::SignalMapping.targetSignal		
Mapping Rule		**Mapping Type**
The SignalMapping contains a reference to the ISignalToIPduMapping element.		local

BSW Module	BSW Context	
Com	Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestinationDescription	
BSW Parameter		**BSW Type**
ComBitPosition		INTEGER-PARAM-DEF
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**	
System Template	This parameter is necessary to describe the bitposition of a signal within a IPdu or SubPDU. 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). Bits within the IPdu are counted as follows (see the OSEK COM v3.0.3 specification) : Bit 0 corresponds to Byte 0 Bit 0 Bit 1 corresponds to Byte 0 Bit 1 ..... Bit 8 corresponds to Byte 1 Bit 0 etc. 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. Note also that the absolute position of the signal in the IPdu is then determined by the definition of the packingByteOrder attribute of the signal.	
M2 Parameter		
CoreCommunication::ISignalToPduMapping.startPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestinationDescription</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComSignalDataInvalidValue</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3">A constant of a primitive datatype.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">DataTypes:PrimitiveType:SwDataDefProps.invalidValue</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>Com</td><td colspan="3">Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestinationDescription</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">INTEGER-PARAM-DEF</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 arrays last byte. The second-least significant byte shall be assigned to the byte arrays last but one byte, and so on.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3">Depending on Rx/Tx, use one of the two ComSpecs above.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">Communcation::UnqueuedReceiverComSpec.initValue OR Communcation::UnqueuedSenderComSpec.initValue</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>Com</td><td colspan="3">Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestinationDescription</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComUpdateBitPosition</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">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 IPdu.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">CoreCommunication::ISignalToIPduMapping.updateIndicationBitPosition</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>

BSW Module	BSW Context		
Com	Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestinationDescription		
BSW Parameter		**BSW Type**	
ComSignalEndianess		ENUMERATION-PARAM-DEF	
BSW Description			
Defines the endianness of the signal's network representation.			
M2 Template	**M2 Description**		
System Template	This parameter defines the order of the bytes of the signal and the packing into the IPdu. 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 IPdu (see the startPosition attribute description).		
M2 Parameter			
CoreCommunication::ISignalToIPduMapping.packingByteOrder			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Com	Com/ComConfig/ComGwMapping/ComGwDestination/ComGwDestinationDescription		
BSW Parameter		**BSW Type**	
ComTransferProperty		ENUMERATION-PARAM-DEF	
BSW Description			
Derived from [18].			
M2 Template	**M2 Description**		
System Template	The triggered transfer property causes immediate transmission of the IPdu, except if transmission mode Periodic or transmission mode NONE is defined for the IPdu. The Pending transfer property does not cause transmission of an I-PDU.		
M2 Parameter			
CoreCommunication::ISignalToIPduMapping.transferProperty			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Com	Com/ComConfig/ComGwMapping/ComGwDestination/ComGwSignal		
BSW Parameter		**BSW Type**	
ComGwSignal		PARAM-CONF-CONTAINER-DEF	
BSW Description			
M2 Template	**M2 Description**		
System Template	An ISignalToIPduMapping describes the mapping of ISignals to IPdus and defines the position of the ISignal within an IPdu. This element does NOT describe signal or I-PDU fan-out. Every ISignal can only be mapped into one IPdu. Several ISignalToPduMappings to the same ISignal are only relevant when the ECU handles the signal gateway. If a mapping for the SystemSignalGroup is defined, only the UpdateIndicationBitPosition is relevant, and the startPosition shall be ignored.		
M2 Parameter			
CoreCommunication::ISignalToIPduMapping			
Mapping Rule			**Mapping Type**

ISignal contains an ISignal, ComGroupSignal or a SystemSignalGroup.	local

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComGwMapping/ComGwSource/ComGwSignal</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComGwSignal</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3"></td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">An ISignalToIPduMapping describes the mapping of ISignals to IPdus and defines the position of the ISignal within an IPdu. This element does NOT describe signal or I-PDU fan-out. Every ISignal can only be mapped into one IPdu. Several ISignalToPduMappings to the same ISignal are only relevant when the ECU handles the signal gateway. If a mapping for the SystemSignalGroup is defined, only the UpdateIndicationBitPosition is relevant, and the startPosition shall be ignored.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">CoreCommunication::ISignalToIPduMapping</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">ISignal contains an ISignal, ComGroupSignal or a SystemSignalGroup.</td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComGwMapping/ComGwSource/ComGwSourceDescription</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComGwSourceDescription</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Description of a gateway source. This container allows to define a gateway source without the configuration of a complete COM signal. This allows to add / change gateway relations postbuild without the configuration of new signals.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">Source destination of the referencing mapping.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex4Multiplatform::SignalMapping.sourceSignal</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">SignalMapping contains a reference to the ISignalToIPduMapping element.</td><td>local</td></tr>
</table>

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComGwMapping/ComGwSource/ComGwSourceDescription	
BSW Parameter		**BSW Type**
ComBitPosition		INTEGER-PARAM-DEF
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**	
System Template	This parameter is necessary to describe the bitposition of a signal (or signalGroup) within an IPdu. 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). Bits within the IPdu are counted as follows (see the OSEK COM v3.0.3 specification) : Bit 0 corresponds to Byte 0 Bit 0 Bit 1 corresponds to Byte 0 Bit 1 ..... Bit 8 corresponds to Byte 1 Bit 0 etc. 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. Note also that the absolute position of the signal in the IPdu is then determined by the definition of the packingByteOrder attribute of the signal.	
M2 Parameter		
CoreCommunication::ISignalToIPduMapping.startPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComGwMapping/ComGwSource/ComGwSourceDescription	
BSW Parameter		**BSW Type**
ComBitSize		INTEGER-PARAM-DEF
BSW Description		
Size in bits.		
M2 Template	**M2 Description**	
System Template	Size of the signal in bits.	
M2 Parameter		
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**	
ComUpdateBitPosition		INTEGER-PARAM-DEF	
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**		
System Template	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 IPdu.		
M2 Parameter			
CoreCommunication::ISignalToIPduMapping.updateIndicationBitPosition			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Com	Com/ComConfig/ComGwMapping/ComGwSource/ComGwSourceDescription		
BSW Parameter		**BSW Type**	
ComSignalEndianess		ENUMERATION-PARAM-DEF	
BSW Description			
Defines the endianness of the signal's network representation.			
M2 Template	**M2 Description**		
System Template	This parameter defines the order of the bytes of the signal and the packing into the IPdu. 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 IPdu (see the startPosition attribute description).		
M2 Parameter			
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		ENUMERATION-PARAM-DEF	
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**		
SW Component Template	Abstract base class for user defined (and AUTOSAR predefined) datatypes.		
M2 Parameter			
SWC Template::DataTypes			
Mapping Rule			**Mapping Type**
Mapping of AUTOSAR data types (defined in the software component template) to COMSignalTypes			local

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComIPdu</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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. Is optional because the Com module might be used for internal communication only. 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>System Template</td><td colspan="2">Represents the I-PDU's handled by Com. The IPdu assembled and disassembled in AUTOSAR COM consists of one or more signals. In case no multiplexing is performed this IPdu is routed to/from the Interface Layer.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">CoreCommunication::SignalIPdu</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">One container per CoreCommunication::SignalPdu</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComIPduRxHandleId</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">The numerical value used as the ID of this I-PDU. The 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.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3"></td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComIPduSize</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">The size of the PDU in bits. The size is limited by the frameLength.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">CoreCommunication::IPdu::length</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	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComIPduSignalProcessing		ENUMERATION-PARAM-DEF
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**
ComIpduDirection		ENUMERATION-PARAM-DEF
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**	
System Template	communication Direction of the Connector Port (input or output Port).	
M2 Parameter		
CommConnectorPort.communicationDirection		
Mapping Rule		**Mapping Type**
An CommConnectorPort (IPduPort) contains a reference to the IPduTriggering. If the communicationDirection of the CommConnectorPort is ”in” than the IPdu is received. If the communicationDirection of the CommConnectorPort is ”out” than the IPdu is transmitted.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComIPduCallout		FUNCTION-NAME-DEF
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		REFERENCE-PARAM-DEF
BSW Description		
Reference to the I-PDU group this I-PDU belongs to.		
M2 Template	**M2 Description**	
System Template	Reference to a set of PDUs, which are contained in the PDU Group.	
M2 Parameter		
CoreCommunication::IPduGroup		
Mapping Rule		**Mapping Type**
Find CoreCommunication::IPduGroup that points to this IPdu		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComIPduSignalGroupRef		REFERENCE-PARAM-DEF
BSW Description		
References to all signal groups contained in this I-Pdu		
M2 Template	**M2 Description**	
System Template	Reference to a set of signals that must always be kept together.	
M2 Parameter		
CoreCommunication::SystemSignalGroup::containedSignals		
Mapping Rule		**Mapping Type**
Find ISignal in the IPdu that refers to this SystemSignalGroup and create reference		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComIPduSignalRef		REFERENCE-PARAM-DEF
BSW Description		
References to all signals contained in this I-PDU.		
M2 Template	**M2 Description**	
System Template	An ISignalToIPduMapping describes the mapping of ISignals to IPdus and defines the position of the ISignal within an IPdu. This element does NOT describe signal or I-PDU fan-out. Every ISignal can only be mapped into one IPdu. Several ISignalToPduMappings to the same ISignal are only relevant when the ECU handles the signal gateway. If a mapping for the SystemSignalGroup is defined, only the UpdateIndicationBitPosition is relevant, and the startPosition shall be ignored.	
M2 Parameter		
CoreCommunication::ISignalToPduMapping		
Mapping Rule		**Mapping Type**
Find Ipdu and create reference for each contained ISignalToPduMapping.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
PduIdRef		REFERENCE-PARAM-DEF
BSW Description		
Reference to the "global" Pdu structure to allow harmonization of handle IDs in the COM-Stack.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu	
BSW Parameter		**BSW Type**
ComTxIPdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains additional transmission related configuration parameters of COM I-PDUs		
M2 Template	**M2 Description**	
System Template	Represents the I-PDU's handled by Com. The IPdu assembled and disassembled in AUTOSAR COM consists of one or more signals. In case no multiplexing is performed this IPdu is routed to/from the Interface Layer.	
M2 Parameter		
CoreCommunication::SignalPdu		
Mapping Rule		**Mapping Type**
create container for each transmitted IPdu		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu	
BSW Parameter		**BSW Type**
ComTxIPduMinimumDelayTimeFactor		INTEGER-PARAM-DEF
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**	
System Template	Minimum Delay in seconds between successive transmissions of this I-PDU, independent of the Transmission Mode.	
M2 Parameter		
CoreCommunication::IPduTriggering:minimumDelay		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu	
BSW Parameter		**BSW Type**
ComTxIPduUnusedAreasDefault		INTEGER-PARAM-DEF
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**	
System Template	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		
CoreCommunication::SignalPdu:unusedBitPattern		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse	
BSW Parameter		**BSW Type**
ComTxModeFalse		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the configuration parameters of COM transmission modes in the case the ComFilter evaluates to false.		
M2 Template	**M2 Description**	
System Template	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 IPduTriggering 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)	
M2 Parameter		
CoreCommunication::SignalPdu::IPduTiming::TransmissionModeDeclaration::TransmissionModeFalseTiming		
Mapping Rule		**Mapping Type**
Find transmissionModeFalseTiming and create this container		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse/ComTxMode	
BSW Parameter		**BSW Type**
ComTxMode		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the configuration parameters of COM transmission modes.		
M2 Template	**M2 Description**	
System Template	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 IPduTriggering 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)	
M2 Parameter		
CoreCommunication::SignalPdu::IPduTiming::TransmissionModeDeclaration::TransmissionModeFalseTiming		
Mapping Rule		**Mapping Type**
Find transmissionModeFalseTiming and create this container		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeNumberOfRepetitions		INTEGER-PARAM-DEF
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**	
System Template	Number of repetitions the pdu is sent from the moment the send condition has been breached.	
M2 Parameter		
TransmissionModeDeclaration::TransmissionModeFalseTiming::EventControlledTiming.numberOfRepeats		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeRepetitionPeriodFactor		INTEGER-PARAM-DEF
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**	
System Template	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		
TransmissionModeDeclaration::TransmissionModeFalseTiming::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		INTEGER-PARAM-DEF
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		
TransmissionModeDeclaration::TransmissionModeFalseTiming::CyclicTiming:StartingTimeRange		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeTimePeriodFactor		INTEGER-PARAM-DEF
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**	
System Template	Specification of the repeating cycle in seconds whenever the pdu described by this timing is sent.	
M2 Parameter		
TransmissionModeDeclaration::TransmissionModeFalseTiming::CyclicTiming.RepeatingTimeRange		
Mapping Rule		**Mapping Type**
1:1 mapping		full

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse/ComTxMode</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComTxModeMode</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">The available transmission modes described in [18] shall be extended by the additional mode None. The transmission mode None shall not have any further sub-attributes in the ComTxMode object.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">AUTOSAR COM provides the possibility to define two different TRANSMISSION MODES (True and False) for each I-PDU.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">IPduTiming::TransmissionModeDeclaration</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">If no timing is assigned the transmission mode ”none” must be set.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComTxModeTrue</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This container contains the configuration parameters of COM transmission modes in the case the ComFilter evaluates to true.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">If the COM Transmission Mode is true the timing can be aggregated directly by the Ipdu.IPduTiming.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">IPduTiming::CyclicTiming and IPduTiming::EventControlledTiming</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">container must be created if timing is aggregated by the IPdu.IPduTiming element</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComTxMode</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This container contains the configuration parameters of COM transmission modes.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">If the COM Transmission Mode is true the timing can be aggregated directly by the IPdu.IPduTiming</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">IPduTiming::CyclicTiming and IPduTiming::EventControlledTiming</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">container must be created if timing is aggregated by the IPdu.IPduTiming element</td><td>full</td></tr>
</table>

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeNumberOfRepetitions		INTEGER-PARAM-DEF
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**	
System Template	Number of repetitions the pdu is sent from the moment the send condition has been breached.	
M2 Parameter		
EventControlledTiming:numberOfRepeats		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeRepetitionPeriodFactor		INTEGER-PARAM-DEF
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**	
System Template	Specification of the time in seconds that elapses before the pdu can be sent the next time (Minimum repeat gap between two pdus).	
M2 Parameter		
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		INTEGER-PARAM-DEF
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**	
System Template	Specification of the time that is needed before the pdu can be sent the first time.	
M2 Parameter		
CyclicTiming:StartingTimeRange		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeTimePeriodFactor		INTEGER-PARAM-DEF
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**	
System Template	Specification of the repeating cycle in seconds whenever the pdu described by this timing is sent.	
M2 Parameter		
Timing:CyclicTiming:RepeatingTimeRange		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeMode		ENUMERATION-PARAM-DEF
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**	
System Template	AUTOSAR COM provides the possibility to define two different TRANSMISSION MODES for each I-PDU. COM supports the following Transmission Modes: Periodic (Cyclic Timing) Direct /n-times (EventControlledTiming) Mixed (Cyclic and EventControlledTiming are assigned) None (no timing is assigned)	
M2 Parameter		
Timing is aggregated by the IPduTiming		
Mapping Rule		**Mapping Type**
If no timing is assigned to the TransmissionMode element, the Transmission mode none should be used		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPduGroup	
BSW Parameter		**BSW Type**
ComIPduGroup		PARAM-CONF-CONTAINER-DEF
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 (ComIpduGroupName). This parameter is only stored in the XML file, and must not be used within the implementation.		
M2 Template	**M2 Description**	
System Template	An PDU group contains zero or more PDUs or PDU groups.	
M2 Parameter		
CoreCommunication::IPduGroup		
Mapping Rule		**Mapping Type**
One container per CoreCommunication::IPduGroup		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPduGroup	
BSW Parameter		**BSW Type**
ComIPduGroupHandleId		INTEGER-PARAM-DEF
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/ComIPduGroup	
BSW Parameter		**BSW Type**
ComIPduGroupGroupRef		REFERENCE-PARAM-DEF
BSW Description		
If the I-PDU Group belongs to an I-PDU group, this is the name of the I-PDU group it belongs to. This I-PDU Group does not belong to another I-PDU group, if this reference is omitted.		
M2 Template	**M2 Description**	
System Template	An I-PDU group can be included in other PDU groups.	
M2 Parameter		
CoreCommunication::IPduGroup::containedPduGroups		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComSignal		PARAM-CONF-CONTAINER-DEF
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**	
System Template	An ISignalToIPduMapping describes the mapping of ISignals to IPdus and defines the position of the ISignal within an IPdu. This element does NOT describe signal or I-PDU fan-out. Every ISignal can only be mapped into one IPdu. Several ISignalToPduMappings to the same ISignal are only relevant when the ECU handles the signal gateway. If a mapping for the SystemSignalGroup is defined, only the UpdateIndicationBitPosition is relevant, and the startPosition shall be ignored.	
M2 Parameter		
CoreCommunication::ISignalToPduMapping		
Mapping Rule		**Mapping Type**
A Com signal must be defined in the Com module configuration for each ISignalToPduMapping that is transmitted or received by the regarded ECU.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComBitPosition		INTEGER-PARAM-DEF
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**	
System Template	This parameter is necessary to describe the bitposition of a signal (or signal-Group) within an IPdu. 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). Bits within the IPdu are counted as follows (see the OSEK COM v3.0.3 specification) : Bit 0 corresponds to Byte 0 Bit 0 Bit 1 corresponds to Byte 0 Bit 1 ..... Bit 8 corresponds to Byte 1 Bit 0 etc. 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. Note also that the absolute position of the signal in the IPdu is then determined by the definition of the packingByteOrder attribute of the signal.	
M2 Parameter		
CoreCommunication::ISignalToIPduMapping.startPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComBitSize		INTEGER-PARAM-DEF
BSW Description		
Size in bits.		
M2 Template	**M2 Description**	
System Template	Size of the signal in bits.	
M2 Parameter		
CoreCommunication::SystemSignal.length		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComFirstTimeoutFactor		INTEGER-PARAM-DEF
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		INTEGER-PARAM-DEF
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**
ComSignalDataInvalidValue		INTEGER-PARAM-DEF
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**	
SW Component Template	A constant of a primitive datatype.	
M2 Parameter		
DataTypes:PrimitiveType:SwDataDefProps.invalidValue		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComSignalInitValue		INTEGER-PARAM-DEF
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 arrays last byte. The second-least significant byte shall be assigned to the byte arrays last but one byte, and so on.		
M2 Template	**M2 Description**	
SW Component Template		
M2 Parameter		
Communcation::UnqueuedReceiverComSpec.initValue OR Communcation::UnqueuedSenderComSpec.initValue		
Mapping Rule		**Mapping Type**
Depending on Rx/Tx, use one of the two ComSpecs above.		full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignal
BSW Parameter	**BSW Type**
ComSignalLength	INTEGER-PARAM-DEF
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**
System Template	Size of the signal in bits.
M2 Parameter	
CoreCommunication::SystemSignal::length	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignal
BSW Parameter	**BSW Type**
ComTimeoutFactor	INTEGER-PARAM-DEF
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 timeout 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**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignal
BSW Parameter	**BSW Type**
ComUpdateBitPosition	INTEGER-PARAM-DEF
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**
System Template	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 IPdu.
M2 Parameter	
ISignalToPduMapping::updateIndicationBitPosition	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComDataInvalidAction		ENUMERATION-PARAM-DEF
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**	
SW Component Template	Specifies strategy of handling the reception of invalidValue (if datatype of the dataelement is complex, the dataelement will be mapped into a signal group). keep: Keep a received invalidValue. This allows handling of Signal Invalidation on RTE API level either by DataReceiveErrorEvent or return of an error code on on read access. replace: Replace a received invalidValue. The replacement value is specified by the initValue.	
M2 Parameter		
Communication:ReceiverComSpec:UnqueuedReceiverComSpec:handleInvalid		
Mapping Rule		**Mapping Type**
If strategy keep is defined than set parameter to notify. If strategy replace is defined than set parameter to replace.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComRxDataTimeoutAction		ENUMERATION-PARAM-DEF
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

<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">ComSignalEndianess</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Defines the endianness of the signal's network representation.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">This parameter defines the order of the bytes of the signal and the packing into the PDU. 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 PDU (see the startPosition attribute description)</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">ISignalToPduMapping::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="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">ComSignalType</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="2">Abstract base class for user defined (and AUTOSAR predefined) datatypes.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">SWC Template::DataTypes</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">Mapping of AUTOSAR data types (defined in the software component template) to COMSignalTypes</td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComSignal</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComTransferProperty</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Derived from [18].</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>

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComErrorNotification		FUNCTION-NAME-DEF
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**
ComInvalidNotification		FUNCTION-NAME-DEF
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		FUNCTION-NAME-DEF
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**
ComTimeoutNotification		FUNCTION-NAME-DEF
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**
SystemTemplateSystemSignalRef		FOREIGN-REFERENCE-PARAM-DEF
BSW Description		
Reference to the SystemSignalToPduMapping 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**
		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilter		PARAM-CONF-CONTAINER-DEF
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**	
SW Component Template	Base class for data filters.	
M2 Parameter		
ReceiverComSpec.filter		
Mapping Rule		**Mapping Type**
		full

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComConfig/ComSignal/ComFilter</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComFilterMask</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3">Base class for data filters.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">ReceiverComSpec.filter</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">If ReceiverComSpec.filter has a mask attribute (MaskedNewEqualsX, MaskedNewDiffersX, MaskedNewEqualsMaskedOld, MaskedNewDiffersMaskedOld) then create this element using the mask value.</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/ComSignal/ComFilter</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComFilterMax</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3">Base class for data filters.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">ReceiverComSpec.filter</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">If ReceiverComSpec.filter has a max attribute (NewIsWithing, NewIsOutside) then create this element using the max value.</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/ComSignal/ComFilter</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComFilterMin</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3">Base class for data filters.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">ReceiverComSpec.filter</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">If ReceiverComSpec.filter has a min attribute (NewIsWithing, NewIsOutside) then create this element using the min value.</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/ComSignal/ComFilter</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComFilterOffset</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant. Range = 0..(ComFilterPeriodFactor-1)</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3">specifies the initial number of messages to occur before the first message is passed</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">OnEveryN.offset</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>Com</td><td colspan="3">Com/ComConfig/ComSignal/ComFilter</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComFilterPeriodFactor</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 are significant.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3">specifies number of messages to occur before the message is passed again</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">OnEveryN.period</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>Com</td><td colspan="3">Com/ComConfig/ComSignal/ComFilter</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComFilterX</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3">Base class for data filters</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">ReceiverComSpec.filter</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">If ReceiverComSpec.filter has an x attribute (MaskedNewEqualsX, MaskedNewDiffersX) then create this element using the x value.</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/ComFilter</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComFilterAlgorithm</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">The range of values is specified in the [17] specification, chapter 2.2.2, Reception Filtering.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="2">Base class for data filters</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">ReceiverComSpec.filter</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">Type of DataFilter, i.e. Always, Never, MakedNewEqualsX, ...</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">ComSignalGroup</td><td>PARAM-CONF-CONTAINER-DEF</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>System Template</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">CoreCommunication::SystemSignalGroup</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">If CoreCommunication::SystemSignalGroup exists create this container.</td><td>full</td></tr>
</table>

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup	
BSW Parameter		**BSW Type**
ComBitPosition		INTEGER-PARAM-DEF
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**	
System Template	This parameter is necessary to describe the bitposition of a signal within an IPdu. 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). Bits within the IPdu are counted as follows (see the OSEK COM v3.0.3 specification) : Bit 0 corresponds to Byte 0 Bit 0 Bit 1 corresponds to Byte 0 Bit 1 ..... Bit 8 corresponds to Byte 1 Bit 0 etc. 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. Note also that the absolute position of the signal in the IPdu is then determined by the definition of the packingByteOrder attribute of the signal.	
M2 Parameter		
CoreCommunication::ISignalToIPduMapping.startPosition		
Mapping Rule		**Mapping Type**
Starting Position of the ComSignalGroup can be derived from the starting Position of the first Signal in the group		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup	
BSW Parameter		**BSW Type**
ComBitSize		INTEGER-PARAM-DEF
BSW Description		
Size in bits.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
SystemSignal.length		
Mapping Rule		**Mapping Type**
Add the length of all Com Group signals together.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup	
BSW Parameter		**BSW Type**
ComFirstTimeoutFactor		INTEGER-PARAM-DEF
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**	
ComHandleId		INTEGER-PARAM-DEF	
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**	
ComTimeoutFactor		INTEGER-PARAM-DEF	
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 timeout 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**		
System Template	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.		
M2 Parameter			
CoreCommunication::SignalPort.timeout			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup	
BSW Parameter		**BSW Type**
ComUpdateBitPosition		INTEGER-PARAM-DEF
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**	
System Template	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 IPdu.	
M2 Parameter		
CoreCommunication::ISignalToIPduMapping.updateIndicationBitPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup	
BSW Parameter		**BSW Type**
ComDataInvalidAction		ENUMERATION-PARAM-DEF
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**	
SW Component Template	Specifies strategy of handling the reception of invalidValue (if datatype of the dataelement is complex, the dataelement will be mapped into a signal group). keep: Keep a received invalidValue. This allows handling of Signal Invalidation on RTE API level either by DataReceiveErrorEvent or return of an error code on on read access. replace: Replace a received invalidValue. The replacement value is specified by the initValue.	
M2 Parameter		
Communication:ReceiverComSpec:UnqueuedReceiverComSpec:handleInvalid		
Mapping Rule		**Mapping Type**
If strategy keep is defined than set parameter to notify. If strategy replace is defined than set parameter to replace.		full

BSW Module	BSW Context		
Com	Com/ComConfig/ComSignalGroup		
BSW Parameter		**BSW Type**	
ComRxDataTimeoutAction		ENUMERATION-PARAM-DEF	
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**	
ComTransferProperty		ENUMERATION-PARAM-DEF	
BSW Description			
Derived from [18].			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
Com	Com/ComConfig/ComSignalGroup		
BSW Parameter		**BSW Type**	
ComErrorNotification		FUNCTION-NAME-DEF	
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**
ComInvalidNotification		FUNCTION-NAME-DEF
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		FUNCTION-NAME-DEF
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**
ComTimeoutNotification		FUNCTION-NAME-DEF
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**
SystemTemplateSignalGroupRef		FOREIGN-REFERENCE-PARAM-DEF
BSW Description		
Reference to the SystemSignalToPduMapping that contains a reference to the ISignal (SystemTemplate) which this ComSignalGroup represents.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal	
BSW Parameter		**BSW Type**
ComGroupSignal		PARAM-CONF-CONTAINER-DEF
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**	
System Template	Reference to a set of signals that must always be kept together.	
M2 Parameter		
CoreCommunication::SystemSignalGroup:containedSignal		
Mapping Rule		**Mapping Type**
create container for each System Signal that is contained in a SystemSignalGroup		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal	
BSW Parameter		**BSW Type**
ComBitPosition		INTEGER-PARAM-DEF
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**	
System Template	This parameter is necessary to describe the bitposition of a signal within an IPdu. 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). Bits within the IPdu are counted as follows (see the OSEK COM v3.0.3 specification) : Bit 0 corresponds to Byte 0 Bit 0 Bit 1 corresponds to Byte 0 Bit 1 ..... Bit 8 corresponds to Byte 1 Bit 0 etc. 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. Note also that the absolute position of the signal in the IPdu is then determined by the definition of the packingByteOrder attribute of the signal.	
M2 Parameter		
CoreCommunication::ISignalToIPduMapping.startPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal	
BSW Parameter		**BSW Type**
ComBitSize		INTEGER-PARAM-DEF
BSW Description		
Size in bits.		
M2 Template	**M2 Description**	
System Template	Size of the signal in bits.	
M2 Parameter		
CoreCommunication::SystemSignal.length		
Mapping Rule		**Mapping Type**
1:1 mapping		full

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComConfig/ComSignalGroup/ComGroupSignal</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComHandleId</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComConfig/ComSignalGroup/ComGroupSignal</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComSignalDataInvalidValue</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3">A constant of a primitive datatype.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">DataTypes:PrimitiveType:SwDataDefProps.invalidValue</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>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">INTEGER-PARAM-DEF</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 arrays last byte. The second-least significant byte shall be assigned to the byte arrays last but one byte, and so on.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">Communcation::UnqueuedReceiverComSpec.initValue OR Communcation::UnqueuedSenderComSpec.initValue</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Depending on Rx/Tx, use one of the two ComSpecs above.</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">ComSignalLength</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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="2">M2 Description</th></tr>
<tr><td>Sw Component Template</td><td colspan="2">The number of bits that are used to make up the opaque type.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">DataType.OpaqueType.numberOfBits</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/ComSignalGroup/ComGroupSignal</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComSignalEndianess</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Defines the endianness of the signal's network representation.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">This parameter defines the order of the bytes of the signal and the packing into the IPdu. 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 IPdu (see the startPosition attribute description).</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">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="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">ComTransferProperty</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Optionally defines whether this group signal shall contribute to the TRIGGERED_ON_CHANGE transfer property of the signal group.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">Also for ISignals which refer to GroupSignals of a SystemSignalGroup the TransferProperty attribute is relevant and shall be evaluated.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">CoreCommunication.ISignalToIPduMapping.transferProperty</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/ComSignalGroup/ComGroupSignal</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComSignalType</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="2">Abstract base class for user defined (and AUTOSAR predefined) datatypes.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">SWC Template::DataTypes</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">Mapping of AUTOSAR data types (defined in the software component template) to COMSignalTypes</td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComSignalGroup/ComGroupSignal</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComInvalidNotification</td><td>FUNCTION-NAME-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.</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>

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal	
BSW Parameter		**BSW Type**
SystemTemplateSystemSignalRef		FOREIGN-REFERENCE-PARAM-DEF
BSW Description		
Reference to the SystemSignalToPduMapping 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**
		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilter		PARAM-CONF-CONTAINER-DEF
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**	
SW Component Template	Base class for data filters.	
M2 Parameter		
ReceiverComSpec.filter		
Mapping Rule		**Mapping Type**
		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterMask		INTEGER-PARAM-DEF
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**	
SW Component Template	Base class for data filters.	
M2 Parameter		
ReceiverComSpec.filter		
Mapping Rule		**Mapping Type**
If ReceiverComSpec.filter has a mask attribute (MaskedNewEqualsX, MaskedNewDiffersX, MaskedNewEqualsMaskedOld, MaskedNewDiffersMaskedOld) then create this element using the mask value.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterMax		INTEGER-PARAM-DEF
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**	
SW Component Template	Base class for data filters.	
M2 Parameter		
ReceiverComSpec.filter		
Mapping Rule		**Mapping Type**
If ReceiverComSpec.filter has a max attribute (NewIsWithing, NewIsOutside) then create this element using the max value.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterMin		INTEGER-PARAM-DEF
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**	
SW Component Template	Base class for data filters.	
M2 Parameter		
ReceiverComSpec.filter		
Mapping Rule		**Mapping Type**
If ReceiverComSpec.filter has a min attribute (NewIsWithing, NewIsOutside) then create this element using the min value.		full

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComFilterOffset</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant. Range = 0..(ComFilterPeriodFactor-1)</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3">OnEveryN.offset</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">specifies the initial number of messages to occur before the first message is passed</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">1:1 mapping</td><td></td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComFilterPeriodFactor</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 are significant.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3">specifies number of messages to occur before the message is passed again</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">OnEveryN.period</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>Com</td><td colspan="3">Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComFilterX</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Only the least significant 32 bits are significant.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SW Component Template</td><td colspan="3">Base class for data filters</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">ReceiverComSpec.filter</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">If ReceiverComSpec.filter has an x attribute (MaskedNewEqualsX, MaskedNewDiffersX) then create this element using the x value.</td><td>full</td></tr>
</table>

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter		
BSW Parameter		**BSW Type**	
ComFilterAlgorithm		ENUMERATION-PARAM-DEF	
BSW Description			
The range of values is specified in the [17] specification, chapter 2.2.2, Reception Filtering.			
M2 Template	**M2 Description**		
SW Component Template	Base class for data filters		
M2 Parameter			
ReceiverComSpec.filter			
Mapping Rule			**Mapping Type**
Type of DataFilter, i.e. Always, Never, MakedNewEqualsX, ...			full

BSW Module	**BSW Context**		
Com	Com/ComGeneral		
BSW Parameter		**BSW Type**	
ComGeneral		PARAM-CONF-CONTAINER-DEF	
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**	
ComConfigurationUseDet		BOOLEAN-PARAM-DEF	
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

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComVersionInfoApi</td><td colspan="2">BOOLEAN-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Activate/Deactivate the version information API (Com_GetVersionInfo). True: version information API activated False: version information API deactivated</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComConfigurationTimeBase</td><td colspan="2">FLOAT-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The period between successive calls to the Main Functions (Rx, Tx, Routing) of AUTOSAR COM in seconds.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">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.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">CoreTopology::ECUInstance::COMProcessingPeriod</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>

9.2 Pdu Router Mapping

BSW Module	**BSW Context**
PduR	PduR/PduRGeneral
BSW Parameter	**BSW Type**
PduRGeneral	PARAM-CONF-CONTAINER-DEF
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	BOOLEAN-PARAM-DEF
BSW Description	
Configuration parameter to enable or disable PDU Router support for CAN interface.	
M2 Template	**M2 Description**
System Template	CAN specific attributes
M2 Parameter	
Fibex4Can::CANCluster	
Mapping Rule	**Mapping Type**
If the regarded ECU is connected to a CANCluster than set this parameter to true.	full

BSW Module	**BSW Context**
PduR	PduR/PduRGeneral
BSW Parameter	**BSW Type**
PduRCanTpSupport	BOOLEAN-PARAM-DEF
BSW Description	
Configuration parameter to enable or disable PDU Router support for CAN TP.	
M2 Template	**M2 Description**
System Template	This is a PDU of the Transport Layer. The main purpose of the TP Layer is to segment and reassemble I-PDUs.
M2 Parameter	
IPduToFrameMapping.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		BOOLEAN-PARAM-DEF
BSW Description		
Configuration parameter to enable or disable PDU Router support for COM.		
M2 Template	**M2 Description**	
System Template	Represents the I-PDU's handled by Com. The IPdu assembled and disassembled in AUTOSAR COM consists of one or more signals. In case no multiplexing is performed this IPdu is routed to/from the Interface Layer.	
M2 Parameter		
CoreCommunication::IPdu::SignalPdu		
Mapping Rule		**Mapping Type**
If a Com I-Pdu exists 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		BOOLEAN-PARAM-DEF
BSW Description		
Configuration parameter to enable or disable PDU Router support for DCM.		
M2 Template	**M2 Description**	
System Template	To distinguish pdus from certain services	
M2 Parameter		
CoreCommunication::IPdu.PduType		
Mapping Rule		**Mapping Type**
if a Dcm I-Pdu exists 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		BOOLEAN-PARAM-DEF
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	BOOLEAN-PARAM-DEF
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**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRGeneral
BSW Parameter	**BSW Type**
PduRFrIfSupport	BOOLEAN-PARAM-DEF
BSW Description	
Configuration parameter to enable or disable PDU Router support for FlexRay interface.	
M2 Template	**M2 Description**
System Template	This could be derived from information in the Sys-T
M2 Parameter	
Fibex4FlexRay::FlexrayCluster	
Mapping Rule	**Mapping Type**
If the regarded ECU is connected to a FlexRayCluster than set this parameter to true	full

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

<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">BOOLEAN-PARAM-DEF</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>
<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">PduRIPduMSupport</td><td colspan="2">BOOLEAN-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Configuration parameter to enable or disable PDU Router support for IPDUM; 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>System Template</td><td colspan="3">Multiplexed PDU (i.e. NOT a COM I-PDU) aggregates one or several SignalIPdus. 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 a 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.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">CoreCommunication::MultiplexedPdu</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">If an IPdu (transmitted or received from ECU under consideration) contains a mulitplexer than set this parameter to true</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">PduRLinIfSupport</td><td colspan="2">BOOLEAN-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Configuration parameter to enable or disable PDU Router support for LIN interface.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">Fibex4Lin::LinCluster</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">If the regarded ECU is connected to a LinCluster than set this parameter to true.</td><td>full</td></tr>
</table>

BSW Module	BSW Context
PduR	PduR/PduRGeneral
BSW Parameter	**BSW Type**
PduRLinTpSupport	BOOLEAN-PARAM-DEF
BSW Description	
Configuration parameter to enable or disable PDU Router support for LIN TP.	
M2 Template	**M2 Description**
System Template	This is a PDU of the Transport Layer. The main purpose of the TP Layer is to segment and reassemble I-PDUs.
M2 Parameter	
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**
PduRMulticastFromIfSupport	BOOLEAN-PARAM-DEF
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**
System Template	This could be derived from information in the Sys-T
M2 Parameter	
CoreCommunication::PduToFrameMapping	
Mapping Rule	**Mapping Type**
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. If several PduTriggerings exist for an IPdu and the PduTriggerings are connected to the same Lin CommConnectorPort, than set this parameter to true.	full

BSW Module	BSW Context
PduR	PduR/PduRGeneral
BSW Parameter	**BSW Type**
PduRMulticastFromTpSupport	BOOLEAN-PARAM-DEF
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**
System Template	
M2 Parameter	
CoreCommunication::PduInstance::NPdu	
Mapping Rule	**Mapping Type**
not supported by the System Template in Release 3.0	local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRMulticastToIfSupport		BOOLEAN-PARAM-DEF
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**	
System Template	This could be derived from information in the Sys-T	
M2 Parameter		
CoreCommunication::PduToFrameMapping		
Mapping Rule		**Mapping Type**
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. If several PduTriggerings exist for an IPdu and the PduTriggerings are connected to the same CommConnector-Port, than set this parameter to true.		full

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRMulticastToTpSupport		BOOLEAN-PARAM-DEF
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**	
System Template		
M2 Parameter		
CoreCommunication::PduInstance::NPdu		
Mapping Rule		**Mapping Type**
not suported by the System Template in Release 3.0		local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRSbTxBufferSupport		BOOLEAN-PARAM-DEF
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**	
PduRVersionInfoApi		BOOLEAN-PARAM-DEF	
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		BOOLEAN-PARAM-DEF	
BSW Description			
If all conditions stated in PDUR165 are fulfilled, 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/PduRGeneral		
BSW Parameter		**BSW Type**	
PduRMemorySize		INTEGER-PARAM-DEF	
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**
PduRMinimumRoutingLoRxPduId		INTEGER-PARAM-DEF
BSW 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**	
System Template		
M2 Parameter		
CoreCommunication::IPdu		
Mapping Rule		**Mapping Type**
Informations about the Pdu can be derived from the System Template. The Pdu Identifier will be set in the ECUC.		local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRMinimumRoutingLoTxPduId		INTEGER-PARAM-DEF
BSW 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**	
System Template		
M2 Parameter		
CoreCommunication::IPdu		
Mapping Rule		**Mapping Type**
Informations about the Pdu can be derived from the System Template. The Pdu Identifier will be set in the ECUC.		local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRMinimumRoutingUpRxPduId		INTEGER-PARAM-DEF
BSW 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**	
System Template		
M2 Parameter		
CoreCommunication::IPdu		
Mapping Rule		**Mapping Type**
Informations about the Pdu can be derived from the System Template. The Pdu Identifier will be set in the ECUC.		local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRMinimumRoutingUpTxPduId		INTEGER-PARAM-DEF
BSW 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**	
System Template		
M2 Parameter		
CoreCommunication::IPdu		
Mapping Rule		**Mapping Type**
Informations about the Pdu can be derived from the System Template. The Pdu Identifier will be set in the ECUC.		local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRMinimumRoutingLoModule		ENUMERATION-PARAM-DEF
BSW 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**
MinimumRouting not described in SystemTemplate Release 3.0		local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRMinimumRoutingUpModule		ENUMERATION-PARAM-DEF
BSW 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**
MinimumRouting not described in SystemTemplate Release 3.0		local

BSW Module	BSW Context	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRSingleIf		ENUMERATION-PARAM-DEF
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**
Calculable: If PduRCanIfSupport = ON and PduRFrIfSupport = OFF and PduRLinIfSupport = OFF then PduRSingleIf = CanIf else if PduRCanIfSupport = OFF and PduRFrIfSupport = ON and PduRLinIfSupport = OFF then PduRSingleIf = FrIf else if PduRCanIfSupport = OFF and PduRFrIfSupport = OFF and PduRLinIfSupport = ON then PduRSingleIf = LinIf		local

BSW Module	**BSW Context**		
PduR	PduR/PduRGeneral		
BSW Parameter		**BSW Type**	
PduRSingleTp		ENUMERATION-PARAM-DEF	
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**
Calculable: If PduRCanTpSupport = ON and PduRFrTpSupport = OFF and PduRLinTpSupport = OFF then PduRSingleTp = CanTp else if PduRCanTp-Support = OFF and PduRFrTpSupport = ON and PduRLinTpSupport = OFF then PduRSingleTp = FrTp else if PduRCanTpSupport = OFF and PduRFrTp-Support = OFF and PduRLinTpSupport = ON then PduRSingleTp = LinTp			local

BSW Module	**BSW Context**		
PduR	PduR/PduRGlobalConfig		
BSW Parameter		**BSW Type**	
PduRGlobalConfig		PARAM-CONF-CONTAINER-DEF	
BSW Description			
This container contains the global configuration parameter of the PduR. It is a MultipleConfigura-tionContainer, i.e. this container and its sub-containers exit once per configuration set.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
Multiple configuration is not handled by this solution. This must be solved within the scope of implementing support for VARIANT handling, i.e. R4.0			local

<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">PduRConfigurationId</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">unique PDURconfiguration identifier</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">CoreTopology::EcuInstance:pduRConfigurationId</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>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">PduRRoutingTable</td><td colspan="2">PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">PDU Router routing table is a subcontainer of PduR. 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>System Template</td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">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/PduRRoutingPath</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">PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This container is a subcontainer of PduRRoutingTable and specifies the routing path of a PDU.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">Calculable</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">CoreCommunication</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">- For each MultiplatformGateway.pduMapping create one PduRRoutingPath. - For each IPduTriggering create one PduRRoutingPath - For each connection between an IPdu and a NPdu create one PduRRoutingPath</td><td>full</td></tr>
</table>

BSW Module	**BSW Context**
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath
BSW Parameter	**BSW Type**
SduLength	INTEGER-PARAM-DEF
BSW Description	
Length of PDU data (SDU). Only required if a TX buffer is configured.	
M2 Template	**M2 Description**
System Template	The size of the IPDU in bits.
M2 Parameter	
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	INTEGER-PARAM-DEF
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**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRDefaultValue
BSW Parameter	**BSW Type**
PduRDefaultValue	PARAM-CONF-CONTAINER-DEF
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**
System Template	Default Value which will be distributed if no pdu has been received since last sending.
M2 Parameter	
Fibex4Multiplatform::IPduMapping::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/PduRDefaultValueElement	
BSW Parameter		**BSW Type**
PduRDefaultValueElement		PARAM-CONF-CONTAINER-DEF
BSW Description		
Each value element is represented by the element and the position in an array.		
M2 Template	**M2 Description**	
System Template	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		
Fibex4Multiplatform::IPduMapping::PduMappingDefaultValue::DefaultValueElement		
Mapping Rule		**Mapping Type**
Container must be created for each DefaultValueElement that is aggregated by PduMappingDefaultValue		full

BSW Module	BSW Context	
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRDefault-Value/PduRDefaultValueElement	
BSW Parameter		**BSW Type**
DefaultValueElement		INTEGER-PARAM-DEF
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**	
System Template	The integer value of a freely defined data byte.	
M2 Parameter		
Fibex4Multiplatform::PduMapping::PduMappingDefaultValue::DefaultValueEle-ment.elementByteValue		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRDefault-Value/PduRDefaultValueElement	
BSW Parameter		**BSW Type**
ElementBytePosition		INTEGER-PARAM-DEF
BSW Description		
This parameter specifies the byte position of the element within the default value		
M2 Template	**M2 Description**	
System Template	This attribute specifies the byte position of the element within the default value	
M2 Parameter		
Fibex4Multiplatform::PduMapping::PduMappingDefaultValue::DefaultValueElement.elementPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRDestPdu	
BSW Parameter		**BSW Type**
PduRDestPdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container is a subcontainer of PduRRoutingPath and specifies one destination for the PDU to be routed.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
Mapping Rule		**Mapping Type**
- For each MultiplatformGateway.IPduMapping create one PduRRoutingPath - For each PduTriggering create one PduRRoutingPath - For each connection between an IPdu and a NPdu create one PduRRoutingPath		full

BSW Module	BSW Context	
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRDestPdu	
BSW Parameter		**BSW Type**
DataProvision		ENUMERATION-PARAM-DEF
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		REFERENCE-PARAM-DEF
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		REFERENCE-PARAM-DEF
BSW Description		
Specifies the assigned transmit buffer. Only required for specific non-TP gateway PDUs.		
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**
PduRSrcPdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container is a subcontainer of PduRRoutingPath and specifies the source of the PDU to be routed.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
Mapping Rule		**Mapping Type**
- For each MultiplatformGateway.IPduMapping create one PduRRoutingPath - For each PduTriggering create one PduRRoutingPath - For each connection between an IPdu and a NPdu create one PduRRoutingPath		full

BSW Module	BSW Context	
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRSrcPdu	
BSW Parameter		**BSW Type**
HandleId		INTEGER-PARAM-DEF
BSW Description		
PDU identifier assigned by PDU Router.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
This parameter is configured by the PduR generator.		local

BSW Module	BSW Context	
PduR	PduR/PduRGlobalConfig/PduRRoutingTable/PduRRoutingPath/PduRSrcPdu	
BSW Parameter		**BSW Type**
SrcPduRef		REFERENCE-PARAM-DEF
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/PduRTpBufferTable	
BSW Parameter		**BSW Type**
PduRTpBufferTable		PARAM-CONF-CONTAINER-DEF
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		INTEGER-PARAM-DEF
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/PduRTpBuffer		
BSW Parameter		**BSW Type**	
PduRTpBuffer		PARAM-CONF-CONTAINER-DEF	
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		INTEGER-PARAM-DEF	
BSW Description			
Length of the buffer.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
PduR	PduR/PduRGlobalConfig/PduRTxBufferTable		
BSW Parameter		**BSW Type**	
PduRTxBufferTable		PARAM-CONF-CONTAINER-DEF	
BSW Description			
This container is a subcontainer of PduR and contains the definition of all transmit buffers (used by specific non-TP PDUs; 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/PduRTxBufferTable		
BSW Parameter		**BSW Type**	
PduRMaxTxBufferNumber		INTEGER-PARAM-DEF	
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/PduRTxBuffer		
BSW Parameter		**BSW Type**	
PduRTxBuffer		PARAM-CONF-CONTAINER-DEF	
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		INTEGER-PARAM-DEF	
BSW Description			
Specifies the depth of the buffer			
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		INTEGER-PARAM-DEF	
BSW Description			
Length of the buffer.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

9.3 IPdu Multiplexer Mapping

BSW Module	**BSW Context**		
IPduMplex	IPduMplex/IPduMConfig		
BSW Parameter		**BSW Type**	
IPduMConfig		PARAM-CONF-CONTAINER-DEF	
BSW Description			
This container contains the sub containers of the IPduMplex module. The IPduMTxPathway sub-container 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**		
System Template	Multiplexed PDU (i.e. NOT a COM I-PDU) aggregates one or several SignalIPdus. 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 a 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.		
M2 Parameter			
CoreCommunication:MultiplexedPdu			
Mapping Rule			**Mapping Type**
Container must be created if System description contains multiplexed IPdus			full

BSW Module	**BSW Context**		
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway		
BSW Parameter		**BSW Type**	
IPduMRxPathway		PARAM-CONF-CONTAINER-DEF	
BSW Description			
Contains the configuration parameters received I-PDUs by the IPduM module.			
M2 Template	**M2 Description**		
System Template			
M2 Parameter			
The composition of the multiplexed IPDU is described by the System Template: CoreCommunication: IPdu:PduMultiplexer			
Mapping Rule			**Mapping Type**
Container must be created if the Frame that contains the multiplexed IPdu is received by the ECU.			full

BSW Module	**BSW Context**		
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication		
BSW Parameter		**BSW Type**	
IPduMRxIndication		PARAM-CONF-CONTAINER-DEF	
BSW Description			
Contains the configuration for incoming RxIndication calls.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication	
BSW Parameter		**BSW Type**
IPduMRxHandleId		INTEGER-PARAM-DEF
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**
Done by ECU Integrator. IPDU IDs are not configured in Sys-T.		local

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication	
BSW Parameter		**BSW Type**
IPduMRxIndicationPduRef		REFERENCE-PARAM-DEF
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**
Done by ECU Integrator.		local

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMBitField		PARAM-CONF-CONTAINER-DEF
BSW Description		
This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMEndBit		INTEGER-PARAM-DEF
BSW Description		
Bit position in an I-PDU of the end of the bit field. Value must fit inside the I-PDU. Value must be the same as or higher than Ipdum_StartBit.		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMStartBit		INTEGER-PARAM-DEF
BSW Description		
Bit position in an I-PDU of the start of the bit field. Value must fit inside the I-PDU. Value must be the same as or lower than Ipdum_EndBit.		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context		
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxDynamicPart		
BSW Parameter		**BSW Type**	
IPduMRxDynamicPart		PARAM-CONF-CONTAINER-DEF	
BSW Description			
This container contains the configuration for the dynamic part of incoming RxIndication calls. When an incoming received I-PDU's selector field matches the IPduM_Selector_Value the I-PDU is unpacked according to the values in the IPduMCopyBitfield and then the new I-PDU constructed and sent out with the I-PDU ID referenced by IPduMOutgoingDyamicPduRef.			
M2 Template	**M2 Description**		
System Template	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			
IPdu:MultiplexedPdu.DynamicPart			
Mapping Rule			**Mapping Type**
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.			full

BSW Module	BSW Context		
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxDynamicPart		
BSW Parameter		**BSW Type**	
IPduMRxSelectorValue		INTEGER-PARAM-DEF	
BSW Description			
This is the selector value that this container refers to.			
M2 Template	**M2 Description**		
System Template	The selector field is part of a multiplexed PDU. It consists of contiguous bits. The value of the selector field selects the layout of the multiplexed part of the PDU. This attribute is only valid for the dynamic part of the PDU.		
M2 Parameter			
Ipdu:MultiplexedIPdu.selectorFieldCode			
Mapping Rule			**Mapping Type**
1:1 mapping			full

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>IPduMplex</td><td colspan="3">IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxDynamicPart</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IPduMOutgoingDynamicPduRef</td><td colspan="2">REFERENCE-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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.</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>IPduMplex</td><td colspan="3">IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxDynamicPart/IPduMCopyBitField</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">PARAM-CONF-CONTAINER-DEF</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>System Template</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">MultiplexedPart.SegmentPosition</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">The bit copy operations in the IPduM can be derived from the Segment description in the System Template</td><td>full</td></tr>
</table>

BSW Module	**BSW Context**		
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxDynamicPart/IPduMCopyBitField		
BSW Parameter		**BSW Type**	
IPduMDestinationBit		INTEGER-PARAM-DEF	
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**		
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.		
M2 Parameter			
MultiplexedPart.SegmentPosition			
Mapping Rule			**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template			full

BSW Module	**BSW Context**		
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxDynamicPart/IPduMCopyBitField/IPduMBitField		
BSW Parameter		**BSW Type**	
IPduMBitField		PARAM-CONF-CONTAINER-DEF	
BSW Description			
This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.			
M2 Template	**M2 Description**		
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.		
M2 Parameter			
MultiplexedPart.SegmentPosition			
Mapping Rule			**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template			full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxDynamicPart/IPduMCopyBitField/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMEndBit		INTEGER-PARAM-DEF
BSW Description		
Bit position in an I-PDU of the end of the bit field. Value must fit inside the I-PDU. Value must be the same as or higher than Ipdum_StartBit.		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxDynamicPart/IPduMCopyBitField/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMStartBit		INTEGER-PARAM-DEF
BSW Description		
Bit position in an I-PDU of the start of the bit field. Value must fit inside the I-PDU. Value must be the same as or lower than Ipdum_EndBit.		
M2 Template	**M2 Description**	
System Template	Segments bit position relatively to the beginning of a multiplexed IPdu.	
M2 Parameter		
MultiplexedPart.segmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxStaticPart	
BSW Parameter		**BSW Type**
IPduMRxStaticPart		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the information on how to unpack the static part of an incoming I-PDU.		
M2 Template	**M2 Description**	
System Template	Some parts/signals of the I-PDU may be the same regardless of the selector field. Such a part is called static part.	
M2 Parameter		
MultiplexedIPdu.StaticPart		
Mapping Rule		**Mapping Type**
create container if static part is described in System Description		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxStaticPart	
BSW Parameter		**BSW Type**
IPduMOutgoingStaticPduRef		REFERENCE-PARAM-DEF
BSW Description		
When the new I-PDU is sent out it is sent with this I-PDU ID. Reference to the sent Pdu representation in the ECU Configuration Description exchange file.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxStaticPart/IPduMCopyBitField	
BSW Parameter		**BSW Type**
IPduMCopyBitField		PARAM-CONF-CONTAINER-DEF
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**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxStaticPart/IPduMCopyBitField	
BSW Parameter		**BSW Type**
IPduMDestinationBit		INTEGER-PARAM-DEF
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**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxStaticPart/IPduMCopyBitField/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMBitField		PARAM-CONF-CONTAINER-DEF
BSW Description		
This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxStaticPart/IPduMCopyBitField/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMEndBit		INTEGER-PARAM-DEF
BSW Description		
Bit position in an I-PDU of the end of the bit field. Value must fit inside the I-PDU. Value must be the same as or higher than Ipdum_StartBit.		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>IPduMplex</td><td colspan="3">IPduMplex/IPduMConfig/IPduMRxPathway/IPduMRxIndication/IPduMRxStaticPart/IPduMCopyBitField/IPduMBitField</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IPduMStartBit</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Bit position in an I-PDU of the start of the bit field. Value must fit inside the I-PDU. Value must be the same as or lower than Ipdum_EndBit.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</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">MultiplexedPart.SegmentPosition</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">The bit copy operations in the IPduM can be derived from the Segment description in the System Template</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>IPduMplex</td><td colspan="3">IPduMplex/IPduMConfig/IPduMTxPathway</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IPduMTxPathway</td><td colspan="2">PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Contains the configuration parameters transmitted I-PDUs by the IPduM module.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">Multiplexed PDU (i.e. NOT a COM I-PDU) aggregates one or several SignalIPdus. 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 a 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.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">MultiplexedIPdu</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Container must be created for each multiplexed Ipdu.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>IPduMplex</td><td colspan="3">IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxConfirmation</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IPduMTxConfirmation</td><td colspan="2">PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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.</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>IPduMplex</td><td colspan="3">IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxConfirmation</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IPduMStaticTxConfirmationIPduRef</td><td colspan="2">REFERENCE-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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.</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>IPduMplex</td><td colspan="3">IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxConfirmation/IPduMDynamicTxConfirmation</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IPduMDynamicTxConfirmation</td><td colspan="2">PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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.</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">done by ECU Integrator</td><td>local</td></tr>
</table>

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxConfirmation/IPduMDynamicTxConfirmation	
BSW Parameter		**BSW Type**
IPduMSelectorValue		INTEGER-PARAM-DEF
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
done by ECU Integrator.		local

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxConfirmation/IPduMDynamicTxConfirmation	
BSW Parameter		**BSW Type**
IPduMDynamicTxConfirmIPduRef		REFERENCE-PARAM-DEF
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	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest	
BSW Parameter		**BSW Type**
IPduMTxRequest		PARAM-CONF-CONTAINER-DEF
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**	
System Template	Multiplexed PDU (i.e. NOT a COM I-PDU) aggregates one or several SignalIPdus. 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 a 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.	
M2 Parameter		
MultiplexedIPdu		
Mapping Rule		**Mapping Type**
Create container for each transmitted multiplexed Ipdu		full

BSW Module	**BSW Context**		
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest		
BSW Parameter		**BSW Type**	
IPduMIPduUnusedAreasDefault		INTEGER-PARAM-DEF	
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**		
System Template	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			
IPdu.unusedBitPattern			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest		
BSW Parameter		**BSW Type**	
IPduMInitialSelectorValue		INTEGER-PARAM-DEF	
BSW Description			
This value is used by the initialization function to set the initial value of the selector field.			
M2 Template	**M2 Description**		
System Template	Dynamic part that shall be used to initialize this multiplexed IPdu. Constraint: Only one DynamicPartAlternative in a DynamicPart shall be the initialDynamicPart.		
M2 Parameter			
MultiplexedIPdu.DynamicPart.DynamicPartAlternative.initialDynamicPart			
Mapping Rule			**Mapping Type**
selectorFieldCode for DynamicPartAlternative with initialDynamicPart = true			full

BSW Module	**BSW Context**		
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest		
BSW Parameter		**BSW Type**	
IPduMSize		INTEGER-PARAM-DEF	
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**		
System Template	The size of the IPDU in bits.		
M2 Parameter			
IPDU.length			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest		
BSW Parameter		**BSW Type**	
IPduMTxTriggerMode		ENUMERATION-PARAM-DEF	
BSW Description			
Selects whether to send the multiplexed I-PDU immediately or at some later date.			
M2 Template	**M2 Description**		
System Template	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			
MultiplexedPdu.triggerMode			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest		
BSW Parameter		**BSW Type**	
IPduMTxConfirmationTimeout		FLOAT-PARAM-DEF	
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		
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest		
BSW Parameter		**BSW Type**	
IPduMOutgoingPduRef		REFERENCE-PARAM-DEF	
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**	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMBitField		PARAM-CONF-CONTAINER-DEF
BSW Description		
This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	**BSW Context**	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMEndBit		INTEGER-PARAM-DEF
BSW Description		
Bit position in an I-PDU of the end of the bit field. Value must fit inside the I-PDU. Value must be the same as or higher than Ipdum_StartBit.		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	**BSW Context**	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMStartBit		INTEGER-PARAM-DEF
BSW Description		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	**BSW Context**		
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxDynamicPart		
BSW Parameter		**BSW Type**	
IPduMTxDynamicPart		PARAM-CONF-CONTAINER-DEF	
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**		
System Template	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			
MultiplexedIPdu.DynamicPart			
Mapping Rule			**Mapping Type**
The outgoing I-PDUs are described in the System Template. Container must be created for each outgoing multiplexed I-PDU.			full

BSW Module	**BSW Context**		
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxDynamicPart		
BSW Parameter		**BSW Type**	
IPduMTxDynamicHandleId		INTEGER-PARAM-DEF	
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**		
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxDynamicPart		
BSW Parameter		**BSW Type**	
IPduMTxDynamicPduRef		REFERENCE-PARAM-DEF	
BSW Description			
Reference to the Pdu representation in the ECU Configuration Description exchange file to be transmitted.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>IPduMplex</td><td colspan="3">IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxDynamicPart/IPduMCopyBitField</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">PARAM-CONF-CONTAINER-DEF</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>System Template</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">MultiplexedPart.SegmentPosition</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">The bit copy operations in the IPduM can be derived from the Segment description in the System Template</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>IPduMplex</td><td colspan="3">IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxDynamicPart/IPduMCopyBitField</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IPduMDestinationBit</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</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">MultiplexedPart.SegmentPosition</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">The bit copy operations in the IPduM can be derived from the Segment description in the System Template</td><td>full</td></tr>
</table>

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxDynamicPart/IPduMCopyBitField/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMBitField		PARAM-CONF-CONTAINER-DEF
BSW Description		
This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxDynamicPart/IPduMCopyBitField/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMEndBit		INTEGER-PARAM-DEF
BSW Description		
Bit position in an I-PDU of the end of the bit field. Value must fit inside the I-PDU. Value must be the same as or higher than Ipdum_StartBit.		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxDynamicPart/IPduMCopyBitField/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMStartBit		INTEGER-PARAM-DEF
BSW Description		
Bit position in an I-PDU of the start of the bit field. Value must fit inside the I-PDU. Value must be the same as or lower than Ipdum_EndBit.		
M2 Template	**M2 Description**	
SystemTemplate	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxStaticPart	
BSW Parameter		**BSW Type**
IPduMTxStaticPart		PARAM-CONF-CONTAINER-DEF
BSW Description		
Configuration parameters for an instance of a Tx_Request call into the IPduM. When a Tx Request with the IPduMTxStaticHandleId is received by the IPduM, 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.		
M2 Template	**M2 Description**	
System Template	Some parts/signals of the I-PDU may be the same regardless of the selector field. Such a part is called static part.	
M2 Parameter		
MultiplexedIPdu.StaticPart		
Mapping Rule		**Mapping Type**
The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxStaticPart	
BSW Parameter		**BSW Type**
IPduMTxStaticHandleId		INTEGER-PARAM-DEF
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**
Done by ECU Integrator.		local

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxStaticPart	
BSW Parameter		**BSW Type**
IPduMTxStaticPduRef		REFERENCE-PARAM-DEF
BSW Description		
Reference to the Pdu representation in the ECU Configuration Description exchange file to be transmitted.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxStaticPart/IPduMCopyBitField	
BSW Parameter		**BSW Type**
IPduMCopyBitField		PARAM-CONF-CONTAINER-DEF
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**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxStaticPart/IPduMCopyBitField	
BSW Parameter		**BSW Type**
IPduMDestinationBit		INTEGER-PARAM-DEF
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**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxStaticPart/IPduMCopyBitField/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMBitField		PARAM-CONF-CONTAINER-DEF
BSW Description		
This is used to specify a contiguous range of bits within an I-PDU. The range is inclusive.		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxStaticPart/IPduMCopyBitField/IPduMBitField	
BSW Parameter		**BSW Type**
IPduMEndBit		INTEGER-PARAM-DEF
BSW Description		
Bit position in an I-PDU of the end of the bit field. Value must fit inside the I-PDU. Value must be the same as or higher than Ipdum_StartBit.		
M2 Template	**M2 Description**	
System Template	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
MultiplexedPart.SegmentPosition		
Mapping Rule		**Mapping Type**
The bit copy operations in the IPduM can be derived from the Segment description in the System Template		full

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>IPduMplex</td><td colspan="2">IPduMplex/IPduMConfig/IPduMTxPathway/IPduMTxRequest/IPduMTxStaticPart/IPduMCopyBitField/IPduMBitField</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">IPduMStartBit</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Bit position in an I-PDU of the start of the bit field. Value must fit inside the I-PDU. Value must be the same as or lower than Ipdum_EndBit.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">MultiplexedPart.SegmentPosition</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">The bit copy operations in the IPduM can be derived from the Segment description in the System Template</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>IPduMplex</td><td colspan="2">IPduMplex/IPduMGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">IPduMGeneral</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Contains the general configuration parameters of IPduMplex.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">Multiplexed PDU (i.e. NOT a COM I-PDU) aggregates one or several SignalIPdus. 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 a 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.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">MultiplexedIPdu</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">Container must be created for each multiplexed IPdu in the System Template</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>IPduMplex</td><td colspan="2">IPduMplex/IPduMGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">IPduMDevErrorDetect</td><td>BOOLEAN-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Active/Deactivate the detection of development errors, for production code this parameter has to be False. True: error detection activated False: error detection deactivated</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>

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMGeneral	
BSW Parameter		**BSW Type**
IPduMStaticPartExists		BOOLEAN-PARAM-DEF
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	
IPduMplex	IPduMplex/IPduMGeneral	
BSW Parameter		**BSW Type**
IPduMVersionInfoApi		BOOLEAN-PARAM-DEF
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

BSW Module	BSW Context	
IPduMplex	IPduMplex/IPduMGeneral	
BSW Parameter		**BSW Type**
IPduMConfigurationTimeBase		FLOAT-PARAM-DEF
BSW Description		
The period between successive ticks of AUTOSAR COM in seconds.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

9.4 FlexRay Interface Mapping

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="2">FrIf/FrIfConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIfConfig</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Configuration of the FlexRay Interface. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">The CommunicationCluster is the main element to describe the topological connection of communicating ECUs.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">container must be created if the ECU is connected to a FlexRay Cluster</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">FrIfCluster</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">The CommunicationCluster is the main element to describe the topological connection of communicating ECUs.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">container must be created if the ECU is connected to a FlexRay Cluster</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">FrIfClstIdx</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.</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>FrIf</td><td colspan="3">FrIf/FrIfConfig/FrIfCluster</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrIfMaxIsrDelay</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The maximum delay in macroticks the FrIf_JoblistExec_<cluster>() function is processed after the absolute timer interrupt was triggered.</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">This has to be configured by the developer (it depends on theCPU, clock-speed, OS implementation, etc).</td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="3">FrIf/FrIfConfig/FrIfCluster</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">GAssumedPreciscion</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Assumed precision of the application network</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>FrIf</td><td colspan="3">FrIf/FrIfConfig/FrIfCluster</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">GClusterDriftDamping</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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].</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">The cluster drift damping factor used in clock synchronization rate correction in microticks</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCommunicationController:clusterDriftDamping</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>

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GColdStartAttempts		INTEGER-PARAM-DEF	
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			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:coldStartAttempts			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GListenNoise		INTEGER-PARAM-DEF	
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**		
System Template	Upper limit for the start up and wake up listen timeout in the presence of noise. Expressed as a multiple of the cluster constant pdListenTimeout. Unit microticks		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:listenNoise			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GMacroPerCycle		INTEGER-PARAM-DEF	
BSW Description			
Number of macroticks in a communication cycle.			
M2 Template	**M2 Description**		
System Template	The number of macroticks in a communication cycle		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:macroPerCycle			
Mapping Rule			**Mapping Type**
1:1 mapping			full

<table>
<tr><td>BSW Module</td><td colspan="3">BSW Context</td></tr>
<tr><td>FrIf</td><td colspan="3">FrIf/FrIfConfig/FrIfCluster</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">GMaxWithoutClockCorrectFatal</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">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].</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td>System Template</td><td colspan="3">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.</td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4">SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:maxWithoutClockCorrectionFatal</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>FrIf</td><td colspan="3">FrIf/FrIfConfig/FrIfCluster</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">GMaxWithoutClockCorrectPassive</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">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]</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td>System Template</td><td colspan="3">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.</td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4">SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:maxWithoutClockCorrectionPassive</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>FrIf</td><td colspan="3">FrIf/FrIfConfig/FrIfCluster</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">GNetworkManagementVectLength</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">Length of the Network Management vector in a cluster [bytes]</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td>System Template</td><td colspan="3">Length of the Network Management vector on a cluster. Unit: Bytes</td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4">SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:networkManagementVectorLength</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>

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GNumberOfMinislots		INTEGER-PARAM-DEF	
BSW Description			
Number of minislots in the dynamic segment			
M2 Template	**M2 Description**		
System Template	number of Minislots in the dynamic segment.		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:numberOfMinislots			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GNumberOfStaticSlots		INTEGER-PARAM-DEF	
BSW Description			
Number of static slots in the static segment			
M2 Template	**M2 Description**		
System Template	The number of static slots in the static segment.		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:numberOfStaticSlots			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GOffsetCorrectionStart		INTEGER-PARAM-DEF	
BSW Description			
Start of the offset correction phase within the NIT, expressed as the number of macroticks from the start of cycle.			
M2 Template	**M2 Description**		
System Template	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			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:offsetCorrectionStart			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GPayloadLengthStatic		INTEGER-PARAM-DEF	
BSW Description			
Payload length of a static frame [16 bit words]			
M2 Template	**M2 Description**		
System Template	Globally configured payload length of a static frame. Unit: 16-bit WORDS.		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:payloadLengthStatic			
Mapping Rule			**Mapping Type**
1:1 Mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GSyncNodeMax		INTEGER-PARAM-DEF	
BSW Description			
Maximum number of nodes that may send frames with the sync frame indicator bit set to one.			
M2 Template	**M2 Description**		
System Template	The maximum number of sync nodes allowed in the cluster		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:syncNodeMax			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GdActionPointOffset		INTEGER-PARAM-DEF	
BSW Description			
Number of Macroticks the action point is offset from the beginning of a Static Slots or symbol window.			
M2 Template	**M2 Description**		
System Template	The offset of the action point in networks		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:actionPointOffset			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdCasRxLowMax		INTEGER-PARAM-DEF
BSW Description		
Upper limit of the CAS acceptance window [gdBit]		
M2 Template	**M2 Description**	
System Template	Upper limit of the Collision Avoidance Symbol (CAS) acceptance window. Unit:bitDuration	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:casRxLowMax		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdDynamicSlotIdlePhase		INTEGER-PARAM-DEF
BSW Description		
Duration of the idle phase within a dynamic slot [Minislots].		
M2 Template	**M2 Description**	
System Template	The duration of the dynamic slot idle phase in minislots.	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:dynamicSlotIdlePhase		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdMiniSlotActionPointOffset		INTEGER-PARAM-DEF
BSW Description		
Number of Macroticks the Minislot action point is offset from the beginning of a Minislot [Macroticks].		
M2 Template	**M2 Description**	
System Template	The Offset of the action point within a minislot. Unit: macroticks	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:minislotActionPointOffset		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GdMinislot		INTEGER-PARAM-DEF	
BSW Description			
Duration of a minislot [Macroticks]			
M2 Template	**M2 Description**		
System Template	The duration of a minislot (dynamic segment). Unit: macroticks.		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:minislotDuration			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GdNit		INTEGER-PARAM-DEF	
BSW Description			
Duration of the Network Idle Time [Macroticks]			
M2 Template	**M2 Description**		
System Template	The duration of the network idle time in macroticks		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:networkIdleTime			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GdStaticSlot		INTEGER-PARAM-DEF	
BSW Description			
Duration of a Static Slot [Macroticks].			
M2 Template	**M2 Description**		
System Template	The duration of a slot in the static segment. Unit: macroticks		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:staticSlotDuration			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdSymbolWindow		INTEGER-PARAM-DEF
BSW Description		
Duration of the symbol window [Macroticks].		
M2 Template	**M2 Description**	
System Template	The duration of the symbol window. Unit: macroticks	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:symbolWindow		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdTssTransmitter		INTEGER-PARAM-DEF
BSW Description		
Number of bits in the Transmission Start Sequence [gdBits].		
M2 Template	**M2 Description**	
System Template	Number of bits in the Transmission Start Sequence [gdBits].	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:transmissionStartSequenceDuration		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdWakeupSymbolRxIdle		INTEGER-PARAM-DEF
BSW Description		
Number of bits used by the node to test the duration of the 'idle' portion of a received wakeup symbol. Duration is equal to (gdWakeupSymbolTxIdle - gdWakeupSymbolTxLow)/2 minus a safe part. (Collisions, clock differences, and other effects can deform the Tx-wakeup pattern.) [gdBit].		
M2 Template	**M2 Description**	
System Template	Number of bits used by the node to test the duration of the idle portion of a received wake up symbol. Unit:bitDuration	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:wakeUpSymbolRxIdle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdWakeupSymbolRxLow		INTEGER-PARAM-DEF
BSW Description		
Number of bits used by the node to test the LOW portion of a received wakeup symbol. This lower limit of zero bits has to be received to detect the LOW portion by the receiver. The duration is equal to gdWakeupSymbolTxLow minus a safe part. (Active stars, clock differences, and other effects can deform the Tx-wakeup pattern.) [gdBits].		
M2 Template	**M2 Description**	
System Template	Number of bits used by the node to test the LOW portion of a received wake up symbol. Unit:bitDuration	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:wakeUpSymbolRxLow		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdWakeupSymbolRxWindow		INTEGER-PARAM-DEF
BSW Description		
The size of the window used to detect wakeups. Detection of a wakeup requires a low and idle period (from one WUS) and a low period (from another WUS) to be detected entirely within a window of this size. The duration is equal to gdWakeupSymbolTxIdle + 2 * gdWakeupSymbolTxLow plus a safe part. (Clock differences and other effects can deform the Tx-wakeup pattern.) [gdBit].		
M2 Template	**M2 Description**	
System Template	Number of bits used by a node to test the overall duration of a received wake up symbol. Unit: gdBit	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:wakeUpSymbolRxWindow		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdWakeupSymbolTxIdle		INTEGER-PARAM-DEF
BSW Description		
Number of bits used by the node to transmit the 'idle' part of a wakeup symbol. The duration is equal to cdWakeupSymbolTxIdle [gdBit].		
M2 Template	**M2 Description**	
System Template	Number of bits used by the node to transmit the idle part of a wake up symbol. Unit: gDbit	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:wakeUpSymbolTxIdle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GdWakeupSymbolTxLow		INTEGER-PARAM-DEF	
BSW Description			
Number of bits used by the node to transmit the LOW part of a wakeup symbol. The duration is equal to cdWakeupSymbolTxLow [gdBit].			
M2 Template	**M2 Description**		
System Template	Number of bits used by the node to transmit the idle part of a wake up symbol. Unit: gDbit		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:wakeUpSymbolTxLow			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GChannels		ENUMERATION-PARAM-DEF	
BSW Description			
The channels that are used by the cluster. ImplementationType: Fr_ChannelType			
M2 Template	**M2 Description**		
SystemTemplate	A physical channel is the transmission medium that is used to send and receive information between two communicating ECUs. Each CommunicationCluster has at least one physical channel. 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.		
M2 Parameter			
SystemTemplate:Fibex:FibexCore:CoreTopology:PhysicalChannel			
Mapping Rule			**Mapping Type**
The channels that are used by the cluster are described in the System Template by the CommunicationCluster-PhysicalChannel relationship			full

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GDMaxMicrotick		ENUMERATION-PARAM-DEF	
BSW Description			
Maximum Microtick length of all Microticks configured within a Cluster.			
M2 Template	**M2 Description**		
System Template	Duration of a microtick. Unit: seconds		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCommunicationController.microtickDuration			
Mapping Rule			**Mapping Type**
maximum of all FlexRayCommunicationController.microtickDuration within a Cluster			full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster
BSW Parameter	**BSW Type**
GdSampleClockPeriod	ENUMERATION-PARAM-DEF
BSW Description	
Sample clock period	
M2 Template	**M2 Description**
System Template	Sample clock period. Unit: seconds
M2 Parameter	
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:sampleClockPeriod	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster
BSW Parameter	**BSW Type**
FrIfMainFunctionCycle	FLOAT-PARAM-DEF
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**
This has to be set by the developer.	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster
BSW Parameter	**BSW Type**
GOffsetCorrectionMax	FLOAT-PARAM-DEF
BSW Description	
describes the maximum value which the offset correction should assume in seconds.	
M2 Template	**M2 Description**
System Template	Cluster global magnitude of the maximum permissible offset correction value Unit:seconds (gOffsetCorrectionMax)
M2 Parameter	
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:OffsetCorrectionMax	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GdBit		FLOAT-PARAM-DEF	
BSW Description			
Nominal bit time in seconds			
M2 Template	**M2 Description**		
System Template	Nominal bit time (= 1 / fx:SPEED). gdBit = cSamplesPerBit * gdSampleClock-Period. Unit: seconds		
M2 Parameter			
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:bit			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GdBitMax		FLOAT-PARAM-DEF	
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**
This has to be set by the developer.			local

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
GdBitMin		FLOAT-PARAM-DEF	
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**
This has to be set by the developer.			local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdCycle		FLOAT-PARAM-DEF
BSW Description		
Length of the cycle, expressed in s		
M2 Template	**M2 Description**	
System Template	Length of the cycle. Unit: seconds	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:cycle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdMacrotick		FLOAT-PARAM-DEF
BSW Description		
Duration of the cluster wide nominal macrotick, expressed in s		
M2 Template	**M2 Description**	
System Template	The duration of the cluster wide nominal macrotick. Unit: seconds	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:macrotickDuration		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdMaxInitializationError		FLOAT-PARAM-DEF
BSW Description		
Maximum error that a node may have following integration in seconds.		
M2 Template	**M2 Description**	
System Template	The maximum error that a node may have after initialization. Unit: seconds	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:maxInitialisationError		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdMaxPropagationDelay		FLOAT-PARAM-DEF
BSW Description		
Maximum propagation delay of a Cluster (in seconds).		
M2 Template	**M2 Description**	
System Template	Maximum propagation delay of a Cluster (in seconds).	
M2 Parameter		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:maxPropagationDelay		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
GdMinPropagationDelay		FLOAT-PARAM-DEF
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		
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster:minPropagationDelay		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController	
BSW Parameter		**BSW Type**
FrIfController		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the configuration of FlexRay CC.		
M2 Template	**M2 Description**	
System Template	The communication controller is a dedicated hardware device by means of which hosts are sending frames to and receiving frames from the communication medium.	
M2 Parameter		
SystemTemplate:Fibex:FibexCore:Topology:EcuInstance:CommunicationController		
Mapping Rule		**Mapping Type**
container must be created if the ECUInstance contains a FlexRay CC		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController	
BSW Parameter		**BSW Type**
FrIfCtrlIdx		INTEGER-PARAM-DEF
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**
This has to be set by the developer.		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfAbsTimer	
BSW Parameter		**BSW Type**
FrIfAbsTimer		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the configuration of an absolute timer of a FlexRay CC.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
Mapping Rule		**Mapping Type**
container must be created if the ECUInstance contains a FlexRay CC		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfAbsTimer	
BSW Parameter		**BSW Type**
FrIfAbsTimerIdx		INTEGER-PARAM-DEF
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**
This reference has to be set by the developer.		local

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="3">FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrIfFrameTriggering</td><td colspan="2">PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">A Frame triggering contains the communication parameters of the FlexRay Frame as well as a reference to the Frame Construction Plan.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">Data frame which is sent over a communication medium. Each Frame can be identified per channel by an Identifier (ID).</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">SystemTemplate:FibexCore:CoreCommunication:Frame</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">container must be created if a FlexRay Frame is transsmitted</td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="3">FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrIfAlwaysTransmit</td><td colspan="2">BOOLEAN-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The FlexRay Driver API service Fr_TransmitTxLSdu() will be called for this FlexRay Frame even if FrIf_Transmit() has not been called for any of the PDUs in the Frame.</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">This must be set by the developer.</td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="3">FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrIfBaseCycle</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This parameter contains the FlexRay Base Cycle used to transmit this FlexRay Frame.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">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.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">SystemTemplate:Fibex4FlexRay:FlexRayFrameTriggering:AbsolutelyScheduledTiming:BaseCycle</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Find scheduleEntry with reference to this frameTriggering FRIF_BASE_CYCLE = baseCycle of this scheduleEntry</td><td>full</td></tr>
</table>

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfCycleRepetition		INTEGER-PARAM-DEF
BSW Description		
This parameter contains the FlexRay Cycle Repetition used to transmit this FlexRay Frame. possible Values: 1,2,4,8,16,32,64		
M2 Template	**M2 Description**	
System Template	The number of communication cycles (after the frist cycle) whenever the frame described by this timing is sent again.	
M2 Parameter		
SystemTemplate:Fibex4FlexRay:FlexRayFrameTriggering:AbsolutelyScheduledTiming:CycleRepetition		
Mapping Rule		**Mapping Type**
Find scheduleEntry with reference to this frameTriggering FRIF_CYCLE_REPETITION = cycleRepetition of this scheduleEntry		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfSlotId		INTEGER-PARAM-DEF
BSW Description		
This parameter contains the FlexRay Slot ID used to transmit this FlexRay Frame.		
M2 Template	**M2 Description**	
System Template	In the static part the SlotID defines the slot in which the frame is transmitted. The SlotID also determines, in combination with FlexrayCluster::numberOfStaticSlots, whether the frame is sent in static or dynamic segment. In the dynamic part, the slot id is equivalent to a priority. Lower dynamic slot ids are all sent until the end of the dynamic segment. Higher numbers, which were ignored that time, have to wait one cycle and then must try again.	
M2 Parameter		
SystemTemplate:Fibex4FlexRay:FlexRayFrameTriggering:AbsolutelyScheduledTiming:slotId		
Mapping Rule		**Mapping Type**
Find scheduleEntry with reference to this frameTriggering FRIF_SLOT_ID = baseSlot of this scheduleEntry		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfChannel		ENUMERATION-PARAM-DEF
BSW Description		
This parameter contains the FlexRay Channel used to transmit this FlexRay Frame.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
Mapping Rule		**Mapping Type**
Calculable: The receiver ECUs and the transmitter ECUs of each frame are described by references from the CommConnectorPort to FrameTriggering. The CommConnectorPort contains a reference to the PhysicalChannel		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfFrameStructureRef		REFERENCE-PARAM-DEF
BSW Description		
Reference to the Construction Plan of the FlexRay Frame.		
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**
FrIfRelTimer		PARAM-CONF-CONTAINER-DEF
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**
container must be created if the ECUInstance contains a FlexRay CC		local

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="2">FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfRelTimer</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIfRelTimerIdx</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This parameter provides a zero-based consecutive index of the relative timers. Upper layer BSW modules use this index to identify a relative timer.</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">This must be set by the developer.</td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="2">FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfTransceiver</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIfTransceiver</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Up to two FlexRay Transceivers may connect a Controller to a Cluster. This container realizes a Controller-Transceiver assignment.</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">container must be created if the ECUInstance contains a FlexRay CC</td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="2">FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfTransceiver</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIfClusterChannel</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This parameter identifies to which one of the two Channels ”A” or ”B” of the Cluster the Transceiver is connected.</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">container must be created if the ECUInstance contains a FlexRay CC</td><td>local</td></tr>
</table>

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfVirtualBuffer	
BSW Parameter		**BSW Type**
FrIfVirtualBuffer		PARAM-CONF-CONTAINER-DEF
BSW Description		
A virtual buffer is an abstraction of the transmit/receive buffer of a FlexRay CC to be used for communication. The virtual buffer identifier does not give any clue which real buffer is used.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
This has to be configured by the developer		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfVirtualBuffer	
BSW Parameter		**BSW Type**
FrIfVirtualBufferIdx		INTEGER-PARAM-DEF
BSW Description		
This parameter identifies the virtual buffer.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
This has to be configured by the developer		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfVirtualBuffer	
BSW Parameter		**BSW Type**
FrIfVBTriggeringRef		REFERENCE-PARAM-DEF
BSW Description		
Reference to the assigned Frame triggering.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
FrameTriggering		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList
BSW Parameter	**BSW Type**
FrIfJobList	PARAM-CONF-CONTAINER-DEF
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**
this container must be created for each FlexRay cluster	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList/FrIfJob
BSW Parameter	**BSW Type**
FrIfJob	PARAM-CONF-CONTAINER-DEF
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**
FrIfCycle	INTEGER-PARAM-DEF
BSW Description	
The FlexRay Cycle in which the communication operation will execute this job	
M2 Template	**M2 Description**
System Template	
M2 Parameter	
SystemTemplate:Fibex4FlexRay:FlexrayCommunication:AbsolutelyScheduledTiming:CycleRepetition	
Mapping Rule	**Mapping Type**
Find scheduleEntry with reference to this frameTriggering FRIF_CYCLE_REPETITION = cycleRepetition of this scheduleEntry	full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList/FrIfJob		
BSW Parameter		**BSW Type**	
FrIfMacrotick		INTEGER-PARAM-DEF	
BSW Description			
Macrotick offset in the Cycle [Macrotick]			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
This has to be set by the developer			local

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList/FrIfJob/FrIfCommunicationOperation		
BSW Parameter		**BSW Type**	
FrIfCommunicationOperation		PARAM-CONF-CONTAINER-DEF	
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**	
FrIfCommunicationOperationIdx		INTEGER-PARAM-DEF	
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**	
FrIfCommunicationAction		ENUMERATION-PARAM-DEF	
BSW Description			
The action to be performed in the FlexRay Operation			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
This has to be configured by the developer			local

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList/FrIfJob/FrIfCommunicationOperation		
BSW Parameter		**BSW Type**	
FrIfVirtualBufferRef		REFERENCE-PARAM-DEF	
BSW Description			
Reference to a virtual buffer.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfFrameStructure		
BSW Parameter		**BSW Type**	
FrIfFrameStructure		PARAM-CONF-CONTAINER-DEF	
BSW Description			
The Frame structure specifies a Construction Plan how a Frame is assembled with PDUs and their respective Update-Bits.			
M2 Template	**M2 Description**		
System Template	Data frame which is sent over a communication medium.		
M2 Parameter			
SystemTemplate:FibexCore:CoreCommunication:Communication:Frame			
Mapping Rule			**Mapping Type**
construction plan is described in the System Template (Frame, PduToFrameMapping and Pdu element)			full

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfFrameStructure		
BSW Parameter		**BSW Type**	
FrIfLSduLength		INTEGER-PARAM-DEF	
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**		
System Template	The used length (in bytes) of the referencing frame. Should not be confused with a static byte length reserved for each frame by some platforms (e.g. FlexRay).		
M2 Parameter			
SystemTemplate:FibexCore:CoreCommunication:Communication:Frame:frameLength			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfFrameStructure/FrIfPdusInFrame		
BSW Parameter		**BSW Type**	
FrIfPdusInFrame		PARAM-CONF-CONTAINER-DEF	
BSW Description			
This container holds all the information about a PDU in a FlexRay Frame.			
M2 Template	**M2 Description**		
System Template	A frames layout as a sequence of PDU Instances.		
M2 Parameter			
SystemTemplate:FibexCore:CoreCommunication:Communication:Frame:PduToFrameMapping			
Mapping Rule			**Mapping Type**
A container must be created for each PduToFrameMapping inside the frame.			full

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfFrameStructure/FrIfPdusInFrame		
BSW Parameter		**BSW Type**	
FrIfPduOffset		INTEGER-PARAM-DEF	
BSW Description			
The value specifies the offset of the PDU within the Frame [bytes].			
M2 Template	**M2 Description**		
System Template	PDUs position inside of a Frame.		
M2 Parameter			
SystemTemplate:FibexCore:CoreCommunication:Communication:Frame:PduInstance:PduPosition			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfFrameStructure/FrIfPdusInFrame	
BSW Parameter		**BSW Type**
FrIfPduUpdateBitOffset		INTEGER-PARAM-DEF
BSW Description		
This value specifies where the PDU's Update-Bit is stored in the Frame (bit location of PDU's Update-Bit in the FlexRay Frame).		
M2 Template	**M2 Description**	
System Template	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 Parameter		
SystemTemplate:FibexCore:CoreCommunication:Communication:Frame:PduInstance:updateIndicationBitPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfFrameStructure/FrIfPdusInFrame	
BSW Parameter		**BSW Type**
FrIfFrIfPduRef		REFERENCE-PARAM-DEF
BSW Description		
This is the reference to the local definition of a PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfPdu	
BSW Parameter		**BSW Type**
FrIfPdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
Contains PDU information. A PDU may be either a transmission PDU or a reception PDU.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
ystemTemplate:FibexCore:CoreCommunication:Communication:Frame:PduToFrameMapping		
Mapping Rule		**Mapping Type**
The container must be created for each PduToFrameMapping		full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfRxPdu
BSW Parameter	**BSW Type**
FrIfRxPdu	PARAM-CONF-CONTAINER-DEF
BSW Description	
Receive PDU	
M2 Template	**M2 Description**
Frame::PduToFrameMapping	An IPdu or a NPdu (XOR).
M2 Parameter	
Frame::PduToFrameMapping::Pdu	
Mapping Rule	**Mapping Type**
Container must be created if the Pdu is received via the FlexRay Channel (Physical Channel)	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfRxPdu
BSW Parameter	**BSW Type**
FrIfPduRef	REFERENCE-PARAM-DEF
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**
FrIfTxPdu	PARAM-CONF-CONTAINER-DEF
BSW Description	
This container specifies transmission PDUs.	
M2 Template	**M2 Description**
Frame::PduToFrameMapping	An IPdu or a NPdu (XOR).
M2 Parameter	
Frame::PduToFrameMapping::Pdu	
Mapping Rule	**Mapping Type**
Container must be created if the Pdu is transmitted via the FlexRay Channel (Physical Channel)	full

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfConfig/FrIfPdu/FrIfPduDirection/FrIfTxPdu		
BSW Parameter		**BSW Type**	
FrIfConfirm		BOOLEAN-PARAM-DEF	
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**	
FrIfImmediate		BOOLEAN-PARAM-DEF	
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**	
FrIfCounterLimit		INTEGER-PARAM-DEF	
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**
FrIfTxPduId		INTEGER-PARAM-DEF
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**
FrIfPduRef		REFERENCE-PARAM-DEF
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/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfGeneral		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the general configuration parameters of the FlexRay Interface.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
Mapping Rule		**Mapping Type**
Container must be created if ECU is part of a FlexRay Cluster		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfAllowSwitchConfig		BOOLEAN-PARAM-DEF
BSW Description		
Enables/disables the existence of the FrIf_SwitchConfig() API service. In AUTOSAR R2.0 this parameter has to be set to OFF. true: FrIf_SwitchConfig() API service exists false: FrIf_SwitchConfig() API service does not exist		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
This parameter has to be set by the developer.		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfDevErrorDetect		BOOLEAN-PARAM-DEF
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**
This parameter has to be set by the developer.		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfVersionInfoApi		BOOLEAN-PARAM-DEF
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

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfApplMemSizeMax		INTEGER-PARAM-DEF
BSW Description		
The maximum RAM memory size to be used by the FrIf for variable data, i.e. storage of flags, states, and temporary data. If the FrIf is configured at post build time, it is not allowed to use more RAM memory than specified by this parameter.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
This parameter has to be set by the developer.		Local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfNumClstSupported		INTEGER-PARAM-DEF
BSW Description		
Maximum number of FlexRay Clusters that the FlexRay Interface supports.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
This parameter has to be set by the developer.		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfNumCtrlSupported		INTEGER-PARAM-DEF
BSW Description		
Maximum number of FlexRay CCs that the FlexRay Interface supports		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
This parameter has to be set by the developer.		local

9.5 FlexRay Driver Mapping

BSW Module	**BSW Context**		
Fr	Fr/FrGeneral		
BSW Parameter		**BSW Type**	
FrGeneral		PARAM-CONF-CONTAINER-DEF	
BSW Description			
General configuration (parameters) of the FlexRay Driver module.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Fr	Fr/FrGeneral		
BSW Parameter		**BSW Type**	
FrDevErrorDetect		BOOLEAN-PARAM-DEF	
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**	
FrRelativeTimerEnable		BOOLEAN-PARAM-DEF	
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		BOOLEAN-PARAM-DEF
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/FrGeneral	
BSW Parameter		**BSW Type**
FrIndex		INTEGER-PARAM-DEF
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		INTEGER-PARAM-DEF
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/FrMultipleConfiguration		
BSW Parameter		**BSW Type**	
FrMultipleConfiguration		PARAM-CONF-CONTAINER-DEF	
BSW Description			
Configuration of the individual controllers.			
M2 Template	**M2 Description**		
System Template			
M2 Parameter			
Fibex:FibexCore:Topology:EcuInstance:CommunicationController			
Mapping Rule			**Mapping Type**
For each controller in the System Template a FrDriver must be configured			local

BSW Module	**BSW Context**		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
FrController		PARAM-CONF-CONTAINER-DEF	
BSW Description			
Configuration of the individual controller.			
M2 Template	**M2 Description**		
System Template			
M2 Parameter			
¿Fibex:FibexCore:Topology:EcuInstance:CommunicationController			
Mapping Rule			**Mapping Type**
container must be created for each FlexRay controller described in the System Template			local

BSW Module	**BSW Context**		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PAllowHaltDueToClock		BOOLEAN-PARAM-DEF	
BSW Description			
Boolean flag that controls the transition to the POC:halt state due to a clock synchronization errors. If set to true, the CC is allowed to transition to POC:halt. If set to false, the CC will not transition to the POC:halt state but will enter or remain in the POC:normal passive state (self healing would still be possible)			
M2 Template	**M2 Description**		
System Template	Boolean flag that controls the transition to the POC:halt state due to a clock synchronization errors. If set to true, the Communication Controller is allowed to transition to POC:halt. If set to false, the Communication Controller will not transition to the POC:halt state but will enter or remain in the normal POC (passive State).		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:allowHaltDueToClock			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PKeySlotUsedForStartup		BOOLEAN-PARAM-DEF	
BSW Description			
Flag indicating whether the Key Slot is used to transmit a startup frame			
M2 Template	**M2 Description**		
System Template	Flag indicating whether the Key Slot is used to transmit a startup frame.		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:keySlotUsed-ForStartUp			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PKeySlotUsedForSync		BOOLEAN-PARAM-DEF	
BSW Description			
Flag indicating whether the Key Slot is used to transmit a sync frame			
M2 Template	**M2 Description**		
System Template	lag indicating whether the Key Slot is used to transmit a sync frame.		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:keySlotUsed-ForSync			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PSingleSlotEnabled		BOOLEAN-PARAM-DEF	
BSW Description			
Flag indicating whether or not the node shall enter single slot mode following startup			
M2 Template	**M2 Description**		
System Template	Flag indicating whether or not the node shall enter single slot mode following startup.		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:SingleSlotEn-abled			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
FrCtrlClock		INTEGER-PARAM-DEF	
BSW Description			
Determines clock connected to the CC [Hz].			
M2 Template	**M2 Description**		
ECU Resource Template	The clock delivers the time for the PU and other HW Elements on the ECU.		
M2 Parameter			
ECUResourceTemplate:CommunicationPeripheral:Clock			
Mapping Rule			**Mapping Type**
Each FlexRay CommunicationController element in the System Template is mapped to a CommunicationPeripheral element in the ECU Resource Template			full

BSW Module	**BSW Context**		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
FrCtrlIdx		INTEGER-PARAM-DEF	
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**	
PAllowPassiveToActive		INTEGER-PARAM-DEF	
BSW Description			
Number of consecutive even/odd cycle pairs that must have valid clock correction terms before the CC will be allowed to transition from the POC:normal passive state to POC:normal active state. If set to zero, the CC is not allowed to transition from POC:normal passive to POC:normal active			
M2 Template	**M2 Description**		
System Template	Number of consecutive even/odd cycle pairs that must have valid clock correction terms before the Communication Controller will be allowed to transition from the POC:normal passive state to POC:normal active state. If set to 0, the Communication Controller is not allowed to transition from POC:norm		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:allowPassiveToActive			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
PClusterDriftDamping	INTEGER-PARAM-DEF
BSW Description	
Local cluster drift damping factor used for rate correction [Microticks]	
M2 Template	**M2 Description**
System Template	The cluster drift damping factor used in clock synchronization rate correction in microticks
M2 Parameter	
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:clusterDrift-Damping	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
PDecodingCorrection	INTEGER-PARAM-DEF
BSW Description	
Value used by the receiver to calculate the difference between primary time reference point and secondary time reference point [Microticks]	
M2 Template	**M2 Description**
System Template	Value used by the receiver to calculate the difference between primary time reference point and secondary time reference point. Unit: mT (pDecodingCor-rection)
M2 Parameter	
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:decodingCorrec-tion	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
PDelayCompensationA	INTEGER-PARAM-DEF
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 microsec to 0.05 microsec. In practice, the minimum of the propagation delays of all sync nodes should be applied [Microticks].	
M2 Template	**M2 Description**
System Template	Value used to compensate for reception delays on channel A Unit: Microticks
M2 Parameter	
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:delayCompensa-tionA	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PDelayCompensationB		INTEGER-PARAM-DEF	
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 microsec to 0.05 microsec. In practice, the minimum of the propagation delays of all sync nodes should be applied [Microticks].			
M2 Template	**M2 Description**		
System Template	Value used to compensate for reception delays on channel B. Unit: Microticks		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:delayCompensationB			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PExternOffsetCorrection		INTEGER-PARAM-DEF	
BSW Description			
Number of microticks added or subtracted to the NIT to carry out a host-requested external offset correction [Microticks].			
M2 Template	**M2 Description**		
System Template	Fixed amount added or subtracted to the calculated offset correction term to facilitate external offset correction, expressed in node-local microticks.		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:externOffsetCorrection			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PExternRateCorrection		INTEGER-PARAM-DEF	
BSW Description			
Number of microticks added or subtracted to the cycle to carry out a host-requested external rate correction [Microticks].			
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			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:externRateCorrection			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
PKeySlotId		INTEGER-PARAM-DEF
BSW Description		
ID of the slot used to transmit the startup frame, sync frame, or designated single slot frame		
M2 Template	**M2 Description**	
System Template	ID of the slot used to transmit the startup frame, sync frame, or designated single slot frame.	
M2 Parameter		
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:keySlotID		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
PLatestTx		INTEGER-PARAM-DEF
BSW Description		
Number of the last minislot in which a frame transmission can start in the dynamic segment [Minislots].		
M2 Template	**M2 Description**	
System Template	The number of the last minislot in which a transmission can start in the dynamic segment for the respective node	
M2 Parameter		
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:latestTX		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
PMacroInitialOffsetA		INTEGER-PARAM-DEF
BSW Description		
Integer number of macroticks between the static slot boundary and the following macrotick boundary of the secondary time reference point based on the nominal macrotick duration [Macroticks].		
M2 Template	**M2 Description**	
System Template	Integer number of macroticks between the static slot boundary and the closest macrotick boundary of the secondary time reference point based on the nominal macrotick duration. (pMacroInitialOffset)	
M2 Parameter		
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:macroInitialOffsetA		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
PMacroInitialOffsetB		INTEGER-PARAM-DEF
BSW Description		
Integer number of macroticks between the static slot boundary and the following macrotick boundary of the secondary time reference point based on the nominal macrotick duration [Macroticks].		
M2 Template	**M2 Description**	
System Template	Integer number of macroticks between the static slot boundary and the closest macrotick boundary of the secondary time reference point based on the nominal macrotick duration. (pMacroInitialOffset)	
M2 Parameter		
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:macroInitialOffsetB		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
PMicroInitialOffsetA		INTEGER-PARAM-DEF
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**	
System Template	Number of microticks between the closest macrotick boundary described by gMacroInitialOffset and the secondary time reference point. The parameter depends on pDelayCompensationA and therefore it has to be set independently for each channel.	
M2 Parameter		
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:MicroInitialOffsetA		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PMicroInitialOffsetB		INTEGER-PARAM-DEF	
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**		
System Template	Number of microticks between the closest macrotick boundary described by gMacroInitialOffset and the secondary time reference point. The parameter depends on pDelayCompensationB and therefore it has to be set independently for each channel.		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:MicroInitialOffsetB			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PMicroPerCycle		INTEGER-PARAM-DEF	
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].			
M2 Template	**M2 Description**		
System Template	The nominal number of microticks in a communication cycle		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:microPerCycle			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PMicroPerMacroNom		INTEGER-PARAM-DEF	
BSW Description			
Number of microticks per nominal macrotick that all implementations must support [Microticks].			
M2 Template	**M2 Description**		
System Template	Number of microticks per nominal macrotick that all implementations must support.		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:MicroPerMacroNom			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
POffsetCorrectionOut		INTEGER-PARAM-DEF	
BSW Description			
Magnitude of the maximum permissible offset correction value [Microticks].			
M2 Template	**M2 Description**		
System Template	Magnitude of the maximum permissible offset correction value. Unit:microtick (pOffsetCorrectionOut)		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:offsetCorrectionOut			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PPayloadLengthDynMax		INTEGER-PARAM-DEF	
BSW Description			
Maximum payload length for dynamic frames [16 bit words].			
M2 Template	**M2 Description**		
System Template	Maximum payload length for the dynamic channel of a frame in 16 bit WORDS.		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:maximumDynamicPayloadLength			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PRateCorrectionOut		INTEGER-PARAM-DEF	
BSW Description			
Magnitude of the maximum permissible rate correction value [Microticks].			
M2 Template	**M2 Description**		
System Template	Magnitude of the maximum permissible rate correction value. Unit:mT (pRateCorrectionOut)		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:rateCorrectionOut			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
PWakeupPattern		INTEGER-PARAM-DEF
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		
M2 Template	**M2 Description**	
System Template	Number of repetitions of the Tx-wakeup symbol to be sent during the CC_WakeupSend state of this Node in the cluster	
M2 Parameter		
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:wakeUpPattern		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
PdAcceptedStartupRange		INTEGER-PARAM-DEF
BSW Description		
Expanded range of measured clock deviation allowed for startup frames during integration [Microticks].		
M2 Template	**M2 Description**	
System Template	Expanded range of measured clock deviation allowed for startup frames during integration. Unit:microtick	
M2 Parameter		
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:acceptedStartupRange		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
PdListenTimeout		INTEGER-PARAM-DEF
BSW Description		
Upper limit for the start up listen timeout and wake up listen timeout [Microticks].		
M2 Template	**M2 Description**	
System Template	Upper limit for the start up listen timeout and wake up listen timeout.	
M2 Parameter		
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:listenTimeout		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PdMaxDrift		INTEGER-PARAM-DEF	
BSW Description			
Maximum drift offset between two nodes that operate with unsynchronized clocks over one communication cycle [Microticks].			
M2 Template	**M2 Description**		
System Template	Maximum drift offset in microticks between two nodes that operate with unsynchronized clocks over one communication cycle.		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:maxDrift			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PChannels		ENUMERATION-PARAM-DEF	
BSW Description			
Channels to which the node is connected			
M2 Template	**M2 Description**		
System Template	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			
Described by the relation between CommunicationCluster and PhysicalChannel: SystemTemplate:FibexCore:CoreTopology:CommunicationCluster:PhysicalChannel			
Mapping Rule			**Mapping Type**
calculable			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
PSamplesPerMicrotick		ENUMERATION-PARAM-DEF	
BSW Description			
Number of samples per microtick			
M2 Template	**M2 Description**		
System Template	Number of samples per microtick		
M2 Parameter			
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:samplesPerMicrotick			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
PWakeupChannel		ENUMERATION-PARAM-DEF
BSW Description		
Channel used by the node to send a wakeup pattern		
M2 Template	**M2 Description**	
System Template	Referenced channel used by the node to send a wakeup pattern. (pWake-upChannel)	
M2 Parameter		
SystemTemplate:Fibex4FlexRay:FlexRayTopology:FlexRayCommunicationConnector:wake-UpChannel		
Mapping Rule		**Mapping Type**
		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
PdMicrotick		ENUMERATION-PARAM-DEF
BSW Description		
Duration of a microtick.		
M2 Template	**M2 Description**	
System Template	Duration of a microtick. This attribute can be derived from samplePerMicrotick and gdSampleClockPeriod. Unit: seconds	
M2 Parameter		
SystemTemplate:Fibex4Flexray:FlexrayTopology:FlexrayCommunicationController:microtickDura-tion		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrAbsoluteTimer	
BSW Parameter		**BSW Type**
FrAbsoluteTimer		PARAM-CONF-CONTAINER-DEF
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		INTEGER-PARAM-DEF
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/FrRelativeTimer	
BSW Parameter		**BSW Type**
FrRelativeTimer		PARAM-CONF-CONTAINER-DEF
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		INTEGER-PARAM-DEF
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/FrTpGeneral		
BSW Parameter		**BSW Type**	
FrTpGeneral		PARAM-CONF-CONTAINER-DEF	
BSW Description			
This container contains the general configuration (parameters) of the FlexRay TP.			
M2 Template	**M2 Description**		
System Template	This is a PDU of the Transport Layer. The main purpose of the TP Layer is to segment and reassemble I-PDUs.		
M2 Parameter			
CoreCommunication::NPdu			
Mapping Rule			**Mapping Type**
Container must be created if a FlexRay Frame that is received or transmitted by an ECU contains a NPdu.			full

BSW Module	BSW Context		
FrTp	FrTp/FrTpGeneral		
BSW Parameter		**BSW Type**	
FrTpDevErrorDetect		BOOLEAN-PARAM-DEF	
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		BOOLEAN-PARAM-DEF	
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		BOOLEAN-PARAM-DEF
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		BOOLEAN-PARAM-DEF
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		BOOLEAN-PARAM-DEF
BSW Description		
Preprocessor switch for enabling Transmit Cancellation.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**		
FrTp	FrTp/FrTpGeneral		
BSW Parameter		**BSW Type**	
FrTpVersionInfoApi		BOOLEAN-PARAM-DEF	
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/FrTpGeneral		
BSW Parameter		**BSW Type**	
FrTpChanNum		INTEGER-PARAM-DEF	
BSW Description			
Preprocessor switch for defining the number of concurrent channels the module supports. Up to 32 channels shall be definable here.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
FrTp	FrTp/FrTpGeneral		
BSW Parameter		**BSW Type**	
FrTpMainFuncCycle		FLOAT-PARAM-DEF	
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/FrTpMultipleConfig
BSW Parameter	**BSW Type**
FrTpMultipleConfig	PARAM-CONF-CONTAINER-DEF
BSW Description	
This container holds one or several multiple configuration sets.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel
BSW Parameter	**BSW Type**
FrTpChannel	PARAM-CONF-CONTAINER-DEF
BSW Description	
This container contains the configuration (parameters) of one FlexRay TP channel.	
M2 Template	**M2 Description**
System Template	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.
M2 Parameter	
TransportProtocols::FrTpChannel	
Mapping Rule	**Mapping Type**
container must be created if FrTpChannel is defined in the System Template	full

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel
BSW Parameter	**BSW Type**
FrTpGrpSeg	BOOLEAN-PARAM-DEF
BSW Description	
Here can be specified, whether segmentation within a 1:n connection is allowed or not.	
M2 Template	**M2 Description**
System Template	This attribute defines whether segmentation within a 1:n connection is allowed or not.
M2 Parameter	
TransportProtocols::FrTpChannel.multicastSegmentation	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpTc		BOOLEAN-PARAM-DEF
BSW Description		
With this switch Transmit Cancellation can be turned on or off for this channel.		
M2 Template	**M2 Description**	
System Template	This attribute states whether Transmit Cancellation is supported on this channel.	
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**
FrTpUsePduFc		BOOLEAN-PARAM-DEF
BSW 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**	
System Template	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.	
M2 Parameter		
TransportProtocols::FlexRayTpConnection.flowControlPdu		
Mapping Rule		**Mapping Type**
Information can be derived from reference o the Flow Control NPdu.		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpChannelId		INTEGER-PARAM-DEF
BSW Description		
The Id of the channel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpConNum		INTEGER-PARAM-DEF
BSW Description		
This parameter states the number of connections used in this channel. At least 256 shall be configurable here.		
M2 Template	**M2 Description**	
System Template	Group of connections that can be used in this channel.	
M2 Parameter		
TransportProtocols::FlexRayTpChannel.tpConnection		
Mapping Rule		**Mapping Type**
Number of connections that are aggregated by the channel		local

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpMaxAr		INTEGER-PARAM-DEF
BSW Description		
This parameter defines the maximum number of trying to send a frame when a TIMEOUT AR occurs (depending on whether retry is configured).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpMaxAs		INTEGER-PARAM-DEF
BSW Description		
This parameter defines the maximum number of trying t osend a frame when a TIMEOUT AS occurs (depending on whether retry is configured)		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpMaxBs		INTEGER-PARAM-DEF
BSW Description		
This parameter is only relevant when having retry activated. It limits the maximal block size the FrTp can choose in order to limit the amount of Tx buffer that will be requested at the sender side in a segmented transfer.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpMaxBufReq		INTEGER-PARAM-DEF
BSW 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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel		
BSW Parameter		**BSW Type**	
FrTpMaxFrIf		INTEGER-PARAM-DEF	
BSW Description			
This parameter defines the maximum number of trying to send a frame when the FrIf returns an error.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel		
BSW Parameter		**BSW Type**	
FrTpMaxRn		INTEGER-PARAM-DEF	
BSW Description			
This parameter defines the maximum number of retries (if retry is configured 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**	
FrTpAckType		ENUMERATION-PARAM-DEF	
BSW Description			
This parameter defines the type of acknowledgement which is used for the specific channel.			
M2 Template	**M2 Description**		
System Template	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		ENUMERATION-PARAM-DEF	
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**		
System Template	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**	
FrTpLm		ENUMERATION-PARAM-DEF	
BSW Description			
This specifies the maximum message length for the particular channel.			
M2 Template	**M2 Description**		
System Template	This specifies the maximum message length for the particular channel.		
M2 Parameter			
TransportProtocols::FlexRayTpChannel.maximumMessageLength			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel		
BSW Parameter		**BSW Type**	
FrTpStMin		FLOAT-PARAM-DEF	
BSW Description			
This parameter defines the minimum amount of time between two succeeding CFs. Specified in seconds.			
M2 Template	**M2 Description**		
System Template	This attribute defines the minimum amount of time (separation Time) between two succeeding CFs. Specified in seconds.		
M2 Parameter			
TransportProtocols::FlexRayTpChannel.minimumSeparationTime			
Mapping Rule			**Mapping Type**
1:1 mapping			full

<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">FrTpTimeBr</td><td>FLOAT-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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_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.</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>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">FrTpTimeBuffer</td><td>FLOAT-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This parameter defines the time in seconds of waiting for the next try (if retry is activated) to get a Tx or Rx buffer.</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>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">FrTpTimeCs</td><td>FLOAT-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This parameter defines the time in seconds between the sending of two consecutive CFs or between a CF and a FC (for Transmit Cancellation) or between reception of an FC or AF and sending of the next CF or a FC (for Transmit Cancellation). It is obvious that FRTP_TIME_CS + FRTP_TIMEOUT_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.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">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).</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">TransportProtocols::FlexRayTpChannel.timeCs</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</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrTpTimeFrIf</td><td>FLOAT-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This parameter defines the time in seconds of waiting for the next try (if retry is activated) to send via FrIf_Transmit.</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>

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpTimeoutAr		FLOAT-PARAM-DEF
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpTimeoutAs		FLOAT-PARAM-DEF
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 or FC (in case of Transmit Cancellation)).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel	
BSW Parameter		**BSW Type**
FrTpTimeoutBs		FLOAT-PARAM-DEF
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel
BSW Parameter	**BSW Type**
FrTpTimeoutCr	FLOAT-PARAM-DEF
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**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection
BSW Parameter	**BSW Type**
FrTpConnection	PARAM-CONF-CONTAINER-DEF
BSW Description	
This container contains the configuration (parameters) of one FlexRay TP connection. A connection can only belong to one channel.	
M2 Template	**M2 Description**
System Template	A connection within a channel identifies the sender and the receiver of this particular communication. The FlexRayTp module routes a Pdu through this connection. .
M2 Parameter	
TransportProtocols::FlexRayTpChannel:FlexRayTpConnection	
Mapping Rule	**Mapping Type**
Container must be created for each existing FlexRayTpConnection	full

BSW Module	BSW Context
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection
BSW Parameter	**BSW Type**
FrTpMultRec	BOOLEAN-PARAM-DEF
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**
System Template	Information can be derived from the System Template.
M2 Parameter	
CoreCommunication::Npdu	
Mapping Rule	**Mapping Type**
If the NPdu is transmitted in Frames on different clusters set FrTpMultRec to "true"	full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection	
BSW Parameter		**BSW Type**
FrTpLa		INTEGER-PARAM-DEF
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**	
System Template	An ECUs TP address on the referenced channel. This represents the diagnostic Address.	
M2 Parameter		
CoreTopology::CommunicationConnector.tpAddress		
Mapping Rule		**Mapping Type**
FrTpConnection contains a reference to the CommunicationConnector.		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection	
BSW Parameter		**BSW Type**
FrTpRa		INTEGER-PARAM-DEF
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**	
System Template	An ECUs TP address on the referenced channel. This represents the diagnostic Address.	
M2 Parameter		
CoreTopology::CommunicationConnector.tpAddress		
Mapping Rule		**Mapping Type**
FrTpConnection contains a reference to the CommunicationConnector.		full

<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>
<tr><td colspan="2">FrTpConPduRef</td><td>REFERENCE-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">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).</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">TransportProtocols::FlexRayTpChannel:FlexRayTpConnection.transmitPdu</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">FrTpConnection contains a reference to the Npdu.</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/FrTpRxSdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrTpRxSdu</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3"></td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">Reference to the IPdu that is segmented by the Transport Protocol.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">TransportProtocols::FlexRayTpChannel:FlexRayTpConnection.pdu</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">container must be created if reference from FlexRayTpConnection to Ipdu exists.</td><td>full</td></tr>
</table>

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection/FrTpRxSdu	
BSW Parameter		**BSW Type**
FrTpRxSduRef		REFERENCE-PARAM-DEF
BSW Description		
M2 Template	**M2 Description**	
System Template	Reference to the IPdu that is segmented by the Transport Protocol.	
M2 Parameter		
TransportProtocols::FlexRayTpChannel:FlexRayTpConnection.pdu		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection/FrTpTxSdu	
BSW Parameter		**BSW Type**
FrTpTxSdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
M2 Template	**M2 Description**	
System Template	Reference to the IPdu that is segmented by the Transport Protocol.	
M2 Parameter		
TransportProtocols::FlexRayTpChannel:FlexRayTpConnection.pdu		
Mapping Rule		**Mapping Type**
container must be created if reference from FlexRayTpConnection to Ipdu exists.		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpConnection/FrTpTxSdu	
BSW Parameter		**BSW Type**
FrTpSduTxId		INTEGER-PARAM-DEF
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.		
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		REFERENCE-PARAM-DEF
BSW Description		
M2 Template	**M2 Description**	
System Template	Reference to the IPdu that is segmented by the Transport Protocol.	
M2 Parameter		
TransportProtocols::FlexRayTpChannel:FlexRayTpConnection.pdu		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpPdu	
BSW Parameter		**BSW Type**
FrTpPdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
M2 Template	**M2 Description**	
System Template	This is a PDU of the Transport Layer. The main purpose of the TP Layer is to segment and reassemble I-PDUs	
M2 Parameter		
CoreCommunication::NPdu		
Mapping Rule		**Mapping Type**
container must be created for each Npdu		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpPdu	
BSW Parameter		**BSW Type**
FrTpPduId		INTEGER-PARAM-DEF
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.		
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/FrTpPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrTpPduDirection</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This parameter defines the direction of the PDU.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">This reference allows to specify explicitly which Frame is received/sent by the connected ECU on the connected channel.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">can be derived from CommConnectorPort-FrameTriggering association</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">if Npdu is transmitted in a frame that is received by the ECU than direction must be set to ”FrTpRx”. if Npdu is transmitted in a frame that is transmitted by the ECU than direction must be set to ”FrTpTx”.</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/FrTpPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrTpPduRef</td><td>REFERENCE-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3"></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>

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpPduFc
BSW Parameter	**BSW Type**
FrTpPduFc	PARAM-CONF-CONTAINER-DEF
BSW Description	
This is the identifier of the FlexRay Interface PDUs (Fr N-PDU, Fr L-SDU) in which the Transport Layer Flow Control and Acknowledgement Frames of this channel should be transmitted.	
M2 Template	**M2 Description**
System Template	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.
M2 Parameter	
TransportProtocols::FlexRayTpChannel:FlexRayTpConnection.flowControlPdu	
Mapping Rule	**Mapping Type**
Container must be created if the FrTpChannel contains the flowControlPdu reference to the NPdu	full

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpPduFc
BSW Parameter	**BSW Type**
FrTpPduFcId	INTEGER-PARAM-DEF
BSW Description	
This is the identifier of the FlexRay Interface PDUs (Fr N-PDU, Fr L-SDU) in which the Transport Layer Flow Control and Acknowledgement Frames of this channel should be transmitted.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpPduFc	
BSW Parameter		**BSW Type**
FrTpPduFcDirection		ENUMERATION-PARAM-DEF
BSW Description		
This parameter defines the direction of the PDU.		
M2 Template	**M2 Description**	
System Template	This reference allows to specify explicitly which Frame is received/sent by the connected ECU on the connected channel.	
M2 Parameter		
can be derived from CommConnectorPort-FrameTriggering association		
Mapping Rule		**Mapping Type**
if the Npdu is transmitted in a frame that is received by the ECU than direction must be set to ”FrTpRx”. if the Npdu is transmitted in a frame that is transmitted by the ECU than direction must be set to ”FrTpTx”.		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpChannel/FrTpPduFc	
BSW Parameter		**BSW Type**
FrTpPduFcRef		REFERENCE-PARAM-DEF
BSW Description		
M2 Template	**M2 Description**	
System Template	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	
M2 Parameter		
TransportProtocols::FlexRayTpChannel:FlexRayTpConnection.flowControlPdu		
Mapping Rule		**Mapping Type**
reference must be created if the FrTpChannel contains the flowControlPdu reference to the NPdu		full

9.7 Lin Interface Mapping

BSW Module	BSW Context
LinIf	LinIf/LinIfGeneral
BSW Parameter	**BSW Type**
LinIfGeneral	PARAM-CONF-CONTAINER-DEF
BSW Description	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
these parameters are global for the LIN interface, and will typically be configured 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	BOOLEAN-PARAM-DEF
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	BOOLEAN-PARAM-DEF
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**
may be derived from other parameters, depending on whether mutliple drivers are configured.	local

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGeneral
BSW Parameter	**BSW Type**
LinIfNcOptionalRequestSupported	BOOLEAN-PARAM-DEF
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**
may be derived from other parameters, considering whether Assign NAD and Conditional Chang NAD frames are configured.	local

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGeneral
BSW Parameter	**BSW Type**
LinIfTpSupported	BOOLEAN-PARAM-DEF
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	BOOLEAN-PARAM-DEF
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/LinIfGlobalConfig	
BSW Parameter		**BSW Type**
LinIfGlobalConfig		PARAM-CONF-CONTAINER-DEF
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**
LinIfTimeBase		FLOAT-PARAM-DEF
BSW Description		
The time-base for this channel in s (normally 0.002, 0.005 or 0.010s)		
M2 Template	**M2 Description**	
Sys-T		
M2 Parameter		
SystemTemplate::Fibex::Fibex4Lin::LinTopology::LinMaster.timeBase		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfChannel		PARAM-CONF-CONTAINER-DEF
BSW Description		
M2 Template	**M2 Description**	
Sys-T		
M2 Parameter		
SystemTemplate::Fibex::FibexCore::CoreTopology::CommunicationConnector		
Mapping Rule		**Mapping Type**
LIN IF channels are being described based on the available channels configured in the LIN driver module. 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)		partial

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfChannelId		INTEGER-PARAM-DEF
BSW Description		
Internal ID for the channel on LIN Interface level		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
assigned locally in ECU configuration		local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfScheduleRequestQueueLength		INTEGER-PARAM-DEF
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 as of AUTOSAR R 2.1		local

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame
BSW Parameter	**BSW Type**
LinIfFrame	PARAM-CONF-CONTAINER-DEF
BSW Description	
Generic container for all types of LIN frames.	
M2 Template	**M2 Description**
Sys-T	
M2 Parameter	
SystemTemplate::Fibex::FibexCore::CoreTopology::PhysicalChannel.frameTriggering	
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**
LinIfFramePriority	INTEGER-PARAM-DEF
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**
Sys-T	
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**
LinIfLength	INTEGER-PARAM-DEF
BSW Description	
Length of the LIN SDU in bytes.	
M2 Template	**M2 Description**
Sys-T	
M2 Parameter	
SystemTemplate::Fibex::FibexCore::CoreCommunication:Frame.frameLength	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame	
BSW Parameter		**BSW Type**
LinIfPid		INTEGER-PARAM-DEF
BSW Description		
Protected ID of the LIN frame. There is no reason to calculate the Parity in run-time.		
M2 Template	**M2 Description**	
Sys-T		
M2 Parameter		
SystemTemplate::Fibex::FibexCore::CoreTopology::PhysicalChannel.frameTriggering.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**
LinIfTxTargetPduId		INTEGER-PARAM-DEF
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/LinIfFrame	
BSW Parameter		**BSW Type**
LinIfChecksumType		ENUMERATION-PARAM-DEF
BSW Description		
Type of checksum that the frame is using.		
M2 Template	**M2 Description**	
Sys-T	Type of checksum that the frame is using.	
M2 Parameter		
SystemTemplate::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**
LinIfFrameType		ENUMERATION-PARAM-DEF
BSW Description		
Type of frame that is described (e.g. sporadic frame). Note that types 7-11 are the fixed MRF types. The sporadic slot is not found among the frame types. A sporadic slot is a set of sporadic frames.		
M2 Template	**M2 Description**	
Sys-T		
M2 Parameter		
RelativelyScheduledTiming, SubstitutionFrame, AssignNadTiming, AssignFrameIdTiming, UnassignFrameIdTiming, DataTiming		
Mapping Rule		**Mapping Type**
		partial

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame	
BSW Parameter		**BSW Type**
LinIfFrameName		STRING-PARAM-DEF
BSW Description		
Optional frame name used to cross-reference with a LDF		
M2 Template	**M2 Description**	
Sys-T		
M2 Parameter		
SystemTemplate::Fibex::FibexCore::CoreCommunication::Frame.shortName		
Mapping Rule		**Mapping Type**
		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfFixedFrameSdu	
BSW Parameter		**BSW Type**
LinIfFixedFrameSdu		PARAM-CONF-CONTAINER-DEF
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**
LinIfFixedFrameSduBytePos		INTEGER-PARAM-DEF
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	
BSW Parameter		**BSW Type**
LinIfFixedFrameSduByteVal		INTEGER-PARAM-DEF
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/LinIfSubstitutionFrames	
BSW Parameter		**BSW Type**
LinIfSubstitutionFrames		PARAM-CONF-CONTAINER-DEF
BSW Description		
List of unconditional Frames that can be sent in an event-triggered Frame or a sporadic Frame slot.		
M2 Template	**M2 Description**	
System Template	A LIN specific extension of the common FRAME to enable the usual frame handling of a placeholder frame that is substituted at runtime.	
M2 Parameter		
Fibex4Lin::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		REFERENCE-PARAM-DEF
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**	
System Template	Collecting the frames that are substituted by the refering one	
M2 Parameter		
Fibex4Lin::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/LinIfPduDirection/LinIfInternalPdu	
BSW Parameter		**BSW Type**
LinIfInternalPdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
Represents a Diagnostic or Configuration frame : no Message ID (no PduId).		
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**
LinIfRxPdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
represents a received PDU/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**	
LinIfRxPduId		INTEGER-PARAM-DEF	
BSW Description			
Identifier of the frame for the LIN Interface			
M2 Template	**M2 Description**		
System Template	To describe a frames identifier on the communication system, usualy with a fixed identifierValue.		
M2 Parameter			
FrameTriggering::identifier			
Mapping Rule			**Mapping Type**
1:1 mapping			

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfPduDirection/LinIfRxPdu		
BSW Parameter		**BSW Type**	
LinIfRxPduRef		REFERENCE-PARAM-DEF	
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/LinIfSlave-ToSlavePdu		
BSW Parameter		**BSW Type**	
LinIfSlaveToSlavePdu		PARAM-CONF-CONTAINER-DEF	
BSW Description			
represents a slave-to-slave PDU/frame. Master does only send the header but doesn't receive the response. Added for completeness			
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**
LinIfTxPdu		PARAM-CONF-CONTAINER-DEF
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**
LinIfTxPduId		INTEGER-PARAM-DEF
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		REFERENCE-PARAM-DEF
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/LinIfMaster	
BSW Parameter		**BSW Type**
LinIfMaster		PARAM-CONF-CONTAINER-DEF
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**	
Sys-T		
M2 Parameter		
SystemTemplate::Fibex::Fibex4Lin::LinTopology::LinMaster		
Mapping Rule		**Mapping Type**
		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfMaster	
BSW Parameter		**BSW Type**
LinIfJitter		INTEGER-PARAM-DEF
BSW Description		
Specifies the difference and the maximum and the minimum delay (ms)		
M2 Template	**M2 Description**	
Sys-T		
M2 Parameter		
SystemTemplate::Fibex::Fibex4Lin::LinTopology::LinMaster.timeBase		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable	
BSW Parameter		**BSW Type**
LinIfScheduleTable		PARAM-CONF-CONTAINER-DEF
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**	
Sys-T		
M2 Parameter		
SystemTemplate::Fibex::Fibex4Lin::LinCommunication::LinScheduleTable		
Mapping Rule		**Mapping Type**
		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable	
BSW Parameter		**BSW Type**
LinIfSchedulePriority		INTEGER-PARAM-DEF
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**	
Sys-T		
M2 Parameter		
SystemTemplate::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		INTEGER-PARAM-DEF
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**
LinIfResumePosition		ENUMERATION-PARAM-DEF
BSW Description		
Defines, where a schedule table shall be proceeded in case if 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		ENUMERATION-PARAM-DEF
BSW Description		
The schedule table can be executed in two different modes.		
M2 Template	**M2 Description**	
Sys-T		
M2 Parameter		
SystemTemplate::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**
LinIfScheduleTableName		STRING-PARAM-DEF
BSW Description		
Optional schedule name used to cross-reference with a LDF. This parameter shall always be accompanied by LIN_IF_SCHEDULE_INDEX.		
M2 Template	**M2 Description**	
Sys-T		
M2 Parameter		
SystemTemplate::Fibex::Fibex4Lin::LinCommunication::LinScheduleTable.shortName		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable/LinIfEntry	
BSW Parameter		**BSW Type**
LinIfEntry		PARAM-CONF-CONTAINER-DEF
BSW Description		
Describes an entry in the schedule table (also known as Frame Slot).		
M2 Template	**M2 Description**	
Sys-T		
M2 Parameter		
SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing::RelativelyScheduledTiming		
Mapping Rule		**Mapping Type**
Each (sub)class or RelativelyScheduledTiming is the reason for a LinIfEntry. RelativelyScheduledTiming.scheduleTableName decides to which schedule table the LinIfEntry belongs.		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable/LinIfEntry	
BSW Parameter		**BSW Type**
LinIfEntryIndex		INTEGER-PARAM-DEF
BSW Description		
Position of the Frame Entry in the Schedule Table.		
M2 Template	**M2 Description**	
System Template	Relative position of the frame described by this timing in the schedule table	
M2 Parameter		
RelativelyScheduledTiming.positionInTable		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable/LinIfEntry	
BSW Parameter		**BSW Type**
LinIfDelay		FLOAT-PARAM-DEF
BSW Description		
Delay to next frame in schedule table in [s]		
M2 Template	**M2 Description**	
Sys-T	Relative delay between this frame and the start of the successor frame in the schedule table in seconds	
M2 Parameter		
SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing::RelativelyScheduledTiming.delay		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable/LinIfEntry	
BSW Parameter		**BSW Type**
LinIfCollisionResolvingRef		REFERENCE-PARAM-DEF
BSW Description		
Reference to the schedule table, which resolves the collision.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable/LinIfEntry		
BSW Parameter		**BSW Type**	
LinIfFrameRef		REFERENCE-PARAM-DEF	
BSW Description			
Reference to the frames that belong to this schedule table entry.			
M2 Template	**M2 Description**		
System Template	Specification of a sending behaviour where the transmission order is predefined.		
M2 Parameter			
LinFrameTriggering.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/LinIfSlave		
BSW Parameter		**BSW Type**	
LinIfSlave		PARAM-CONF-CONTAINER-DEF	
BSW Description			
The Node attributes of the Slaves are provided with these parameter.			
M2 Template	**M2 Description**		
System Template	Describing the properties of the refering ecu as a LIN slave.		
M2 Parameter			
LinCommunicationController::LinSlave			
Mapping Rule			**Mapping Type**
Container must be created if ECU is LinSlave in the System Description			full

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave		
BSW Parameter		**BSW Type**	
LinIfConfiguredNad		INTEGER-PARAM-DEF	
BSW Description			
Definition of the initial node address			
M2 Template	**M2 Description**		
System Template	To distinguish LIN slaves that are used twice or more within the same cluster.		
M2 Parameter			
LinSlave:ConfiguredNad			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave	
BSW Parameter		**BSW Type**
LinIfFunctionId		INTEGER-PARAM-DEF
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**
LinIfResponseErrorBitPos		INTEGER-PARAM-DEF
BSW Description		
Specifies the frame and the position in the frame		
M2 Template	**M2 Description**	
System Template	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.	
M2 Parameter		
SystemTemplate::Fibex::Fibex4Lin::LinTopology::LinSlave:LinErrorResponse.responseErrorPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave	
BSW Parameter		**BSW Type**
LinIfSupplierId		INTEGER-PARAM-DEF
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		INTEGER-PARAM-DEF	
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/LinIfSlave		
BSW Parameter		**BSW Type**	
LinIfProtocolVersion		STRING-PARAM-DEF	
BSW Description			
Defines the LIN Protocol version which is used by the slave.			
M2 Template	**M2 Description**		
Sys-T			
M2 Parameter			
SystemTemplate::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**	
LinIfResponseErrorEventRef		REFERENCE-PARAM-DEF	
BSW Description			
Reference to DEM Event			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave		
BSW Parameter		**BSW Type**	
LinIfResponseErrorFrameRef		REFERENCE-PARAM-DEF	
BSW Description			
Reference to the frame which contains the response error bit.			
M2 Template	**M2 Description**		
System Template	Reference to an unconditional frame that transmits the response error. The referenced LinFrameTriggering shall contain a reference to an unconditionalFrame.		
M2 Parameter			
SystemTemplate::Fibex::Fibex4Lin::LinTopology::LinSlave::LinErrorResponse.frameTriggering			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave/LinIfNodeComposition		
BSW Parameter		**BSW Type**	
LinIfNodeComposition		PARAM-CONF-CONTAINER-DEF	
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		STRING-PARAM-DEF	
BSW Description			
M2 Template	**M2 Description**		
Sys-T			
M2 Parameter			
SystemTemplate::Fibex::Fibex4Lin::LinTopology::LinSlave.shortName			
Mapping Rule			**Mapping Type**
1:1 mapping			full

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>LinIf</td><td colspan="2">LinIf/LinIfGlobalConfig/LinIfChannel/LinIfWakeUpSource</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">LinIfWakeUpSource</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This container contains the configuration (parameters) needed to configure a wakeup capable channel</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>

9.8 Lin Driver Mapping

BSW Module	BSW Context	
Lin	Lin/LinGeneral	
BSW Parameter		**BSW Type**
LinGeneral		PARAM-CONF-CONTAINER-DEF
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		BOOLEAN-PARAM-DEF
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**
LinVersionInfoApi		BOOLEAN-PARAM-DEF
BSW Description		
Switches the Lin_GetVersionInfo function ON or OFF.		
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>Lin</td><td colspan="2">Lin/LinGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">LinIndex</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.</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>Lin</td><td colspan="2">Lin/LinGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">LinTimeoutDuration</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Specifies the maximum number of loops for blocking function until a timeout is raised in short term wait loops</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>Lin</td><td colspan="2">Lin/LinGlobalConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">LinGlobalConfig</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.</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>partial</td></tr>
</table>

BSW Module	BSW Context	
Lin	Lin/LinGlobalConfig/LinChannel	
BSW Parameter		**BSW Type**
LinChannel		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the configuration (parameters) of the LIN Controller(s).		
M2 Template	**M2 Description**	
Sys-T		
M2 Parameter		
SystemTemplate::Fibex::FibexCore::CoreTopology::PhysicalChannel		
Mapping Rule		**Mapping Type**
a LinChannel container is constructed per CommunicationConnector belonging to the CommunicationController associated with the owning Lin Module container		partial

BSW Module	BSW Context	
Lin	Lin/LinGlobalConfig/LinChannel	
BSW Parameter		**BSW Type**
LinChannelWakeUpSupport		BOOLEAN-PARAM-DEF
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**
LinChannelBaudRate		INTEGER-PARAM-DEF
BSW Description		
Specifies the baud rate of the LIN channel		
M2 Template	**M2 Description**	
Sys-T		
M2 Parameter		
SystemTemplate::Fibex::FibexCore::CoreTopology::CommunicationCluster.speed		
Mapping Rule		**Mapping Type**
		full

BSW Module	BSW Context		
Lin	Lin/LinGlobalConfig/LinChannel		
BSW Parameter		**BSW Type**	
LinChannelId		INTEGER-PARAM-DEF	
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**	
LinClockRef		REFERENCE-PARAM-DEF	
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

9.9 LinTP Mapping

BSW Module	**BSW Context**
LinTp	LinTp/LinTpGeneral
BSW Parameter	**BSW Type**
LinTpGeneral	PARAM-CONF-CONTAINER-DEF
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	BOOLEAN-PARAM-DEF
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/LinTpGlobalConfig
BSW Parameter	**BSW Type**
LinTpGlobalConfig	PARAM-CONF-CONTAINER-DEF
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**
System Template	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	
LinTransportProtocol:LinTpChannel	
Mapping Rule	**Mapping Type**
Container must be created if a LinTpChannel is described	full

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig	
BSW Parameter		**BSW Type**
LinTpNumberOfRxNSdu		INTEGER-PARAM-DEF
BSW Description		
Number of transport protocol messages that can be received for all channels this node is connected to. Can't that be calculated from the NSdus?		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig	
BSW Parameter		**BSW Type**
LinTpNumberOfTxNSdu		INTEGER-PARAM-DEF
BSW Description		
Number of transport protocol messages that can be transmitted for all channels this node is connected to.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig/LinTpRxNSdu	
BSW Parameter		**BSW Type**
LinTpRxNSdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
For each received N-SDU on any channel the node is connected to.		
M2 Template	**M2 Description**	
System Template	Reference to the IPdu that is segmented by the Transport Protocol.	
M2 Parameter		
LinTpChannel.linTpNsdu		
Mapping Rule		**Mapping Type**
Container must be created for each received NSdu.		full

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig/LinTpRxNSdu	
BSW Parameter		**BSW Type**
LinTpDl		INTEGER-PARAM-DEF
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**	
System Template	The size of the IPDU in bits.	
M2 Parameter		
CoreCommunication:IPdu.length		
Mapping Rule		**Mapping Type**
The data length of the RxNsdu is correlated to the IPdu Length		full

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig/LinTpRxNSdu	
BSW Parameter		**BSW Type**
LinTpRxNSduId		INTEGER-PARAM-DEF
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		INTEGER-PARAM-DEF
BSW Description		
A N-SDU transported on LIN is identified using the NAD for the specific slave.		
M2 Template	**M2 Description**	
System Template	An ECUs TP address on the referenced channel. This represents the diagnostic Address.	
M2 Parameter		
CommunicationConnector.tpAddress (targetAddress)		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig/LinTpRxNSdu	
BSW Parameter		**BSW Type**
LinTpRxNSduPduRef		REFERENCE-PARAM-DEF
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/LinTpTxNSdu	
BSW Parameter		**BSW Type**
LinTpTxNSdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
For each transmitted N-SDU on any channel the node is connected to.		
M2 Template	**M2 Description**	
System Template	This is a PDU of the Transport Layer. The main purpose of the TP Layer is to segment and reassemble I-PDUs.	
M2 Parameter		
CoreCommunication::Npdu		
Mapping Rule		**Mapping Type**
Container must be created if a NPdu is transmitted		full

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig/LinTpTxNSdu	
BSW Parameter		**BSW Type**
LinTpTxNSduId		INTEGER-PARAM-DEF
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		INTEGER-PARAM-DEF
BSW Description		
A N-SDU transported on LIN is identified using the NAD for the specific slave.		
M2 Template	**M2 Description**	
System Template	An ECUs TP address on the referenced channel. This represents the diagnostic Address.	
M2 Parameter		
CoreCommunication::CommunicationConnector.TpAddress		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinTp	LinTp/LinTpGlobalConfig/LinTpTxNSdu	
BSW Parameter		**BSW Type**
LinTpTxNSduPduRef		REFERENCE-PARAM-DEF
BSW Description		
Reference to the global PDU		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

9.10 Can Interface Mapping

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet	
BSW Parameter		**BSW Type**
CanIfInitConfigSet		PARAM-CONF-CONTAINER-DEF
BSW Description		
This is a multiple configuration set container.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
SystemTemplate:Topology:ECUInstance:CommunicationController:CommunicationConnector: PhysicalChannel		
Mapping Rule		**Mapping Type**
Container must be created if ECU is connected to a CAN channel		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfControllerConfig	
BSW Parameter		**BSW Type**
CanIfControllerConfig		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the configuration (parameters) of all addressed CAN controllers by each underlying CAN driver.		
M2 Template	**M2 Description**	
System Template	CommunicationControllers of the ECU.	
M2 Parameter		
SystemTemplate:Topology:ECUInstance:CommunicationController		
Mapping Rule		**Mapping Type**
Container must be created for each CAN Controller that is part of the ECU		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfControllerConfig	
BSW Parameter		**BSW Type**
CanIfDriverNameRef		REFERENCE-PARAM-DEF
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/CanIfInitConfigSet/CanIfControllerConfig	
BSW Parameter		**BSW Type**
CanIfNetworkIdRef		REFERENCE-PARAM-DEF
BSW Description		
Reference to CanIfNetworkConfig container. This parameter refers to the Network Id to which the current controller belongs to. Note that more than one controllers can be connected to same CAN Network.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfControllerConfig/CanIfInitControllerConfig	
BSW Parameter		**BSW Type**
CanIfInitControllerConfig		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the references to the configuration setup of each underlying CAN driver.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
Container must be created for each controller that is described in the system template		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfControllerConfig/CanIfInitControllerConfig	
BSW Parameter		**BSW Type**
CanIfControllerRefConfig		STRING-PARAM-DEF
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/CanIfInitConfigSet/CanIfDispatchConfig	
BSW Parameter		**BSW Type**
CanIfDispatchConfig		PARAM-CONF-CONTAINER-DEF
BSW Description		
Holds: Call-out functions for CANIF with respect to Network. This call-out functions defined in this container are common to all the configured networks.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfDispatchConfig	
BSW Parameter		**BSW Type**
CanIfBusoffNotifFun		FUNCTION-NAME-DEF
BSW Description		
Name of target BusOff notification services to target upper layers (PduRouter, CanNm, CanTp and ComplexDeviceDrivers).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfDispatchConfig	
BSW Parameter		**BSW Type**
CanIfWakeupNotifFun		FUNCTION-NAME-DEF
BSW Description		
Name of target wakeup notification services to target upper layers (PduRouter, CanNm, CanTp and ComplexDeviceDrivers). If parameter is 0 no call-out function is configured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context
CanIf	CanIf/CanIfInitConfigSet/CanIfDispatchConfig
BSW Parameter	**BSW Type**
CanIfWakeupValidNotifFun	FUNCTION-NAME-DEF
BSW Description	
Name of target wakeup validation notification services to target upper layers (ECU State Manager). If parameter is 0 no call-out function is configured.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig
BSW Parameter	**BSW Type**
CanIfDriverConfig	PARAM-CONF-CONTAINER-DEF
BSW Description	
Configuration parameters for all the undelying CAN driver are aggregated under this container.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig
BSW Parameter	**BSW Type**
CanIfBusoffNotifFun	BOOLEAN-PARAM-DEF
BSW Description	
Selects whether BusOff indication notification is supported.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig	
BSW Parameter		**BSW Type**
CanIfReceiveIndFun		BOOLEAN-PARAM-DEF
BSW Description		
Selects whether receive indication notification is supported.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig	
BSW Parameter		**BSW Type**
CanIfTransmitCancFun		BOOLEAN-PARAM-DEF
BSW Description		
Selects whether transmit cancellation is supported		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig	
BSW Parameter		**BSW Type**
CanIfTxConfirmation		BOOLEAN-PARAM-DEF
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/CanIfInitConfigSet/CanIfDriverConfig		
BSW Parameter		**BSW Type**	
CanIfWakeupNotifFun		BOOLEAN-PARAM-DEF	
BSW Description			
Selects whether wakeup indication notification is supported.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig		
BSW Parameter		**BSW Type**	
CanIfDriverRef		REFERENCE-PARAM-DEF	
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/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitDriverConfig		
BSW Parameter		**BSW Type**	
CanIfInitDriverConfig		PARAM-CONF-CONTAINER-DEF	
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/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitDriverConfig
BSW Parameter	**BSW Type**
CanIfDriverRefConfig	STRING-PARAM-DEF
BSW Description	
Reference to the CAN controller specific configuration setup.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitDriverConfig
BSW Parameter	**BSW Type**
CanIfRefConfig	STRING-PARAM-DEF
BSW Description	
Selects the CAN Interface specific configuration setup.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig
BSW Parameter	**BSW Type**
CanIfInitHohConfig	PARAM-CONF-CONTAINER-DEF
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/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig	
BSW Parameter		**BSW Type**
CanIfRefConfigSet		STRING-PARAM-DEF
BSW Description		
Selects the CAN Interface specific configuration setup.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig/CanIfHrhConfig	
BSW Parameter		**BSW Type**
CanIfHrhConfig		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains parameters specific to HRH.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig/CanIfHrhConfig	
BSW Parameter		**BSW Type**
CanIfSoftwareFilterHrh		BOOLEAN-PARAM-DEF
BSW Description		
Enables/Disables the software filtering for this particular HRH.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig/CanIfHrhConfig	
BSW Parameter		**BSW Type**
CanIfHrhType		ENUMERATION-PARAM-DEF
BSW Description		
Defines the HRH type i.e, whether its a BasicCan or FullCan.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig/CanIfHrhConfig	
BSW Parameter		**BSW Type**
CanIfCanControllerHrhIdRef		REFERENCE-PARAM-DEF
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/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig/CanIfHrhConfig/CanIfHrhListConfig	
BSW Parameter		**BSW Type**
CanIfHrhListConfig		PARAM-CONF-CONTAINER-DEF
BSW Description		
Defines the parameters required for configuring list of CANIDs for a given same HRH.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig/CanIfHrhConfig/CanIfHrhListConfig	
BSW Parameter		**BSW Type**
CanIfHrhListCanId		INTEGER-PARAM-DEF
BSW Description		
Every instance of this parameter holds single CAN ID. Since there can be more than one CANID assigned to same HRH, the lowermultiplicity is 1 and uppermultiplicity is *. Note that for every CANID mentioned with this parameter should have corresponding entry in the "CanIfRxPduConfig" container.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig/CanIfHrhConfig/CanIfHrhRangeConfig	
BSW Parameter		**BSW Type**
CanIfHrhRangeConfig		PARAM-CONF-CONTAINER-DEF
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/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig/CanIfHrhConfig/CanIfHrhRangeConfig	
BSW Parameter		**BSW Type**
CanIfRxPduLowerCanId		INTEGER-PARAM-DEF
BSW Description		
Lower CAN Identifier of a receive CAN L-PDU for identifier range definition, in which all CAN Ids shall pass the software filtering.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig/CanIfHrhConfig/CanIfHrhRangeConfig		
BSW Parameter		**BSW Type**	
CanIfRxPduUpperCanId		INTEGER-PARAM-DEF	
BSW Description			
Upper CAN Identifier of a receive CAN L-PDU for identifier range definition, in which all CAN Ids shall pass the software filtering.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig/CanIfHthConfig		
BSW Parameter		**BSW Type**	
CanIfHthConfig		PARAM-CONF-CONTAINER-DEF	
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/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig/CanIfHthConfig		
BSW Parameter		**BSW Type**	
CanIfHthType		ENUMERATION-PARAM-DEF	
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/CanIfInitConfigSet/CanIfDriverConfig/CanIfInitHohConfig/CanIfHthConfig		
BSW Parameter		**BSW Type**	
CanIfCanControllerHthIdRef		REFERENCE-PARAM-DEF	
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/CanIfInitConfigSet/CanIfInitConfig		
BSW Parameter		**BSW Type**	
CanIfInitConfig		PARAM-CONF-CONTAINER-DEF	
BSW Description			
This container contains parameters required for specific configuration setup.			
M2 Template	**M2 Description**		
System Template			
M2 Parameter			
Mapping Rule			**Mapping Type**
Container must be created if ECU is connected to a CAN Communication Cluster			local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig		
BSW Parameter		**BSW Type**	
CanIfNumberOfCanRXPduIds		INTEGER-PARAM-DEF	
BSW Description			
Total number of CanRxPduIds to be handled.			
M2 Template	**M2 Description**		
System Template			
M2 Parameter			
Mapping Rule			**Mapping Type**
For all frames whose are referenced in inputPorts and who are referenced from busses with name = CAN do CanIfNumberOfCanTXPduIds++			full

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig	
BSW Parameter		**BSW Type**
CanIfNumberOfCanTXPduIds		INTEGER-PARAM-DEF
BSW Description		
Total number of CanTxPduIds to be handled.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
Mapping Rule		**Mapping Type**
For all frames whose are referenced in outputPorts and who are referenced from busses with name = CAN do CanIfNumberOfCanTXPduIds++		full

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig	
BSW Parameter		**BSW Type**
CanIfNumberOfDynamicCanTXPduIds		INTEGER-PARAM-DEF
BSW Description		
Total number of dynamic CanTxPduIds to be handled.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
This can be calculated by adding up the number of configured CanIfTxPduConfig Containers with CanIfCanTxPduType set to Dynamic.		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig	
BSW Parameter		**BSW Type**
CanIfConfigSet		STRING-PARAM-DEF
BSW Description		
Selects the CAN Interface specific configuration setup.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
CanIfRxPduConfig		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the configuration (parameters) of each receive CAN L-PDU.		
M2 Template	**M2 Description**	
System Template	Data frame which is sent over a communication medium. Each Frame can be identified per channel by an Identifier (ID).	
M2 Parameter		
SystemTemplate:Communication:Frame		
Mapping Rule		**Mapping Type**
Container must be created for each CAN frame that is received		full

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
CanIfReadRxPduData		BOOLEAN-PARAM-DEF
BSW Description		
Enables and disables the Rx buffering for reading of received L-PDU data.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
CanIfReadRxPduNotifyStatus		BOOLEAN-PARAM-DEF
BSW Description		
Enables and disables receive indication for each receive CAN L-PDU for reading its' notification status.		
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>CanIf</td><td colspan="2">CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfRxPduConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanIfCanRxPduCanId</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">CAN Identifier of Receive CAN L-PDUs used by the CAN Interface. Exa: Software Filtering.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">To describe a frames identifier on the communication system, usualy with a fixed identifierValue.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">SystemTemplate:Communication:FrameTriggering:Identifier</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>CanIf</td><td colspan="2">CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfRxPduConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanIfCanRxPduDlc</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Data Length code of received CAN L-PDUs used by the CAN Interface. Exa: DLC check.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">The used length (in bytes) of the referencing frame. Should not be confused with a static byte length reserved for each frame by some platforms (e.g. FlexRay).</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">SystemTemplate:Communication:Frame:frameLength</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>CanIf</td><td colspan="2">CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfRxPduConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanIfCanRxPduId</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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</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>

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
CanIfRxPduIdCanIdType		ENUMERATION-PARAM-DEF
BSW Description		
The parameter defines the CANID type. The value of this parameter shall be used for validating CANID of Rx L-PDUs.		
M2 Template	**M2 Description**	
System Template	The CAN bus supports 11-Bit ("Standard") and 29-Bit ("Extended") identifiers. This attributes constrains a CAN bus to the selected formats. On Extended- Addressing it is also possible to have 11-Bit and 29-Bit CAN-identifiers. Predefined values are "Standard" and "Extended".	
M2 Parameter		
CanCluster:CanAddressingMode		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
CanIfRxUserType		ENUMERATION-PARAM-DEF
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**	
System Template		
M2 Parameter		
FibexCore:CoreCommunication:PduToFrameMapping		
Mapping Rule		**Mapping Type**
This information can be derived from the SysT: CanTp: PduToFrameMapping contains a reference to a N-Pdu PduR: PduToFrameMapping contains a reference to an I-Pdu CanNm: IPdu.PduType is Nm		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
CanIfUserRxInd		FUNCTION-NAME-DEF
BSW Description		
Name of target indication services to target upper layers (PduRouter, CanNm, CanTp and ComplexDeviceDrivers). If parameter is 0 no call-out function is configured.		
M2 Template	**M2 Description**	
System Template	A PduToFrameMapping defines the position of a PDU within a frame	
M2 Parameter		
SystemTemplate:FibexCore:CoreCommunication:PduToFrameMapping SystemTemplate:FibexCore:CoreCommunication:IPdu.PduType		
Mapping Rule		**Mapping Type**
This can be calculated for PduR, CanNm and CanTp. For ComplexDeviceDrivers this has to be set by the developer: CanTp: PduToFrameMapping contains a N-Pdu PduR: PduToFrameMapping contains an I-Pdu CanNm: PduType is Nm		partial

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
CanIfCanRxPduHrhRef		REFERENCE-PARAM-DEF
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/CanIfInitConfigSet/CanIfInitConfig/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
CanIfCanRxPduIdNetworkRef		REFERENCE-PARAM-DEF
BSW Description		
Reference to the CAN network ID to which the receive CAN L-PDU belongs to.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
CommunicationCluster.shortName		
Mapping Rule		**Mapping Type**
reference must be created in the ECUC		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfRxPduConfig	
BSW Parameter		**BSW Type**
PduIdRef		REFERENCE-PARAM-DEF
BSW Description		
Reference to the ”global” Pdu structure to allow harmonization of handle IDs in the COM-Stack.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>CanIf</td><td colspan="3">CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfTxPduConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanIfTxPduConfig</td><td colspan="2">PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This container contains the configuration (parameters) of each transmit CAN L-PDU.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">Data frame which is sent over a communication medium. Each Frame can be identified per channel by an Identifier (ID).</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">SystemTemplate:Communication:Frame</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Container must be created for each CAN frame that is transmitted</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>CanIf</td><td colspan="3">CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfTxPduConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanIfReadTxPduNotifyStatus</td><td colspan="2">BOOLEAN-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Enables and disables the API for reading the notification status of transmit L-PDUs.</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/CanIfInitConfigSet/CanIfInitConfig/CanIfTxPduConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanIfCanTxPduCanId</td><td colspan="2">INTEGER-PARAM-DEF</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. Range: 11 Bit For Standard CAN Identifier ... 29 Bit For Extended CAN identifier</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">To describe a frames identifier on the communication system, usualy with a fixed identifierValue.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">SystemTemplate:Communication:FrameTriggering:FrameID</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>

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfTxPduConfig		
BSW Parameter		**BSW Type**	
CanIfCanTxPduDlc		INTEGER-PARAM-DEF	
BSW Description			
Data length code of transmit CAN L-PDUs used by the CAN Driver for CAN L-PDU transmission.			
M2 Template	**M2 Description**		
System Template	The used length (in bytes) of the referencing frame. Should not be confused with a static byte length reserved for each frame by some platforms (e.g. FlexRay).		
M2 Parameter			
SystemTemplate:Communication:Frame:frameLength			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfTxPduConfig		
BSW Parameter		**BSW Type**	
CanIfCanTxPduId		INTEGER-PARAM-DEF	
BSW Description			
ECU wide unique, symbolic handle for transmit CAN L-PDU. The CanIfCanTxPduId is configurable at pre-compile and post-built time.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfTxPduConfig		
BSW Parameter		**BSW Type**	
CanIfCanTxPduType		ENUMERATION-PARAM-DEF	
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/CanIfInitConfigSet/CanIfInitConfig/CanIfTxPduConfig	
BSW Parameter		**BSW Type**
CanIfTxPduIdCanIdType		ENUMERATION-PARAM-DEF
BSW Description		
CAN Identifier of transmit CAN L-PDUs used by the CAN Driver for CAN L-PDU transmission.		
M2 Template	**M2 Description**	
System Template	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.	
M2 Parameter		
SystemTemplate:Fibex:Fibex4Can:CanCommunication:CanFrameTriggering:canAdressingMode		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfTxPduConfig	
BSW Parameter		**BSW Type**
CanIfTxUserType		ENUMERATION-PARAM-DEF
BSW Description		
This parameter defines the type of the transmit confirmation call-out called to the corresponding upper layer the used TargetTxPduId belongs to.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfTxPduConfig	
BSW Parameter		**BSW Type**
CanIfTxPduTxConfirmCallOut		FUNCTION-NAME-DEF
BSW Description		
Name of target confirmation services to target upper layers (PduR, CanNm and CanTp. 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**	
System Template	A PduToFrameMapping defines the position of a PDU within a frame	
M2 Parameter		
SystemTemplate:FibexCore:CoreCommunication:PduToFrameMapping SystemTemplate:FibexCore:CoreCommunication:Pdu.PduType		
Mapping Rule		**Mapping Type**
This can be calculated for PduR, CanNm andCanTp. For ComplexDeviceDrivers this has to be set by the developer: CanTp: PduToFrameMapping contains a N-Pdu PduR: PduToFrameMapping contains an I-Pdu CanNm: PduType is Nm		partial

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfTxPduConfig	
BSW Parameter		**BSW Type**
CanIfCanTxPduHthRef		REFERENCE-PARAM-DEF
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/CanIfInitConfigSet/CanIfInitConfig/CanIfTxPduConfig	
BSW Parameter		**BSW Type**
CanIfCanTxPduIdNetworkRef		REFERENCE-PARAM-DEF
BSW Description		
This parameter holds the NETWORK ID to which this PDU belongs to.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
CommінicationCluster.ShortName		
Mapping Rule		**Mapping Type**
Reference must be set in the ECUC.		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitConfigSet/CanIfInitConfig/CanIfTxPduConfig	
BSW Parameter		**BSW Type**
PduIdRef		REFERENCE-PARAM-DEF
BSW Description		
Reference to the ”global” Pdu structure to allow harmonization of handle IDs in the COM-Stack.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitConfigSet/CanIfNetworkConfig		
BSW Parameter		**BSW Type**	
CanIfNetworkConfig		PARAM-CONF-CONTAINER-DEF	
BSW Description			
This container contains parameter required for configuring Network.			
M2 Template	**M2 Description**		
System Template	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.		
M2 Parameter			
SystemTemplate:Fibex:Fibex4Can:CanTopology:CanCluster			
Mapping Rule			**Mapping Type**
Container must be created for each CanCluster in the System description			full

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitConfigSet/CanIfNetworkConfig		
BSW Parameter		**BSW Type**	
CanIfNetworkId		INTEGER-PARAM-DEF	
BSW Description			
CanIfNetworkId is a Logical handle that defines the corresponding CAN network. This will ranges from 0..max. number of underlying supported networks The value of Network Id is unique across the system.			
M2 Template	**M2 Description**		
System Template			
M2 Parameter			
SystemTemplate:Fibex:Fibex4Can:CanTopology:CanCluster:ShortName			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitConfigSet/CanIfNetworkConfig		
BSW Parameter		**BSW Type**	
CanIfWakeupSource		ENUMERATION-PARAM-DEF	
BSW Description			
Defines different types of sources for controller wakeup.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfNetworkConfig	
BSW Parameter		**BSW Type**
CanIfWakeupSupport		ENUMERATION-PARAM-DEF
BSW Description		
Enables wakeup support and defines the source device of a wakeup event.		
M2 Template	**M2 Description**	
System Template	May the ECU be woken up by this CAN Controller? TRUE: wake up is possible FALSE: wake up is not supported Note: This flag may only be set to TRUE if the feature is supported by both hardware and basic software.	
M2 Parameter		
In case of Controller this can be derived from SystemTemplate: SystemTemplate:CoreTopology: EcuInstance:CommunicationController.wakeUpByControllerSupported		
Mapping Rule		**Mapping Type**
		partial

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfNetworkConfig/CanIfInitNetworkConfig	
BSW Parameter		**BSW Type**
CanIfInitNetworkConfig		PARAM-CONF-CONTAINER-DEF
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/CanIfInitConfigSet/CanIfNetworkConfig/CanIfInitNetworkConfig	
BSW Parameter		**BSW Type**
CanIfControllerConfigSet		STRING-PARAM-DEF
BSW Description		
Selects the CAN controller specific configuration setup of the CAN Interface.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfNetworkConfig/CanIfInitNetworkConfig	
BSW Parameter		**BSW Type**
CanIfNetRefConfigSet		STRING-PARAM-DEF
BSW Description		
Selects the CAN Interface specific configuration setup.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitConfigSet/CanIfTransceiverDrvConfig	
BSW Parameter		**BSW Type**
CanIfTransceiverDrvConfig		PARAM-CONF-CONTAINER-DEF
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/CanIfInitConfigSet/CanIfTransceiverDrvConfig	
BSW Parameter		**BSW Type**
CanIfTrcvWakeupNotification		BOOLEAN-PARAM-DEF
BSW Description		
Selects whether wakeup indication notification is supported.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitConfigSet/CanIfTransceiverDrvConfig		
BSW Parameter		**BSW Type**	
CanIfTrcvVendorId		INTEGER-PARAM-DEF	
BSW Description			
Name of the corresponding CAN Transceiver. Range : 0..max. number of underlying supported CAN transceivers This information has to be derived from the CAN Transceiver configuration.			
M2 Template	**M2 Description**		
ECU Resource Template			
M2 Parameter			
CommunicationTransceiver:ShortName			
Mapping Rule			**Mapping Type**
			1:1 mapping

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitConfigSet/CanIfTransceiverDrvConfig/CanIfInitTrcvConfig		
BSW Parameter		**BSW Type**	
CanIfInitTrcvConfig		PARAM-CONF-CONTAINER-DEF	
BSW Description			
This container contains the references to the transceiver initialization configuration setup of each underlying CAN Transceiver Driver.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitConfigSet/CanIfTransceiverDrvConfig/CanIfInitTrcvConfig		
BSW Parameter		**BSW Type**	
CanIfTrcvRefConfigset		STRING-PARAM-DEF	
BSW Description			
Selects the CAN Interface specific configuration setup.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitConfigSet/CanIfTransceiverDrvConfig/CanIfTrcvDeviceConfig	
BSW Parameter		**BSW Type**
CanIfTrcvDeviceConfig		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the references to the transceiver network configuration setup of each underlying CAN Transceiver Driver.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPrivateConfig	
BSW Parameter		**BSW Type**
CanIfPrivateConfig		PARAM-CONF-CONTAINER-DEF
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/CanIfPrivateConfig	
BSW Parameter		**BSW Type**
CanIfDlcCheck		BOOLEAN-PARAM-DEF
BSW Description		
Selects whether the DLC check is supported		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfPrivateConfig	
BSW Parameter		**BSW Type**
CanIfNumberOfTxBuffers		INTEGER-PARAM-DEF
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.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfPrivateConfig	
BSW Parameter		**BSW Type**
CanIfSoftwareFilterType		ENUMERATION-PARAM-DEF
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**
The available software filter types depend on the implementation of the driver. This has to be set by the developer.		

BSW Module	BSW Context	
CanIf	CanIf/CanIfPublicConfig	
BSW Parameter		**BSW Type**
CanIfPublicConfig		PARAM-CONF-CONTAINER-DEF
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/CanIfPublicConfig	
BSW Parameter		**BSW Type**
CanIfDevErrorDetect		BOOLEAN-PARAM-DEF
BSW Description		
Enables and disables the development error detection and notification mechanism.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfPublicConfig	
BSW Parameter		**BSW Type**
CanIfMulDrvSupport		BOOLEAN-PARAM-DEF
BSW Description		
Selects support for multiple CAN Drivers.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfPublicConfig	
BSW Parameter		**BSW Type**
CanIfPollingBusoff		BOOLEAN-PARAM-DEF
BSW Description		
Selects polling mode for BusOff events for each underlying CAN driver.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
It is up to the developer to decide whether polling mode or interrupt mode shall be used.		local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfPublicConfig		
BSW Parameter		**BSW Type**	
CanIfPollingReceive		BOOLEAN-PARAM-DEF	
BSW Description			
Selects polling mode for receive events for each underlying CAN driver.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
It is up to the developer to decide whether polling mode or interrupt mode shall be used.			

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfPublicConfig		
BSW Parameter		**BSW Type**	
CanIfPollingTransmit		BOOLEAN-PARAM-DEF	
BSW Description			
Selects polling mode for transmit events for each underlying CAN driver.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
It is up to the developer to decide whether polling mode or interrupt mode shall be used.			local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfPublicConfig		
BSW Parameter		**BSW Type**	
CanIfPollingWakeup		BOOLEAN-PARAM-DEF	
BSW Description			
Selects polling mode for wakeup events for each underlying CAN driver.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
It is up to the developer to decide whether polling mode or interrupt mode shall be used.			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfPublicConfig		
BSW Parameter		**BSW Type**	
CanIfReadRxPduDataApi		BOOLEAN-PARAM-DEF	
BSW Description			
Enables / Disables the API CanIf_ReadRxPduData() for reading received L-PDU data.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfPublicConfig		
BSW Parameter		**BSW Type**	
CanIfReadRxPduNotifStatusApi		BOOLEAN-PARAM-DEF	
BSW Description			
Enables / Disables the API CanIf_ReadRxNotifStatus() for reading the notification status of receive L-PDUs.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfPublicConfig		
BSW Parameter		**BSW Type**	
CanIfReadTxPduNotifStatusApi		BOOLEAN-PARAM-DEF	
BSW Description			
Enables / Disables the API CanIf_ReadTxNotifStatus() for reading the notification status of transmit L-PDUs.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPublicConfig	
BSW Parameter		**BSW Type**
CanIfSetDynamicTxIdApi		BOOLEAN-PARAM-DEF
BSW Description		
Enables / Disables the API CanIf_SetDynamicTxId(), for reconfiguring CAN ID of Dynamic Transmit L-PDUs.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPublicConfig	
BSW Parameter		**BSW Type**
CanIfVersionInfoApi		BOOLEAN-PARAM-DEF
BSW Description		
Enables / Disables the API CanIf_GetVersionInfo(), for reading the version information of CAN Interface module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPublicConfig	
BSW Parameter		**BSW Type**
CanIfWakeupEventApi		BOOLEAN-PARAM-DEF
BSW Description		
Enables / Disables the API, for 'wakeup notification'. When value of this parameter is set to FALSE, no call-out functions will be provided to CAN Driver Module. If it is set to TRUE, CANIF provides call-out functions for each underlying CAN Driver,		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
CanIf	CanIf/CanIfPublicConfig		
BSW Parameter		**BSW Type**	
CanIfNumOfCanHwUnits		INTEGER-PARAM-DEF	
BSW Description			
Holds: 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**
The setting of this parameter depends on the used hardware.			

BSW Module	BSW Context		
CanIf	CanIf/CanIfPublicConfig		
BSW Parameter		**BSW Type**	
CanIfNumOfNetworks		INTEGER-PARAM-DEF	
BSW Description			
Number of served CAN networks.			
M2 Template	**M2 Description**		
System Template			
M2 Parameter			
SystemTemplate:Fibex:Fibex4Can:CanTopology:CanCluster			
Mapping Rule			**Mapping Type**
Number of used CAN Clusters			Calculable

9.11 Can Driver Mapping

BSW Module	**BSW Context**	
Can	Can/CanConfigSet	
BSW Parameter		**BSW Type**
CanConfigSet		PARAM-CONF-CONTAINER-DEF
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/CanController	
BSW Parameter		**BSW Type**
CanController		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the configuration parameters of the CAN controller(s).		
M2 Template	**M2 Description**	
System Template	The communication controller is a dedicated hardware device by means of which hosts are sending frames to and receiving frames from the communication medium.	
M2 Parameter		
SystemTemplate:Fibex:CoreTopology:Topology:EcuInstance:CommunicationController		
Mapping Rule		**Mapping Type**
Container must be created for each CAN controller that is described in the System Template		full

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanControllerActivation		BOOLEAN-PARAM-DEF
BSW Description		
Defines if a CAN controller is used in the configuration.		
M2 Template	**M2 Description**	
System Template		
M2 Parameter		
SystemTemplate:Fibex:CoreTopology:Topology:CommunicationConnector		
Mapping Rule		**Mapping Type**
This is set to ON if a communication connector in the System Template references the communication controller.		full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanControllerBaudRate		INTEGER-PARAM-DEF
BSW Description		
Specifies the buadrate of the controller in kbps.		
M2 Template	**M2 Description**	
ECU Resource Template	Abstract element to describe communication speed. This can be either a fixed value, a range or a list of allowed communication speed.	
M2 Parameter		
CommunicationPeripheral:CommunicationHWPort:CommunicationSpeed		
Mapping Rule		**Mapping Type**
CommunicationControllerMapping in the System Template specifies the Mapping between the Communication Controller in the System Template and the the CommunicationPeripheral in the ECU Resource Template.		full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanControllerId		INTEGER-PARAM-DEF
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**	
SystemTemplate		
M2 Parameter		
SystemTemplate:Fibex:CoreTopology:Topology:EcuInstance:CommunicationController:ShortName		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanControllerPropSeg		INTEGER-PARAM-DEF
BSW Description		
Specifies propagation delay in time quantas.		
M2 Template	**M2 Description**	
System Template	The propagation time segment in quanta.	
M2 Parameter		
Fibex4Can:CanCommunicationController:CanControllerConfiguration.propagationDelay		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanControllerSeg1		INTEGER-PARAM-DEF
BSW Description		
Specifies phase segment 1 in time quantas.		
M2 Template	**M2 Description**	
System Template	The number of quanta before the sampling point	
M2 Parameter		
SystemTemplate:Fibex:Fibec4Can:CanTopology:CanCommunicationController:CanControllerConfiguration.timeSeg1		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanControllerSeg2		INTEGER-PARAM-DEF
BSW Description		
Specifies phase segment 2 in time quantas.		
M2 Template	**M2 Description**	
System Template	The number of quanta after the sampling point	
M2 Parameter		
SystemTemplate:Fibex:Fibec4Can:CanTopology:CanCommunicationController:CanControllerConfiguration.timeSeg2		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanControllerTimeQuanta		FLOAT-PARAM-DEF
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**	
System Template		
M2 Parameter		
Mapping Rule		**Mapping Type**
This parameter can be calculated from timeSeg1, timeSeg2, propagationDelay		full

BSW Module	BSW Context		
Can	Can/CanConfigSet/CanController		
BSW Parameter		**BSW Type**	
CanCpuClockRef		REFERENCE-PARAM-DEF	
BSW Description			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
Can	Can/CanConfigSet/CanController/CanFilterMask		
BSW Parameter		**BSW Type**	
CanFilterMask		PARAM-CONF-CONTAINER-DEF	
BSW Description			
This container contains the configuration (parameters) of the CAN Filter Mask(s).			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
Can	Can/CanConfigSet/CanController/CanFilterMask		
BSW Parameter		**BSW Type**	
CanFilterMaskValue		INTEGER-PARAM-DEF	
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**
			local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanHardwareObject		PARAM-CONF-CONTAINER-DEF
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**
CanIdValue		INTEGER-PARAM-DEF
BSW Description		
Specifies (together with the filter mask)- the identifiers range that passes the hardware filter for of RX objects.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanObjectId		INTEGER-PARAM-DEF
BSW Description		
Holds the handle ID of HRH or HTH. The value of this parameter is unique in a given CAN Driver, and it should start with 0 and continue without any gaps. The HRH and HTH Ids are defined under two different name-spaces. Example: HRH0-0, HRH1-1, HTH0-2, HTH1-3		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanHandleType		ENUMERATION-PARAM-DEF
BSW Description		
Specifies the type (Full-CAN or Basic-CAN) of a hardware object.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanIdType		ENUMERATION-PARAM-DEF
BSW Description		
Specifies whether the IdValue is of type - standard identifier - extended identifier - mixed mode		
M2 Template	**M2 Description**	
System Template	The CAN bus supports 11-Bit (”Standard”) and 29-Bit (”Extended”) identifiers. This attributes constrains a CAN bus to the selected formats. On Extended- Addressing it is also possible to have 11-Bit and 29-Bit CAN-identifiers. Predefined values are ”Standard” and ”Extended”.	
M2 Parameter		
SystemTemplate:Fibex:Fibex4Can:CanTopology:CanCluster:CanAdressingMode		
Mapping Rule		**Mapping Type**
		partial

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanObjectType		ENUMERATION-PARAM-DEF
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

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Can</td><td colspan="2">Can/CanConfigSet/CanHardwareObject</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanControllerRef</td><td>REFERENCE-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Reference to CAN Controller to which the HOH is associated to.</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/CanHardwareObject</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanFilterMaskRef</td><td>REFERENCE-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Reference to the filter mask that is used for hardware filtering togerther with the CAN_ID_VALUE</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/CanGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanGeneral</td><td>PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This container contains the parameters related each CAN Driver Unit.</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>Can</td><td colspan="3">Can/CanGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanDevErrorDetection</td><td colspan="2">BOOLEAN-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Switches the Development Error Detection and Notification ON or OFF.</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>Can</td><td colspan="3">Can/CanGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanHardwareCancellation</td><td colspan="2">BOOLEAN-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Specifies if hardware cancellation shall be supported.ON or OFF</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>Can</td><td colspan="3">Can/CanGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanMultiplexedTransmission</td><td colspan="2">BOOLEAN-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Specifies if multiplexed transmission shall be supported.ON or OFF</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>

BSW Module	**BSW Context**
Can	Can/CanGeneral
BSW Parameter	**BSW Type**
CanVersionInfoApi	BOOLEAN-PARAM-DEF
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**
CanWakeup_Support	BOOLEAN-PARAM-DEF
BSW Description	
CAN driver support for wakeup over CAN Bus.	
M2 Template	**M2 Description**
System Template	Driver support for wakeup over Bus.
M2 Parameter	
CoreTopology:EcuInstance.wakeUpOverBusSupported	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Can	Can/CanGeneral
BSW Parameter	**BSW Type**
CanIndex	INTEGER-PARAM-DEF
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**	
CanTimeoutDurationFactor		INTEGER-PARAM-DEF	
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**	
CanCpuClockRef		REFERENCE-PARAM-DEF	
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

9.12 CanTP Mapping

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpGeneral	
BSW Parameter		**BSW Type**
CanTpGeneral		PARAM-CONF-CONTAINER-DEF
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**
CanTpDevErrorDetect		BOOLEAN-PARAM-DEF
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**
CanTpTc		BOOLEAN-PARAM-DEF
BSW Description		
Preprocessor switch for enabling Transmit Cancellation.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**		
CanTp	CanTp/CanTpGeneral		
BSW Parameter		**BSW Type**	
CanTpMainFunctionPeriod		FLOAT-PARAM-DEF	
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/CanTpRxNSdu		
BSW Parameter		**BSW Type**	
CanTpRxNSdu		PARAM-CONF-CONTAINER-DEF	
BSW Description			
The following parameters needs to be configured for each CAN N-SDU that the CanTp module shall receive.			
M2 Template	**M2 Description**		
System Template	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		
M2 Parameter			
TransportProtocols::CanTransportProtocol:CanTpConnectionChannel			
Mapping Rule			**Mapping Type**
This container must be created for each IPdu that is transmitted via CanTP.			full

BSW Module	BSW Context	
CanTp	CanTp/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpBs		INTEGER-PARAM-DEF
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**	
System Template	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 Parameter		
TransportProtocols::CanTransportProtocol:CanTpConnectionChannel.blocksize		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpRxChannel		INTEGER-PARAM-DEF
BSW Description		
Link to the Rx connection channel, which has to be used for receiving this N-PDU.		
M2 Template	**M2 Description**	
System Template	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	
M2 Parameter		
TransportProtocols::CanTransportProtocol:CanTpConnectionChannel		
Mapping Rule		**Mapping Type**
CanTpRxChannel is described by CanTpConnectionChannel		full

BSW Module	BSW Context		
CanTp	CanTp/CanTpRxNSdu		
BSW Parameter		**BSW Type**	
CanTpRxDl		INTEGER-PARAM-DEF	
BSW Description			
Data Length Code of this RxNsdu. In case of variable message length, this value indicates the minimum data length. Depending on SF or FF N-SDU the value will be limited to 7 (6 for an extended addressing format) and 4095 respectively.			
M2 Template	**M2 Description**		
System Template	The used length (in bytes) of the referencing frame.		
M2 Parameter			
FibexCore::Communication::Frame.framelength			
Mapping Rule			**Mapping Type**
			full

BSW Module	BSW Context		
CanTp	CanTp/CanTpRxNSdu		
BSW Parameter		**BSW Type**	
CanTpRxWftMax		INTEGER-PARAM-DEF	
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/CanTpRxNSdu		
BSW Parameter		**BSW Type**	
CanTpSTmin		INTEGER-PARAM-DEF	
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**		
System Template	This attribute defines the minimum amount of time (separation Time) between two succeeding CFs. Specified in seconds.		
M2 Parameter			
TransportProtocols::CanTransportProtocolCanTpConnectionChannel.minimumSeparationTime			
Mapping Rule			**Mapping Type**
1:1 mapping			full

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>CanTp</td><td colspan="2">CanTp/CanTpRxNSdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanTpAddressingFormat</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">Declares which communication addressing mode is supported.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">TransportProtocols::CanTransportProtocolCanTpConnectionChannel.addressingFormat</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>CanTp</td><td colspan="2">CanTp/CanTpRxNSdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanTpRxPaddingActivation</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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)</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">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)</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">TransportProtocols::CanTransportProtocol:CanTpConnectionChannel.paddingActivation</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		
CanTp	CanTp/CanTpRxNSdu		
BSW Parameter		**BSW Type**	
CanTpRxTaType		ENUMERATION-PARAM-DEF	
BSW Description			
Declares the communication type of this Rx N-SDU.			
M2 Template	**M2 Description**		
System Template	Specifies the communication type: true: 1:n communication (Functional) false: 1:1 communication (Physical)		
M2 Parameter			
TransportProtocols::CanTransportProtocol:CanTpConnectionChannel.multicastAddressing			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
CanTp	CanTp/CanTpRxNSdu		
BSW Parameter		**BSW Type**	
CanTpNar		FLOAT-PARAM-DEF	
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/CanTpRxNSdu		
BSW Parameter		**BSW Type**	
CanTpNbr		FLOAT-PARAM-DEF	
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**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
CanTp	CanTp/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpNcr		FLOAT-PARAM-DEF
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanTp	CanTp/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpRxNSduRef		REFERENCE-PARAM-DEF
BSW Description		
Reference to a Pdu in the COM-Stack.		
M2 Template	**M2 Description**	
System Template	Reference to the IPdu that is segmented by the Transport Protocol.	
M2 Parameter		
TransportProtocols::CanTpConnectionChannel.tpSdu		
Mapping Rule		**Mapping Type**
Reference can be derived from the System Template		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpRxNSdu/CanTpNSa	
BSW Parameter		**BSW Type**
CanTpNSa		PARAM-CONF-CONTAINER-DEF
BSW Description		
Contains the parameters needed to configure each RxNSdu or TxNSdu with CanTpAddressingFormat set to CanTpExtended.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context
CanTp	CanTp/CanTpRxNSdu/CanTpNSa
BSW Parameter	**BSW Type**
CanTpNSa	INTEGER-PARAM-DEF
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**
System Template	The source of the TP connection.
M2 Parameter	
TransportProtocols::CanTpConnectionChannel.source	
Mapping Rule	**Mapping Type**
transport protocol source address can be derived from CommunicationConnector.tpAddress	full

BSW Module	BSW Context
CanTp	CanTp/CanTpRxNSdu/CanTpNTa
BSW Parameter	**BSW Type**
CanTpNTa	PARAM-CONF-CONTAINER-DEF
BSW Description	
The following parameters need to be configured for each RxNsdu or TxNsdu with the CanTpAddressingFormat set to CanTpExtended.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanTp	CanTp/CanTpRxNSdu/CanTpNTa
BSW Parameter	**BSW Type**
CanTpNTa	INTEGER-PARAM-DEF
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**
System Template	The target of the TP connection.
M2 Parameter	
TransportProtocols::CanTpConnectionChannel.target	
Mapping Rule	**Mapping Type**
transport protocol target address can be derived from CommunicationConnector.tpAddress	full

BSW Module	BSW Context	
CanTp	CanTp/CanTpRxNSdu/CanTpRxNPdu	
BSW Parameter		**BSW Type**
CanTpRxNPdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
Used for grouping of the ID of a PDU and the Reference to a PDU.		
M2 Template	**M2 Description**	
System Template	This is a PDU of the Transport Layer. The main purpose of the TP Layer is to segment and reassemble I-PDUs.	
M2 Parameter		
TransportProtocols::CanTransportProtocol:CanTpConnectionChannel.dataPdu		
Mapping Rule		**Mapping Type**
Container must be created for each received data NPdu.		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpRxNSdu/CanTpRxNPdu	
BSW Parameter		**BSW Type**
CanTpRxNPduId		INTEGER-PARAM-DEF
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/CanTpRxNSdu/CanTpRxNPdu	
BSW Parameter		**BSW Type**
CanTpRxNPduRef		REFERENCE-PARAM-DEF
BSW Description		
Reference to a Pdu in the COM-Stack.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>CanTp</td><td colspan="3">CanTp/CanTpRxNSdu/CanTpTxFcNPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanTpTxFcNPdu</td><td colspan="2">PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Used for grouping of the ID of a PDU and the Reference to a PDU.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</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">TransportProtocols::CanTransportProtocol:CanTpConnectionChannel.flowControlPdu</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Container must be created for each transmitted flowControl NPdu.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>CanTp</td><td colspan="3">CanTp/CanTpRxNSdu/CanTpTxFcNPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanTpTxFcNPduRef</td><td colspan="2">REFERENCE-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Reference to a Pdu in the COM-Stack.</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>

BSW Module	BSW Context	
CanTp	CanTp/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpTxNSdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
The following parameters needs to be configured for each CAN N-SDU that the CanTp module shall transmitt.		
M2 Template	**M2 Description**	
System Template	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).	
M2 Parameter		
TransportProtocols::CanTransportProtocol:CanTpConnectionChannel.dataPdu		
Mapping Rule		**Mapping Type**
container must be created for each transmitted data NPdu		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpTxChannel		INTEGER-PARAM-DEF
BSW Description		
Link to the connection channel which has to be used for transmission of this N-PDU.		
M2 Template	**M2 Description**	
System Template	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	
M2 Parameter		
TransportProtocols::CanTransportProtocol:CanTpConnectionChannel		
Mapping Rule		**Mapping Type**
CanTpConnectionChannel is described in the System Template		full

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>CanTp</td><td colspan="2">CanTp/CanTpTxNSdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanTpTxDl</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Data Length Code of this TxNsdu. In case of variable length message, this value indicates the minimum data length.</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>CanTp</td><td colspan="2">CanTp/CanTpTxNSdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanTpTxNSduId</td><td>INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Unique identifier to a structure that contains all useful information to process the transmission of a TxNsdu.</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>CanTp</td><td colspan="2">CanTp/CanTpTxNSdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanTpAddressingMode</td><td>ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">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).</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">Declares which communication addressing mode is supported.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">CanTransportProtocol:CanTpConnectionChannel.addressingFormat</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>CanTp</td><td colspan="3">CanTp/CanTpTxNSdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanTpTxPaddingActivation</td><td colspan="2">ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">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)</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">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)</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">CanTransportProtocol:CanTpConnectionChannel.paddingActivation</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>CanTp</td><td colspan="3">CanTp/CanTpTxNSdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanTpTxTaType</td><td colspan="2">ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Declares the communication type of this TxNsdu. Enumeration values: CanTpPhysical. Used for 1:1 communication. CanTpFunctional. Used for 1:n communication.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">Specifies the communication type: true: 1:n communication (Functional) false: 1:1 communication (Physical)</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">CanTransportProtocol:CanTpConnectionChannel.multicastAddressing</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>

BSW Module	BSW Context	
CanTp	CanTp/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpNas		FLOAT-PARAM-DEF
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanTp	CanTp/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpNbs		FLOAT-PARAM-DEF
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanTp	CanTp/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpNcs		FLOAT-PARAM-DEF
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
CanTp	CanTp/CanTpTxNSdu		
BSW Parameter		**BSW Type**	
CanTpTxNSduRef		REFERENCE-PARAM-DEF	
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/CanTpTxNSdu/CanTpNSa		
BSW Parameter		**BSW Type**	
CanTpNSa		PARAM-CONF-CONTAINER-DEF	
BSW Description			
Contains the parameters needed to configure each RxNSdu or TxNSdu with CanTpAddressingFormat set to CanTpExtended.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanTp	CanTp/CanTpTxNSdu/CanTpNSa		
BSW Parameter		**BSW Type**	
CanTpNSa		INTEGER-PARAM-DEF	
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**		
System Template	The source of the TP connection.		
M2 Parameter			
TransportProtocols::CanTpConnectionChannel.source			
Mapping Rule			**Mapping Type**
Transport protocol source address can be derived from CommunicationConnector.tpAddress			full

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>CanTp</td><td colspan="3">CanTp/CanTpTxNSdu/CanTpNTa</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanTpNTa</td><td colspan="2">PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The following parameters need to be configured for each RxNsdu or TxNsdu with the CanTpAddressingFormat set to CanTpExtended.</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>CanTp</td><td colspan="3">CanTp/CanTpTxNSdu/CanTpNTa</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanTpNTa</td><td colspan="2">INTEGER-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">If an RxNsdu or a TxNsdu is configured for extended addressing format, this parameter contains the transport protocol target address's value.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">The target of the TP connection.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">TransportProtocols::CanTpConnectionChannel.target</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Transport protocol target address can be derived from CommunicationConnector.tpAddress</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>CanTp</td><td colspan="3">CanTp/CanTpTxNSdu/CanTpRxFcNPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanTpRxFcNPdu</td><td colspan="2">PARAM-CONF-CONTAINER-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Used for grouping of the ID of a PDU and the Reference to a PDU.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</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">TransportProtocols::CanTransportProtocol:CanTpConnectionChannel.flowControlPdu</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Container must be created for each received flowControl NPdu.</td><td>local</td></tr>
</table>

BSW Module	BSW Context	
CanTp	CanTp/CanTpTxNSdu/CanTpRxFcNPdu	
BSW Parameter		**BSW Type**
CanTpRxFcNPduId		INTEGER-PARAM-DEF
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/CanTpTxNSdu/CanTpRxFcNPdu	
BSW Parameter		**BSW Type**
CanTpRxFcNPduRef		REFERENCE-PARAM-DEF
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/CanTpTxNSdu/CanTpTxNPdu	
BSW Parameter		**BSW Type**
CanTpTxNPdu		PARAM-CONF-CONTAINER-DEF
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

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>CanTp</td><td colspan="3">CanTp/CanTpTxNSdu/CanTpTxNPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">CanTpTxNPduRef</td><td colspan="2">REFERENCE-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Reference to a Pdu in the COM-Stack.</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>

9.13 Generic NM Interface

BSW Module	BSW Context		
Nm	Nm/NmGlobalConfig		
BSW Parameter		**BSW Type**	
NmGlobalConfig		PARAM-CONF-CONTAINER-DEF	
BSW Description			
This container contains all global configuration parameters of the Nm Interface.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
Nm	Nm/NmGlobalConfig		
BSW Parameter		**BSW Type**	
NmBusSynchronizationEnabled		BOOLEAN-PARAM-DEF	
BSW Description			
Pre-processor switch for enabling bus synchronization support. This feature is required for gateway nodes only.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
Nm	Nm/NmGlobalConfig		
BSW Parameter		**BSW Type**	
NmComControlEnabled		BOOLEAN-PARAM-DEF	
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	
Nm	Nm/NmGlobalConfig	
BSW Parameter		**BSW Type**
NmCoordinatorSupportEnabled		BOOLEAN-PARAM-DEF
BSW Description		
Switch to inform if NM coordinator needs to be supported.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Nm	Nm/NmGlobalConfig	
BSW Parameter		**BSW Type**
NmDevErrorDetect		BOOLEAN-PARAM-DEF
BSW Description		
Pre-processor switch for enabling development error detection and notification.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Nm	Nm/NmGlobalConfig	
BSW Parameter		**BSW Type**
NmMultipleChannelsEnabled		BOOLEAN-PARAM-DEF
BSW Description		
Pre-processor switch for enabling channel multiplicity support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
Nm	Nm/NmGlobalConfig		
BSW Parameter		**BSW Type**	
NmNodeDetectionEnabled		BOOLEAN-PARAM-DEF	
BSW Description			
Pre-processor switch for enabling the Request Repeat Message Request support.			
M2 Template	**M2 Description**		
System Template	switch for enabling the node detection support.		
M2 Parameter			
CoreTopology::CommunicationCluster.nmModeDetectionEnabled			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Nm	Nm/NmGlobalConfig		
BSW Parameter		**BSW Type**	
NmNodeIdEnabled		BOOLEAN-PARAM-DEF	
BSW Description			
Pre-processor switch for enabling the source node identifier.			
M2 Template	**M2 Description**		
SystemTemplate	switch for enabling the source node identifier.		
M2 Parameter			
CoreTopology::CommunicationCluster.nmNodeIdEnabled			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Nm	Nm/NmGlobalConfig		
BSW Parameter		**BSW Type**	
NmOsekSupportEnabled		BOOLEAN-PARAM-DEF	
BSW Description			
Switch to inform if NM coordinator needs to support direct OSEK NM.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Nm	Nm/NmGlobalConfig		
BSW Parameter		**BSW Type**	
NmPassiveModeEnabled		BOOLEAN-PARAM-DEF	
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**		
Nm	Nm/NmGlobalConfig		
BSW Parameter		**BSW Type**	
NmPduRxIndicationEnabled		BOOLEAN-PARAM-DEF	
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**		
Nm	Nm/NmGlobalConfig		
BSW Parameter		**BSW Type**	
NmRemoteSleepIndEnabled		BOOLEAN-PARAM-DEF	
BSW Description			
Pre-processor switch for enabling remote sleep indication support. This feature is required for gateway nodes only.			
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>Nm</td><td colspan="2">Nm/NmGlobalConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">NmStateChangeIndEnabled</td><td>BOOLEAN-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Pre-processor switch for enabling the CAN Network Management state change notification.</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>Nm</td><td colspan="2">Nm/NmGlobalConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">NmUserDataEnabled</td><td>BOOLEAN-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Pre-processor switch for enabling user data support.</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>Nm</td><td colspan="2">Nm/NmGlobalConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">NmVersionInfoApi</td><td>BOOLEAN-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Pre-processor switch for enabling version info API support.</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>

BSW Module	BSW Context		
Nm	Nm/NmGlobalConfig		
BSW Parameter		**BSW Type**	
NmAutosarGatewayRounds		INTEGER-PARAM-DEF	
BSW Description			
Number of rounds the coordinator shall keep a bus which runs AUTOSAR NM awake after all nodes including itself are ready to sleep.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
Nm	Nm/NmGlobalConfig		
BSW Parameter		**BSW Type**	
NmNumberOfChannels		INTEGER-PARAM-DEF	
BSW Description			
Number of NM channels allowed within one ECU.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
Nm	Nm/NmGlobalConfig		
BSW Parameter		**BSW Type**	
NmOsekGatewayRounds		INTEGER-PARAM-DEF	
BSW Description			
Number of rounds the coordinator shall keep a bus which runs OSEK NM awake after all nodes including itself are ready to sleep.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
Nm	Nm/NmGlobalConfig	
BSW Parameter		**BSW Type**
NmCycletimeMainFunction		FLOAT-PARAM-DEF
BSW Description		
The period between successive calls to the Main Function of the NM Interface in seconds.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Nm	Nm/NmGlobalConfig/NmChannelConfig	
BSW Parameter		**BSW Type**
NmChannelConfig		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the configuration (parameters) of the bus channel(s). The channel parameter shall be harmonized within the whole communication stack.		
M2 Template	**M2 Description**	
System Template	The CommunicationCluster is the main element to describe the topological connection of communicating ECUs.	
M2 Parameter		
CoreTopology::CommunicationCluster		
Mapping Rule		**Mapping Type**
Container must be created for each communication cluster.		full

BSW Module	BSW Context	
Nm	Nm/NmGlobalConfig/NmChannelConfig	
BSW Parameter		**BSW Type**
NmChannelId		INTEGER-PARAM-DEF
BSW Description		
Channel identification number of the corresponding channel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Nm</td><td colspan="3">Nm/NmGlobalConfig/NmChannelConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">NmBusType</td><td colspan="2">ENUMERATION-PARAM-DEF</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Identifies the bus type of the channel. LIN is not yet supported.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="3">The CommunicationCluster is the main element to describe the topological connection of communicating ECUs.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">CoreTopology::CommunicationCluster</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Information can be derived from the Topology description</td><td>full</td></tr>
</table>

9.14 Can Nm

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmGlobalConfig		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the global configuration parameter of the CanNm. The parameters and the parameters of the sub containers shall be mapped to the C data type CanNm_ConfigType (for parameters where it is possible) which is passed to the CanNm_Init function. This container is a MultipleConfigurationContainer (only for variant 3), i.e. this container and its sub-containers exit once per configuration set.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmBusLoadReductionEnabled		BOOLEAN-PARAM-DEF
BSW Description		
Pre-processor switch for enabling busload reduction support.		
M2 Template	**M2 Description**	
System Template	Switch for enabling busload reduction support.	
M2 Parameter		
CanCluster.nmBusLoadReductionEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmDevErrorDetect		BOOLEAN-PARAM-DEF
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		BOOLEAN-PARAM-DEF
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**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmImmediateTxconfEnabled		BOOLEAN-PARAM-DEF
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**
CanNmVersionInfoApi		BOOLEAN-PARAM-DEF
BSW Description		
Pre-processor switch for enabling version info API support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmConfigPtr		INTEGER-PARAM-DEF
BSW Description		
Pointer to configuration of CanNm.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmNumberOfChannels		INTEGER-PARAM-DEF
BSW Description		
Number of Can NM channels allowed within one ECU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmChannelConfig		PARAM-CONF-CONTAINER-DEF
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**
CanNmBusLoadReductionActive		BOOLEAN-PARAM-DEF
BSW Description		
This parameter defines if bus load reduction for the respective NM channel is active or not.		
M2 Template	**M2 Description**	
System Template	It determines if bus load reduction for the respective NM channel is active or not. True: active False: inactive	
M2 Parameter		
CanCluster.nmBusLoadReductionActive		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmChannelActive		BOOLEAN-PARAM-DEF
BSW Description		
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 NM instance 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**
CanNmNodeId		INTEGER-PARAM-DEF
BSW Description		
Node identifier of local node. This parameter is only valid if CanNmPassiveModeEnabled = False and CanNmNodeDetectionEnabled = True		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmPduLength		INTEGER-PARAM-DEF
BSW Description		
Defines the length of the NM PDU.		
M2 Template	**M2 Description**	
System Template	The size of the NmPDU in bits.	
M2 Parameter		
NmPdu.length		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmUserDataLentgh		INTEGER-PARAM-DEF
BSW Description		
Defines the length of the user data contained in the NM PDU		
M2 Template	**M2 Description**	
System Template	Defines the length in Bytes of the user data contained in the NM PDU	
M2 Parameter		
NmPdu.nmUserDataLength		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmPduCbvPosition		ENUMERATION-PARAM-DEF
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)		
M2 Template	**M2 Description**	
System Template	Defines the position of the control bit vector within the NM PDU (Bitpositon).	
M2 Parameter		
NmPdu.nmCbvPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig		
BSW Parameter		**BSW Type**	
CanNmPduNidPosition		ENUMERATION-PARAM-DEF	
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)			
M2 Template	**M2 Description**		
System Template	Defines the bitposition of the source node identifier within the NM PDU.		
M2 Parameter			
NmPdu.nmNidPosition			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig		
BSW Parameter		**BSW Type**	
CanNmMainFunctionPeriod		FLOAT-PARAM-DEF	
BSW Description			
Call cycle in seconds of CanNm_MainFunction_x for the respective instance.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig		
BSW Parameter		**BSW Type**	
CanNmMsgCycleOffset		FLOAT-PARAM-DEF	
BSW Description			
Time offset in the periodic transmission node. It determines the start delay of the transmission. Specified in seconds. This parameter is only valid if CanNmPassiveModeEnabled is False.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmMsgCycleTime		FLOAT-PARAM-DEF
BSW Description		
Period of a NM-message in seconds. It determines the periodic rate in the ”periodic transmission mode with bus load reduction” and is the basis for transmit scheduling in the ”periodic transmission mode without bus load reduction”. This parameter is only valid if CanNmPassiveModeEnabled is False.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmMsgReducedtime		FLOAT-PARAM-DEF
BSW Description		
Node specific bus cycle time in the periodic transmission mode with bus load reduction. Specified in seconds. This parameter is only valid if CanNmBusLoadReductionEnabled == True and CanNmBus-LoadReductionActive == True and CanNmPassiveModeEnabled == False Otherwise this parameter is not used.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmMsgTimeoutTime		FLOAT-PARAM-DEF
BSW Description		
Transmission Timeout of NM-message. If there is no transmission confirmation by the CAN Interface within this timeout, the CANNM module shall give an error notification. This parameter is only valid if CANNM_PASSIVE_MODE_ENABLED is disabled.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmRemoteSleepIndTime		FLOAT-PARAM-DEF
BSW Description		
Timeout for Remote Sleep Indication. It defines the time in seconds how long it shall take to recognize that all other nodes are ready to sleep. Typically it should be equal to: n * CanNmMsgCycleTime, where n denotes the number of NM-Messages that are normally sent before Remote Sleep Indication is detected. The value of n decremented by one determines the amount of lost NM-Messages that can be tolerated by the Remote Sleep Indication procedure. The value 0 denotes that no Remote Sleep Indication functionality is configured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmRepeatMessageTime		FLOAT-PARAM-DEF
BSW Description		
Timeout for Repeat Message State. It defines the time in seconds how long the NM shall stay in the Repeat Message State. Typically it should be equal to: n * CanNmMsgCycleTime, where n denotes the number of NM-Messages that are normally sent in the Repeat Message State. The value of n decremented by one determines the amount of lost NM-Messages that can be tolerated by the node detection procedure. The value 0 denotes that no Repeat Message State is configured. It means that Repeat Message State is transient what implicates that it is left immediately after entrance and in result no start-up stability is guaranteed and no node detection procedure is possible.		
M2 Template	**M2 Description**	
System Template	It defines how long the NM shall stay in the Repeat Message State (in seconds)	
M2 Parameter		
CanCluster.nmRepeatMessageStateTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>CanNm</td><td colspan="2">CanNm/CanNmGlobalConfig/CanNmChannelConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanNmTimeoutTime</td><td>FLOAT-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Network Timeout for NM-Messages. It denotes the time in seconds how long the NM shall stay in the Network Mode before transition into Prepare Bus-Sleep Mode shall take place. It shall be equal for all nodes in the cluster. It shall be greater than CanNmMsgCycleTime. Typically it should be equal to: n * CanNmMsgCycleTime, where n denotes the number of NM-Message cycle times in the Ready Sleep State before transition into the Bus-Sleep Mode is initiated. The value of n decremented by one determines the amount of lost NM-Messages that can be tolerated by the coordination algorithm.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">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.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">CanCluster.nmTimeoutTime</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>CanNm</td><td colspan="2">CanNm/CanNmGlobalConfig/CanNmChannelConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanNmWaitBusSleepTime</td><td>FLOAT-PARAM-DEF</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Timeout for bus calm down phase. It denotes the time in seconds how long the NM shall stay in the Prepare Bus-Sleep Mode before transition into Bus-Sleep Mode shall take place. It shall be equal for all nodes in the cluster. It shall be long enough to make all Tx-buffer empty.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">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.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">CanCluster.nmWaitBusSleepTime</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	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig/CanNmTxPdu	
BSW Parameter		**BSW Type**
CanNmTxPdu		PARAM-CONF-CONTAINER-DEF
BSW Description		
This container contains the CanNmTxPduId and the CanNmTxPduRef.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig/CanNmTxPdu	
BSW Parameter		**BSW Type**
CanNmTxPduId		INTEGER-PARAM-DEF
BSW Description		
L-PDU handle of the NM PDU to be transmitted by CanIf_Transmit and passed to CanNm_TxConfirmation by the CanIf. This handle specifies the corresponding CAN frame ID and implicitly the CAN driver instance as well as the corresponding CAN controller device. This parameter is only valid if CanNmPassiveModeEnabled = False. ImplementationType: PduIdType		
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		REFERENCE-PARAM-DEF
BSW Description		
The reference to the common PDU structure.		
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).

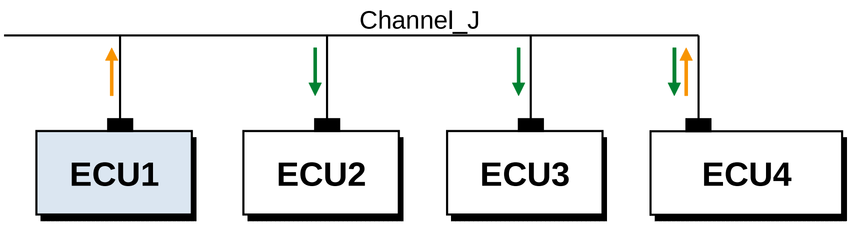

Figure A.1: Example of network setup with one Frame being received and sent on the same ECU and channel

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.

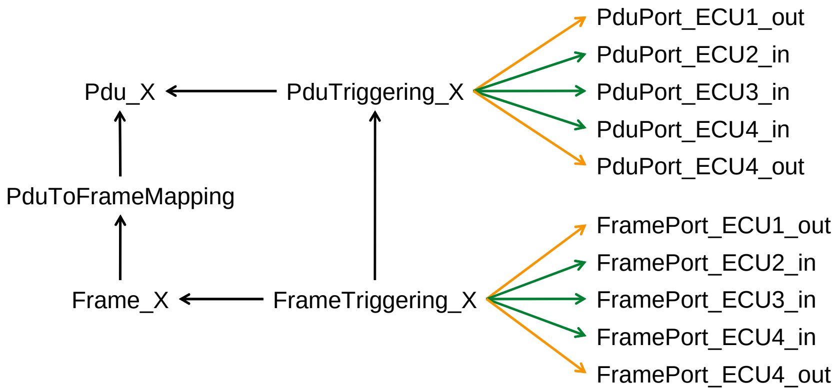

Figure A.2: Structure to reflect the frame- and pdu-triggering setup of one Frame being received and sent on the same ECU and channel

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.

B Detailed Representation of InstanceRef Associations in the System Template

B.1 Data Mapping

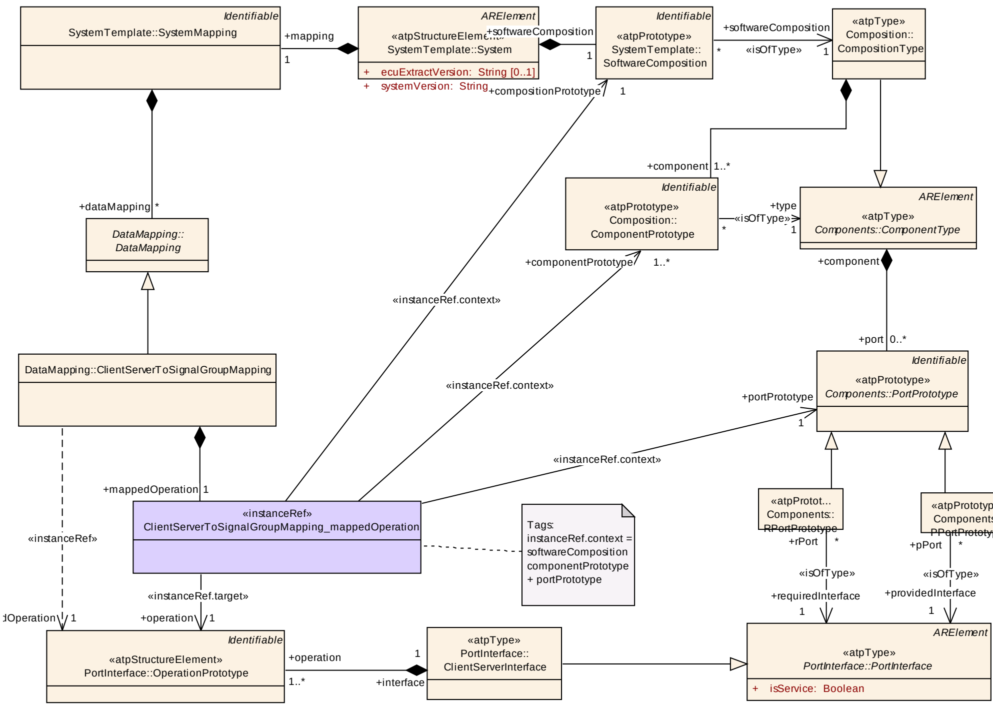

Figure B.1: Operation Mapping

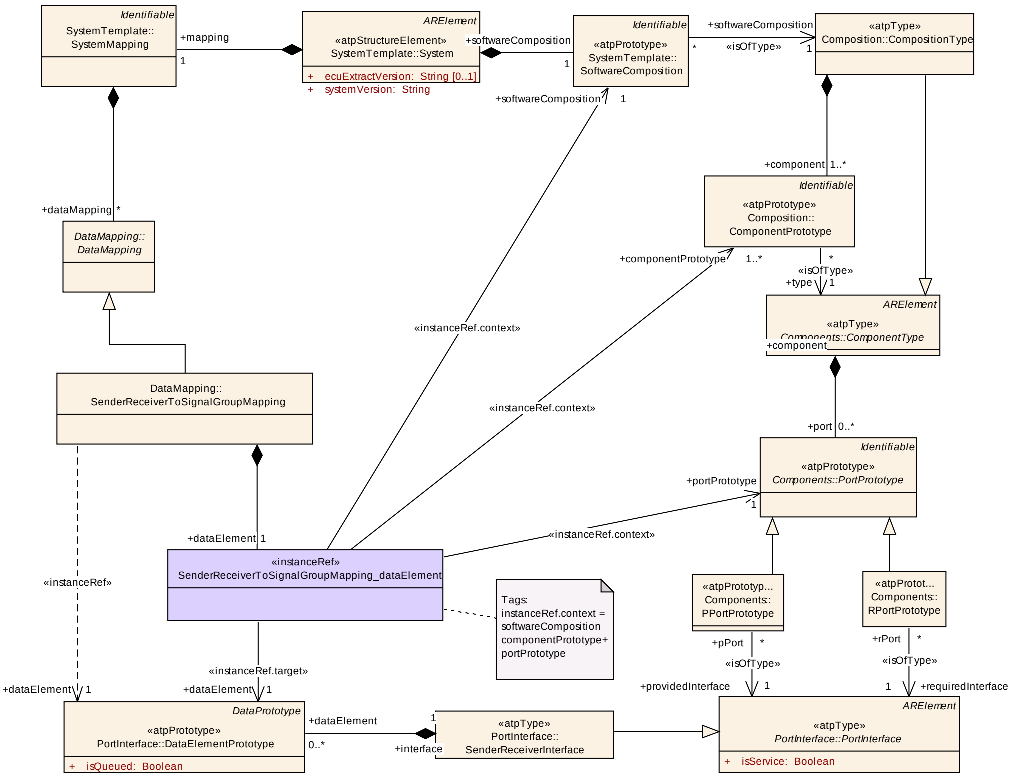

Figure B.2: composite Datatype Mapping

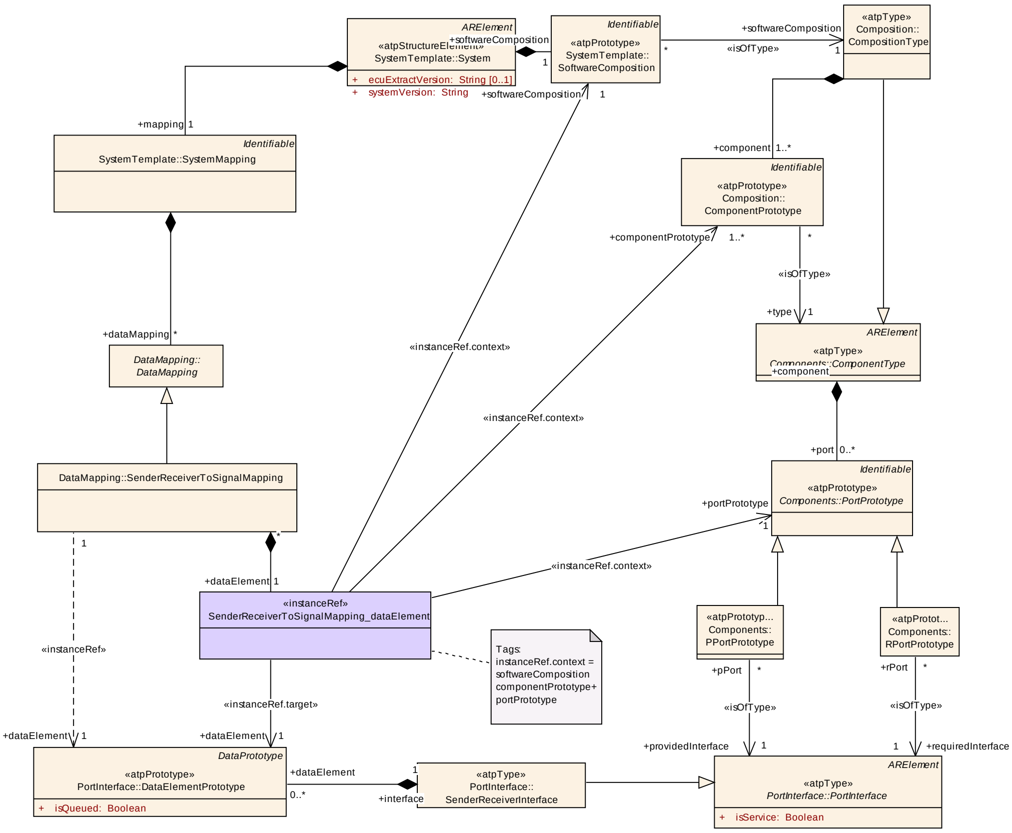

Figure B.3: primitive Datatype Mapping

B.2 Software Component Mapping

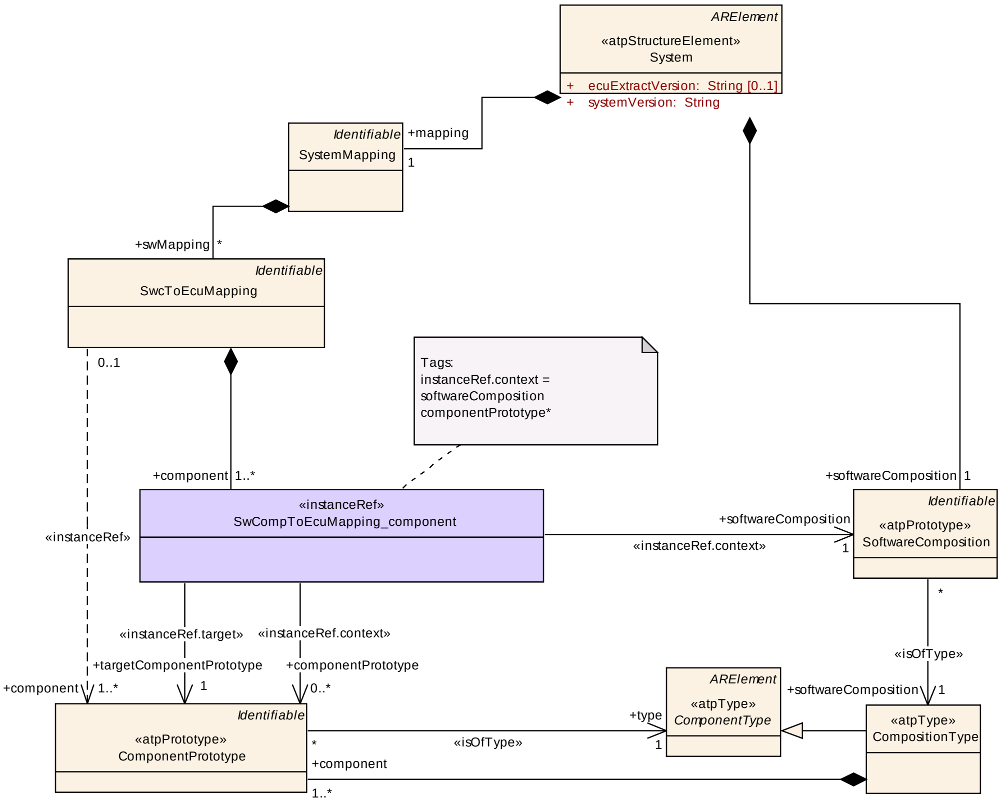

Figure B.4: SW Component To ECU Mapping

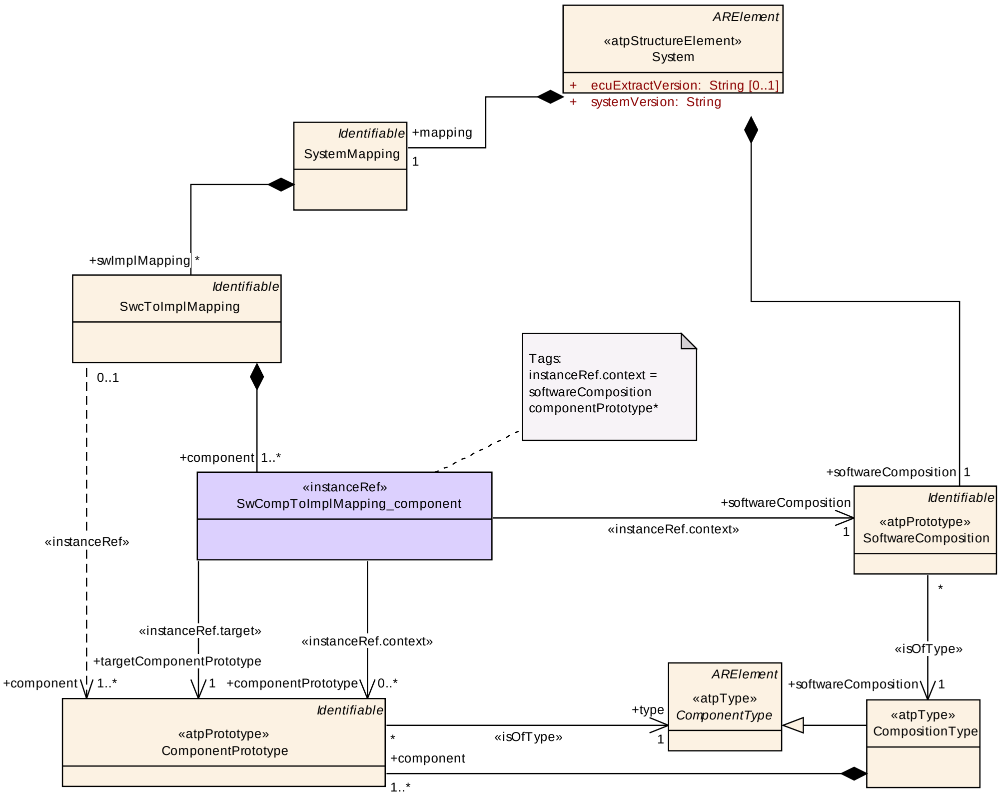

Figure B.5: SW Component To SWC Implementation Mapping

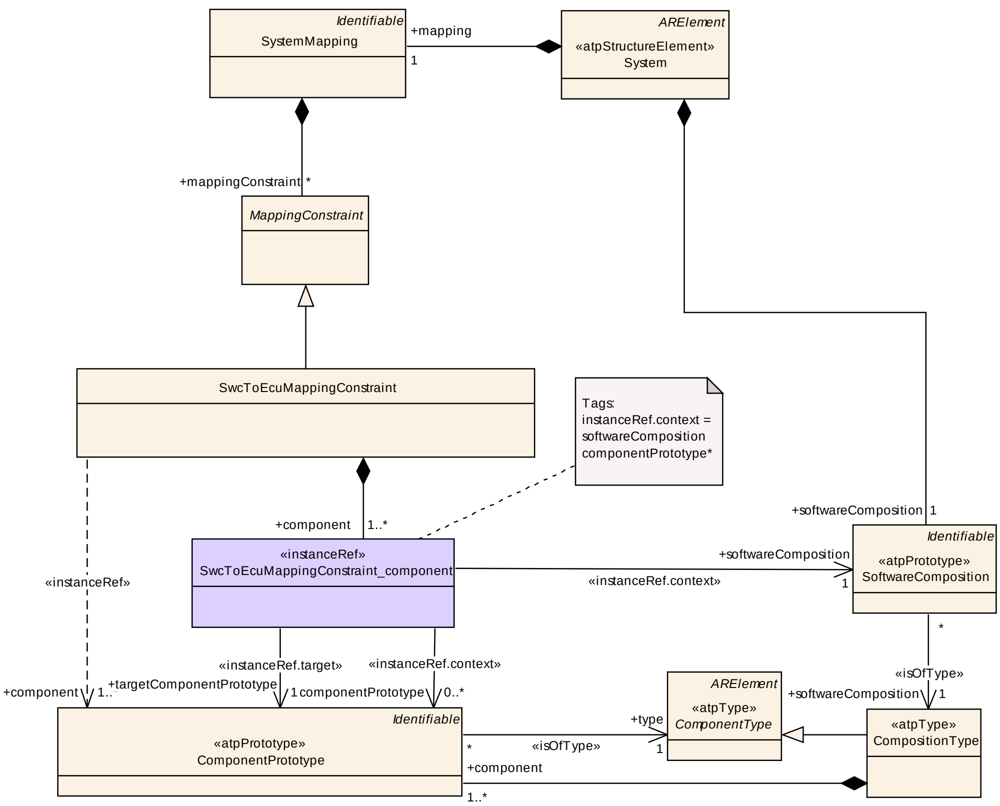

Figure B.6: SW Component To ECU Mapping Constraint

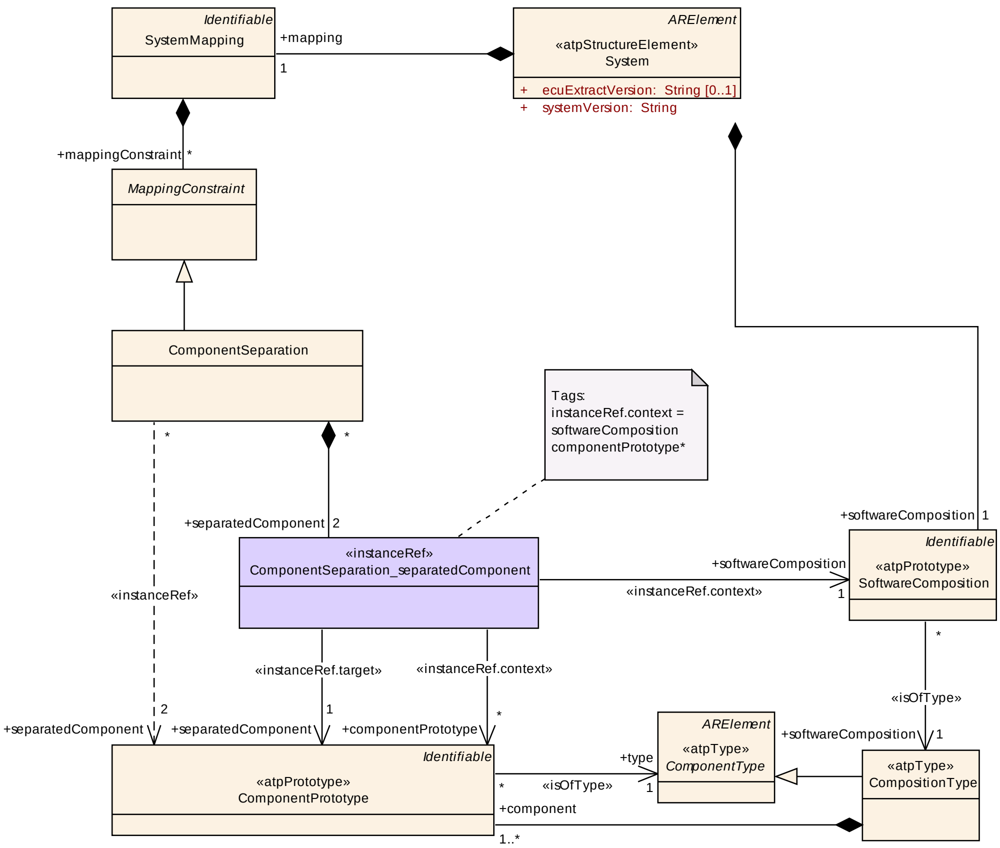

Figure B.7: SW Component Separation

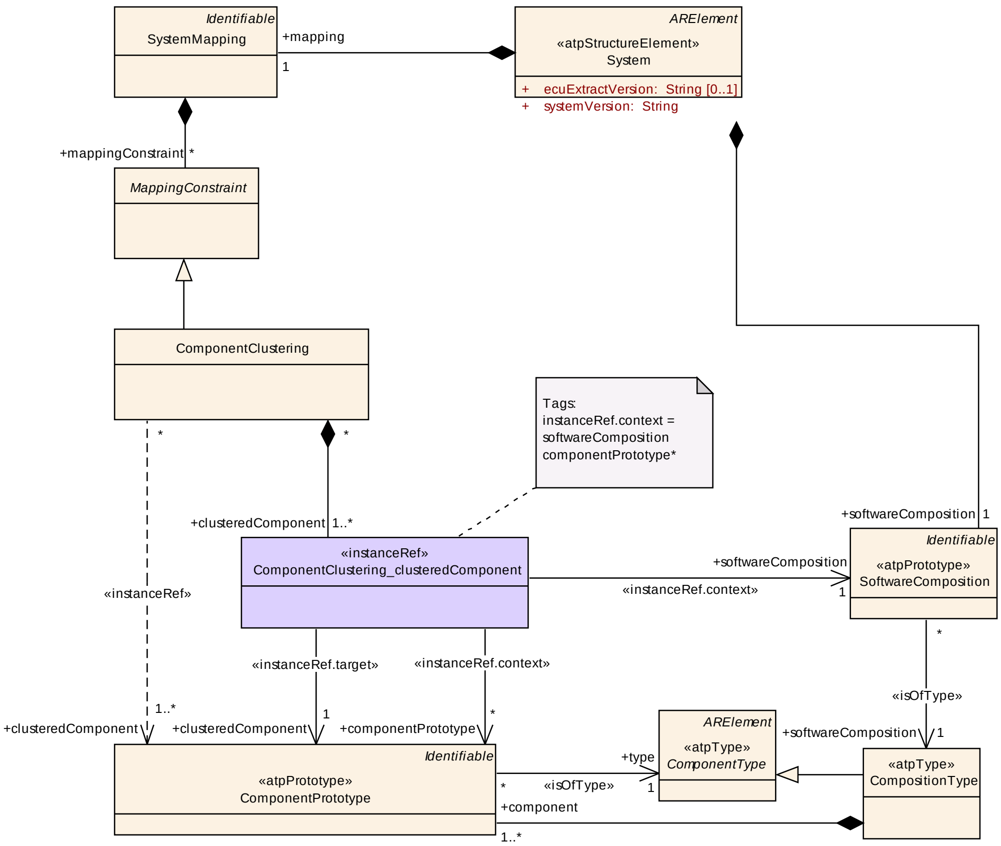

Figure B.8: SW Component Clustering

B.3 Signal Paths

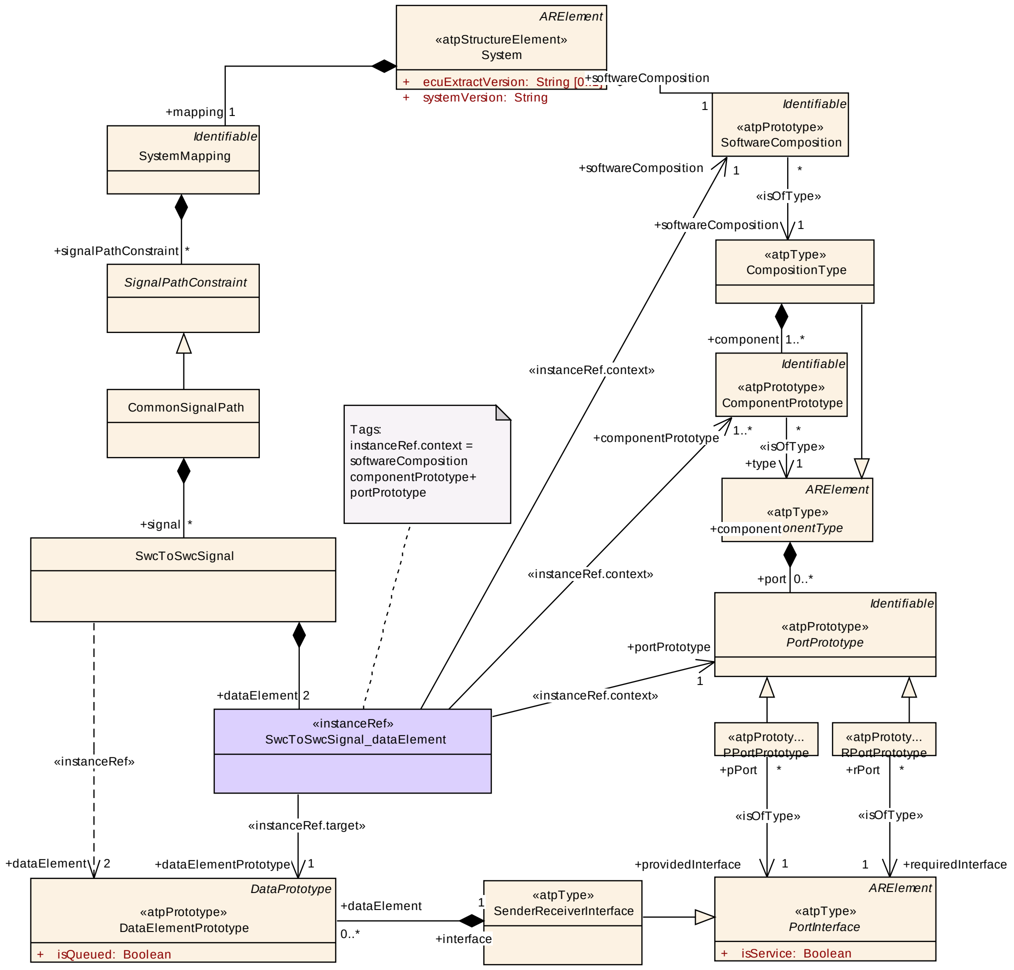

Figure B.9: SwcToSwcSignal

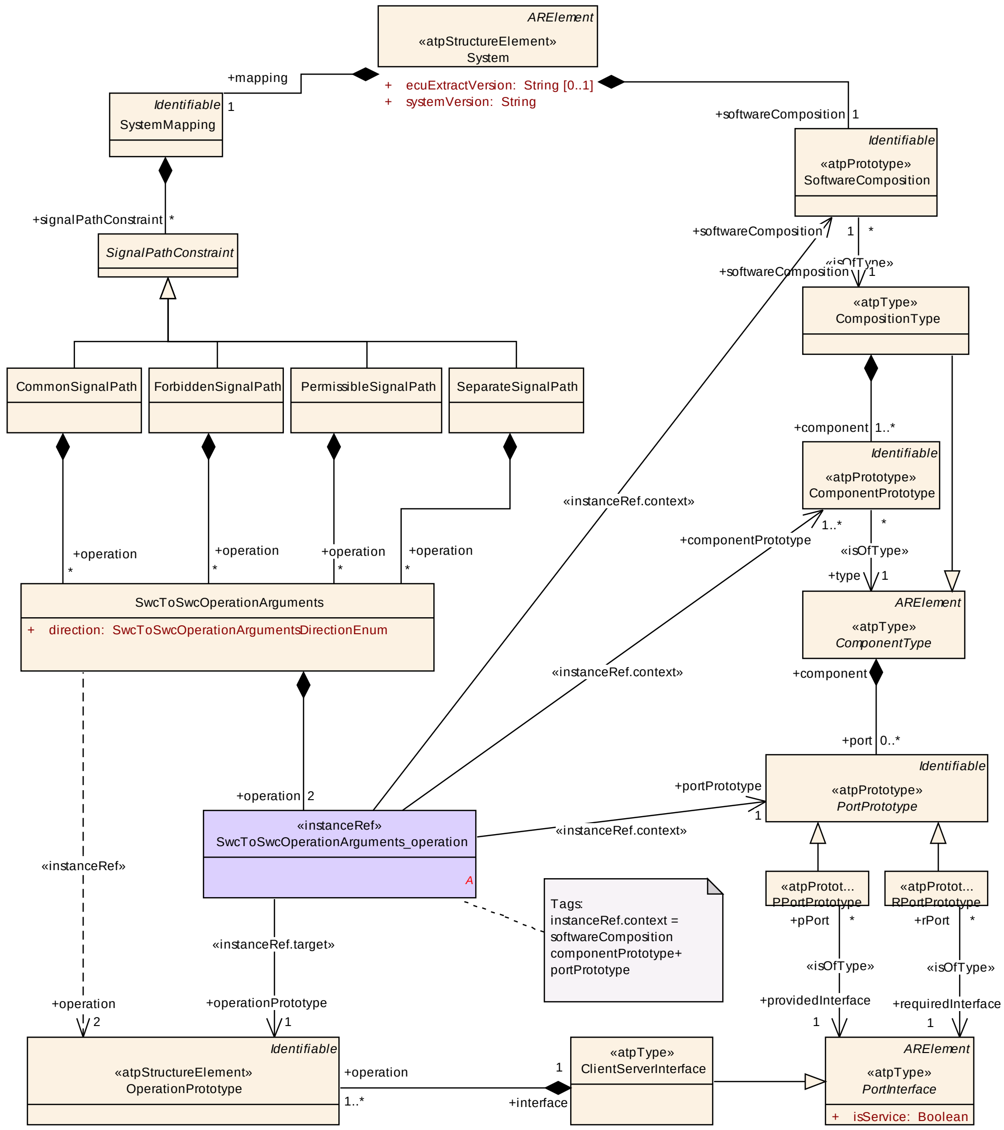

Figure B.10: SwcToSwcOperationArguments