Document Title	Specification of Module Efficient COM for Large Data
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	655
Document Classification	Standard

Document Status	Final
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	4.3.0

Document Change History

Date	Release	Changed by	Change Description
2016-11-30	4.3.0	AUTOSAR Release Management	• Introduced reliable TxConfirmation • Minor corrections
2015-07-31	4.2.2	AUTOSAR Release Management	• Fixed TriggerTransmit for dynamic length PDUs • Added PreCompile configuration class for all symbolicNameValue parameters
2014-10-31	4.2.1	AUTOSAR Release Management	• Initial Release

Disclaimer

Advice for users

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

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

Table of Contents

1 Introduction and functional overview

This specification specifies the functionality, API and the configuration of the AUTOSAR Basic Software module LdCom.

Within the AUTOSAR Layered Architecture the AUTOSAR LdCom module is placed between RTE and the PDU Router, see [1].

The AUTOSAR LdCom module provides an alternative Interaction Layer Mechanism. By focusing on spontaneous, non-cyclic communication without serializing, filtering and conversion an efficient implementation of the module without local buffers is achieved.

Main Features:

- Provision of signal oriented data interface for the RTE
- Provision of received signals to RTE
- Support of large and dynamic length data types
- Support of IF- and TP-based communication
- Provision of PDU oriented data interface towards PduR

2 Acronyms and abbreviations

Abbreviation / Acronym:	*Description:*
DEM	Diagnostic Event Manager
DET	Default Error Tracer

3 Related documentation

3.1 Input documents

[1] AUTOSAR Layered Software Architecture
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

[2] AUTOSAR General Requirements on Basic Software Modules
AUTOSAR_SRS_BSWGeneral.pdf

[3] AUTOSAR General Specification for Basic Software Modules
AUTOSAR_SWS_BSWGeneral.pdf

[4] Specification of RTE
AUTOSAR_SWS_RTE.pdf

[5] Specification of PDU Router
AUTOSAR_SWS_PDURouter.pdf

[6] Specification of System Template
AUTOSAR_RS_SystemTemplate.pdf

3.2 Related standards and norms

3.3 Related specification

AUTOSAR provides a General Specification on Basic Software (SWS BSW General) [3], which is also valid for this SWS.

Thus, the specification SWS BSW General [3] shall be considered as additional and required specification for this SWS.

4 Constraints and assumptions

4.1 Limitations

Efficient COM supports communication of linear opaque byte wise data in a very resource-saving way. It does so by skipping all functionality not required for event based non-cyclic communication.
Efficient COM does not apply any changes like for instance endianness conversion to the data it transports.
Prerequisites for usage of Efficient COM:

- PDU contains only 1 Signal and no ISignalGroup
- The Signal is of type byte array with either fixed or dynamic length
- Transmission mode is either triggered or triggered without repetition
- Transmission mode selection is not used
- No update bit is used
- No minimum delay time is used
- No timeout supervision is used
- No byte order conversion is used
- No Rx/Tx Filtering
- No Signal Invalidation

4.2 Applicability to car domains

No restrictions.

5 Dependencies to other modules

5.1 RTE

For RTE the AUTOSAR LdCom module is an additional mean to send and receive signals. In AUTOSAR, the RTE is the higher layer above the LdCom module. For further information, see [4].

5.2 PDU Router

The AUTOSAR LdCom module uses both sets of PDU Router's upper layer module APIs. That is the APIs for upper layer modules that use TP and the APIs for upper layer modules that do not use TP. This is necessary since the LdCom module forwards I-PDUs either unfragmented via simple L-PDUs or fragmented via TP.

The following summarizes the functionality of the AUTOSAR LdCom module needs from the underlying layer PDU Router:

- Indication of incoming I-PDUs
- Sending interface for outgoing I-PDUs including the confirmation if an I-PDU has been sent by the communication controller
- Trigger interface to enable the PDU router to cause a transmission from the AUTOSAR LdCom module
- Data forwarding for TP communication

5.3 Default Error Tracer (DET)

The DET provides services to store development errors (see Section 7.6).

5.4 File structure

[SWS_LDCOM_00001] ⌈The LdCom.c file shall include:
•
• PduR_LdCom.h
•
• Rte_Cbk.h
LdCom_Lcfg.h
• LdCom_PBcfg.h⌋ (SRS_BSW_00346, SRS_BSW_00381, SRS_BSW_00412)

[SWS_LDCOM_00002] ⌈The LdCom.h file shall include:
• LdCom_Cfg.h
• ComStack_Types.h
• LdCom_Types.h
•
•

] (SRS_BSW_00346, SRS_BSW_00381, SRS_BSW_00412)

[SWS_LDCOM_00050] ⌈The LdCom implementation shall include Det.h if LdComDevErrorDetect is enabled. ⌋ (SRS_BSW_00350)

[SWS_LDCOM_00051] ⌈The LdCom implementation shall additionally provide LdCom_Lcfg.c and LdComPBcfg.c. ⌋ (SRS_BSW_00344, SRS_BSW_00405, SRS_BSW_00345)

6 Requirements traceability

Requirement	Description	Satisfied by
SRS_BSW_00003	All software modules shall provide version and identification information	SWS_LDCOM_00024, SWS_LDCOM_00045
SRS_BSW_00101	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	SWS_LDCOM_00007, SWS_LDCOM_00008, SWS_LDCOM_00022
SRS_BSW_00305	Data types naming convention	SWS_LDCOM_00052
SRS_BSW_00336	Basic SW module shall be able to shutdown	SWS_LDCOM_00023
SRS_BSW_00337	Classification of development errors	SWS_LDCOM_00018
SRS_BSW_00344	BSW Modules shall support link-time configuration	SWS_LDCOM_00022, SWS_LDCOM_00051
SRS_BSW_00345	BSW Modules shall support pre-compile configuration	SWS_LDCOM_00051
SRS_BSW_00346	All AUTOSAR Basic Software Modules shall provide at least a basic set of module files	SWS_LDCOM_00001, SWS_LDCOM_00002
SRS_BSW_00350	All AUTOSAR Basic Software Modules shall allow the enabling/disabling of detection and reporting of development errors.	SWS_LDCOM_00050
SRS_BSW_00358	The return type of init() functions implemented by AUTOSAR Basic Software Modules shall be void	SWS_LDCOM_00022
SRS_BSW_00381	The pre-compile time parameters shall be placed into a separate configuration header file	SWS_LDCOM_00001, SWS_LDCOM_00002
SRS_BSW_00384	The Basic Software Module specifications shall specify at least in the description which other modules they require	SWS_LDCOM_00020, SWS_LDCOM_00035
SRS_BSW_00400	Parameter shall be selected from multiple sets of parameters after code has been loaded and started	SWS_LDCOM_00052
SRS_BSW_00404	BSW Modules shall	SWS_LDCOM_00022, SWS_LDCOM_00052

	support post-build configuration	
SRS_BSW_00405	BSW Modules shall support multiple configuration sets	SWS_LDCOM_00022, SWS_LDCOM_00051
SRS_BSW_00407	Each BSW module shall provide a function to read out the version information of a dedicated module implementation	SWS_LDCOM_00024, SWS_LDCOM_00045
SRS_BSW_00412	References to c-configuration parameters shall be placed into a separate h-file	SWS_LDCOM_00001, SWS_LDCOM_00002
SRS_BSW_00414	Init functions shall have a pointer to a configuration structure as single parameter	SWS_LDCOM_00022
SRS_BSW_00438	Configuration data shall be defined in a structure	SWS_LDCOM_00052
SRS_Com_02044	AUTOSAR COM and LargeDataCOM shall provide a transmit confirmation function	SWS_LDCOM_00046, SWS_LDCOM_00053
SRS_Com_02108	Support of Large Data COM	SWS_LDCOM_00005, SWS_LDCOM_00009, SWS_LDCOM_00035, SWS_LDCOM_00046
SRS_Com_02109	Large Data COM shall support Transport Protocol-like communication	SWS_LDCOM_00012, SWS_LDCOM_00013, SWS_LDCOM_00015, SWS_LDCOM_00016, SWS_LDCOM_00017, SWS_LDCOM_00027, SWS_LDCOM_00028, SWS_LDCOM_00029, SWS_LDCOM_00030, SWS_LDCOM_00031, SWS_LDCOM_00035, SWS_LDCOM_00036, SWS_LDCOM_00037, SWS_LDCOM_00038, SWS_LDCOM_00039, SWS_LDCOM_00040, SWS_LDCOM_00048, SWS_LDCOM_00049
SRS_Com_02110	Large Data COM shall support Interface-like communication	SWS_LDCOM_00010, SWS_LDCOM_00014, SWS_LDCOM_00026, SWS_LDCOM_00032, SWS_LDCOM_00035, SWS_LDCOM_00041, SWS_LDCOM_00046, SWS_LDCOM_00054, SWS_LDCOM_00055, SWS_LDCOM_00056
SRS_Com_02111	Large Data COM shall support Transmission Triggered by lower layer	SWS_LDCOM_00011, SWS_LDCOM_00033, SWS_LDCOM_00042, SWS_LDCOM_00047
SRS_Rte_00246	Support of Efficient COM for large data	SWS_LDCOM_00041

7 Functional specification

7.1 Initialization

[SWS_LDCOM_00007] ⌈The AUTOSAR LdCom module's initialization function LdCom_Init shall initialize all internal data. ⌋ (SRS_BSW_00101)

7.2 De-initialization

[SWS_LDCOM_00008] ⌈The AUTOSAR LdCom module shall provide the API function LdCom_DeInit for de-initialization of the AUTOSAR LdCom module. Inside this function call all de-initialization shall take place.⌋ (SRS_BSW_00101)

7.3 Overall

[SWS_LDCOM_00005] ⌈When called by PduR LdCom shall use the passed PDU Id as Handle Id (LdComHandleId ECUC_LdCom_00005), to derive the actual API from configuration and use it when passing the call towards RTE. ⌋ (SRS_Com_02108)

See Table 1: API to Parameter mapping for a mapping of API names used in this document to the ECUC Parameter containing the actual name configured for this API per signal. As per naming convention of the RTE <sn> is the name of the LdComIPdu.

API-Name	ECUC Parameter
Rte_LdComCbkCopyTxData_<sn>	LdComTxCopyTxData
Rte_LdComCbkTpTxConfirmation_<sn>	LdComTpTxConfirmation
Rte_LdComCbkRxIndication_<sn>	LdComRxIndication
Rte_LdComCbkStartOfReception_<sn>	LdComRxStartOfReception
Rte_LdComCbkCopyRxData_<sn>	LdComRxCopyRxData
Rte_LdComCbkTpRxIndication_<sn>	LdComTpRxIndication
Rte_LdComCbkTriggerTransmit_<sn>	LdComTxTriggerTransmit
Rte_LdComCbkTxConfirmation_<sn>	LdComTxConfirmation

Table 1: API to Parameter mapping

[SWS_LDCOM_00009] ⌈When called by RTE LdCom shall use the Signal Id as Handle Id (ECUC_LdCom_00005), passed as Parameter to derive the according PDU (LdComPduRef ECUC_LdCom_00010) when passing the call towards PduR. ⌋ (SRS_Com_02108)

7.4 Transmission

Transmission is initiated by RTE (LdCom_Transmit) or PduR (TriggerTransmit) but not by LdCom on its own.

7.4.1 IF

[SWS_LDCOM_00010] ⌈When LdCom_Transmit is invoked, LdCom shall invoke PduR_LdComTransmit by replacing the Signal Id by the according PDU Id. ⌋ (SRS_Com_02110)

[SWS_LDCOM_00011] ⌈When LdCom_TriggerTransmit is invoked, LdCom shall invoke Rte_LdComCbkTriggerTransmit_<sn> based on the PDU Id passed to of LdCom_TriggerTransmit as parameter.⌋
(SRS_Com_02111)

[SWS_LDCOM_00046] ⌈When LdCom_TxConfirmation is invoked, LdCom shall invoke Rte_LdComCbkTxConfirmation_<sn> based on the PDU Id passed to of LdCom_TxConfirmation as parameter ⌋
(SRS_Com_02044, SRS_Com_02108, SRS_Com_02110)

7.4.2 TP

[SWS_LDCOM_00012] ⌈LdCom shall pass invocations of LdCom_Transmit to PduR_LdComTransmit by replacing the Signal Id by the according PDU Id. ⌋ (SRS_Com_02109)

[SWS_LDCOM_00013] ⌈LdCom shall forward invocations of LdCom_CopyTxData and LdCom_TpTxConfirmation to RTE by invoking the corresponding Rte_LdComCbkCopyTxData_<sn> or Rte_LdComCbkTpTxConfirmation_<sn> based on the PDU Id passed to LdCom_CopyTxData and LdCom_TpTxConfirmation as parameter.⌋
(SRS_Com_02109)

7.5 Reception

7.5.1 IF

[SWS_LDCOM_00014] ⌈When LdCom_RxIndication is invoked, LdCom shall call the corresponding Rte_LdComCbkRxIndication_<sn> based on the PDU Id passed to of LdCom_RxIndication as parameter.⌋

(SRS_Com_02110)

7.5.2 TP

[SWS_LDCOM_00015] [When LdCom_StartOfReception is invoked by PduR, LdCom shall call the corresponding Rte_LdComCbkStartOfReception_<sn>> based on the PDU Id passed to of LdCom_StartOfReception as parameter.]
(SRS_Com_02109)

[SWS_LDCOM_00016] [When LdCom_CopyRxData is invoked by PduR, LdCom shall call Rte_LdComCbkCopyRxData_<sn> based on the PDU Id passed to of LdCom_CopyRxData as parameter.]
(SRS_Com_02109)

[SWS_LDCOM_00017] [When LdCom_TpRxIndication is invoked by PduR, LdCom shall call the corresponding Rte_LdComTpRxIndication_<sn> based on the PDU Id passed to of LdCom_TpRxIndication as parameter.]
(SRS_Com_02109)

7.6 Development Errors

[SWS_LDCOM_00018] [Development Error Types

Type of error	***Related error code***	***Value [hex]***
Error code if any other API service, except LdCom_GetVersionInfo is called before the AUTOSAR LdCom module was initialized with LdCom_Init or after a call to LdCom_Deinit	LDCOM_E_UNINIT	0x02
API service called with a NULL pointer. In case of this error, the API service shall return immediately without any further action, except for reporting this development error.	LDCOM_E_PARAM_POINTER	0x03
API service called with wrong PDU-ID	LDCOM_E_INVALID_PDU_SDU_ID	0x04
API service called with wrong Signal-ID	LDCOM_E_INVALID_SIGNAL_ID	0x05
Invalid configuration set selection	LDCOM_E_INIT_FAILED	0x06

] (SRS_BSW_00337)

7.7 Production Errors

No production errors are specified in LdCom.

7.8 Extended Production Errors

No extended production errors are specified LdCom.

7.9 Error notification

Defined in SWS BSW General.

7.10 Error classification

7.10.1 Runtime Errors

There are no runtime errors.

7.10.2 Transient Faults

There are no transient faults.

8 API specification

8.1 Imported types

In this chapter, all types included from the following modules are listed:

Imported Types
[SWS_LDCOM_00020] [

Module	*Imported Type*
ComStack_Types	BufReq_ReturnType
	PduIdType
	PduInfoType
	PduLengthType
	RetryInfoType
Std_Types	Std_ReturnType
	Std_VersionInfoType

] (SRS_BSW_00384)

8.2 Type definitions

8.2.1 LdCom_ConfigType

[SWS_LDCOM_00052] [

Name:	`LdCom_ConfigType`	
Type:	`Structure`	
Range:	`implementation specific`	The contents of the initialization data structure are implementation specific
Description:	This type contains the implementation-specific post build configuration structure.	

] (SRS_BSW_00400, SRS_BSW_00438, SRS_BSW_00404, SRS_BSW_00305)

8.3 Function definitions

This is a list of functions provided for upper layer modules.

8.3.1 LdCom_Init

[SWS_LDCOM_00022] [

Service name:	LdCom_Init	
Syntax:	`void LdCom_Init( const LdCom_ConfigType* config )`	
Service ID[hex]:	0x01	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	config	Pointer to the AUTOSAR LdCom module's configuration data.

Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	This service initializes internal and external interfaces and variables of the AUTOSAR LdCom module for the further processing.

] (SRS_BSW_00344, SRS_BSW_00404, SRS_BSW_00405, SRS_BSW_00101, SRS_BSW_00358, SRS_BSW_00414)

8.3.2 LdCom_DeInit

[SWS_LDCOM_00023] [

Service name:	LdCom_DeInit

Syntax:	`void LdCom_DeInit(`
	`  void`
	`)`

Service ID[hex]:	0x02
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	With a call to LdCom_DeInit the AUTOSAR LdCom module is put into an not initialized state.

] (SRS_BSW_00336)

8.3.3 LdCom_GetVersionInfo

[SWS_LDCOM_00024] [

Service name:	LdCom_GetVersionInfo	

Syntax:	`void LdCom_GetVersionInfo(`
	`  Std_VersionInfoType* versioninfo`
	`)`	

Service ID[hex]:	0x03	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	versioninfo	Pointer to where to store the version information of this module.
Return value:	None	
Description:	Returns the version information of this module.	

] (SRS_BSW_00407, SRS_BSW_00003)

[SWS_LDCOM_00045]
[
The API LdCom_GetVersionInfo shall be configured by LdComVersionInfoAPI.
](SRS_BSW_00407, SRS_BSW_00003)

8.3.4 LdCom_Transmit

[SWS_LDCOM_00026] [

Service name:	LdCom_Transmit	

Syntax:	`void LdCom_Transmit(`
	`  PduIdType Id,`
	`  const PduInfoType* PduInfoPtr`
	`)`	

Service ID[hex]:	0x05	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant for same HandleId, otherwise Reentrant	
Parameters (in):	Id	Id of the signal to be sent
	PduInfoPtr	Length and pointer to the buffer of the Signal.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Initiate a transmission of a signal using either IF- or TP-API set.	

] (SRS_Com_02110)

8.4 Call-back functions and notifications

This is a list of functions provided for other modules. The function prototypes of the callback functions shall be provided in the file LdCom_Cbk.h.

[SWS_LDCOM_00048]
[LdCom_CopyTxData, LdCom_TpTxConfirmation shall only be available if at least one LdComIPdu has LdComIPduDirection configured to LDCOM_SEND and LdComApiType configured to LDCOM_TP.] (SRS_Com_02109)

[SWS_LDCOM_00049]
[LdCom_StartOfReception, LdCom_CopyRxData, LdCom_TpRxIndication shall only be available if at least one LdComIPdu has LdComIPduDirection configured to LDCOM_RECEIVE and LdComApiType configured to LDCOM_TP.]
(SRS_Com_02109)

[SWS_LDCOM_00054]
[
LdCom_TxConfirmation shall only be available if at least one LdComIPdu has LdComIPduDirection configured to LDCOM_SEND and LdComApiType configured to LDCOM_IF.
] (SRS_Com_02110)

[SWS_LDCOM_00055]
[
LdCom_RxIndication shall only be available if at least one LdComIPdu has LdComIPduDirection configured to LDCOM_RECEIVE and LdComApiType configured to LDCOM_IF.

] (SRS_Com_02110)

8.4.1 LdCom_CopyTxData

[SWS_LDCOM_00027] [

Service name:	LdCom_CopyTxData	
Syntax:	`BufReq_ReturnType LdCom_CopyTxData( PduIdType id, const PduInfoType* info, RetryInfoType* retry, PduLengthType* availableDataPtr )`	
Service ID[hex]:	0x43	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	id	Identification of the transmitted I-PDU.
	info	Provides the destination buffer (SduDataPtr) and the number of bytes to be copied (SduLength). If not enough transmit data is available, no data is copied by the upper layer module and BUFREQ_E_BUSY is returned. The lower layer module may retry the call. An SduLength of 0 can be used to indicate state changes in the retry parameter or to query the current amount of available data in the upper layer module. In this case, the SduDataPtr may be a NULL_PTR.
	retry	This parameter is used to acknowledge transmitted data or to retransmit data after transmission problems. If the retry parameter is a NULL_PTR, it indicates that the transmit data can be removed from the buffer immediately after it has been copied. Otherwise, the retry parameter must point to a valid RetryInfoType element. If TpDataState indicates TP_CONFPENDING, the previously copied data must remain in the TP buffer to be available for error recovery. TP_DATACONF indicates that all data that has been copied before this call is confirmed and can be removed from the TP buffer. Data copied by this API call is excluded and will be confirmed later. TP_DATARETRY indicates that this API call shall copy previously copied data in order to recover from an error. In this case TxTpDataCnt specifies the offset in bytes from the current data copy position.
Parameters (inout):	None	
Parameters (out):	availableDataPtr	Indicates the remaining number of bytes that are available in the upper layer module's Tx buffer. availableDataPtr can be used by TP modules that support dynamic payload lengths (e.g. FrIsoTp) to determine the size of the following CFs.
Return value:	BufReq_ReturnType	BUFREQ_OK: Data has been copied to the transmit buffer completely as requested. BUFREQ_E_BUSY: Request could not be fulfilled, because the required amount of Tx data is not available. The lower layer module may retry this call later on. No data has been copied.

		BUFREQ_E_NOT_OK: Data has not been copied. Request failed.
Description:	This function is called to acquire the transmit data of an I-PDU segment (N-PDU). Each call to this function provides the next part of the I-PDU data unless retry->TpDataState is TP_DATARETRY. In this case the function restarts to copy the data beginning at the offset from the current position indicated by retry->TxTpDataCnt. The size of the remaining data is written to the position indicated by availableDataPtr.	

] (SRS_Com_02109)

8.4.2 LdCom_TpTxConfirmation

[SWS_LDCOM_00028] [

Service name:	LdCom_TpTxConfirmation	

Syntax:	`void LdCom_TpTxConfirmation(`
	`  PduIdType id,`
	`  Std_ReturnType result`
	`)`	

Service ID[hex]:	0x48	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	id	Identification of the transmitted I-PDU.
	result	Result of the transmission of the I-PDU.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	This function is called after the I-PDU has been transmitted on its network, the result indicates whether the transmission was successful or not.	

] (SRS_Com_02109)

8.4.3 LdCom_StartOfReception

[SWS_LDCOM_00029] [

Service name:	LdCom_StartOfReception	

Syntax:	`BufReq_ReturnType LdCom_StartOfReception(`
	`  PduIdType id,`
	`  const PduInfoType* info,`
	`  PduLengthType TpSduLength,`
	`  PduLengthType* bufferSizePtr`
	`)`	

Service ID[hex]:	0x46	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	id	Identification of the I-PDU.
	info	Pointer to a PduInfoType structure containing the payload data (without protocol information) and payload length of the first frame or single frame of a transport protocol I-PDU reception, and the MetaData related to this PDU. If neither first/single frame data nor MetaData are available, this

		parameter is set to NULL_PTR.
	TpSduLength	Total length of the N-SDU to be received.
Parameters (inout):	None	
Parameters (out):	bufferSizePtr	Available receive buffer in the receiving module. This parameter will be used to compute the Block Size (BS) in the transport protocol module.
Return value:	BufReq_ReturnType	BUFREQ_OK: Connection has been accepted. bufferSizePtr indicates the available receive buffer; reception is continued. If no buffer of the requested size is available, a receive buffer size of 0 shall be indicated by bufferSizePtr. BUFREQ_E_NOT_OK: Connection has been rejected; reception is aborted. bufferSizePtr remains unchanged. BUFREQ_E_OVFL: No buffer of the required length can be provided; reception is aborted. bufferSizePtr remains unchanged.
Description:	This function is called at the start of receiving an N-SDU. The N-SDU might be fragmented into multiple N-PDUs (FF with one or more following CFs) or might consist of a single N-PDU (SF). The service shall provide the currently available maximum buffer size when invoked with TpSduLength equal to 0.	

] (SRS_Com_02109)

8.4.4 LdCom_CopyRxData

[SWS_LDCOM_00030] [

Service name:	LdCom_CopyRxData	

Syntax:	`BufReq_ReturnType LdCom_CopyRxData(`
	`  PduIdType id,`
	`  const PduInfoType* info,`
	`  PduLengthType* bufferSizePtr`
	`)`	

Service ID[hex]:	0x44	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	id	Identification of the received I-PDU.
	info	Provides the source buffer (SduDataPtr) and the number of bytes to be copied (SduLength). An SduLength of 0 can be used to query the current amount of available buffer in the upper layer module. In this case, the SduDataPtr may be a NULL_PTR.
Parameters (inout):	None	
Parameters (out):	bufferSizePtr	Available receive buffer after data has been copied.
Return value:	BufReq_ReturnType	BUFREQ_OK: Data copied successfully BUFREQ_E_NOT_OK: Data was not copied because an error occurred.
Description:	This function is called to provide the received data of an I-PDU segment (N-PDU) to the upper layer. Each call to this function provides the next part of the I-PDU data. The size of the remaining data is written to the position indicated by bufferSizePtr.	

] (SRS_Com_02109)

8.4.5 LdCom_TpRxIndication

[SWS_LDCOM_00031] [

Service name:	LdCom_TpRxIndication	

Syntax:	`void LdCom_TpRxIndication(`
	`  PduIdType id,`
	`  Std_ReturnType result`
	`)`	

Service ID[hex]:	0x45	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	id	Identification of the received I-PDU.
	result	Result of the reception.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Called after an I-PDU has been received via the TP API, the result indicates whether the transmission was successful or not.	

] (SRS_Com_02109)

8.4.6 LdCom_RxIndication

[SWS_LDCOM_00032] [

Service name:	LdCom_RxIndication	

Syntax:	`void LdCom_RxIndication(`
	`  PduIdType RxPduId,`
	`  const PduInfoType* PduInfoPtr`
	`)`	

Service ID[hex]:	0x42	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in):	RxPduId	ID of the received PDU.
	PduInfoPtr	Contains the length (SduLength) of the received PDU, a pointer to a buffer (SduDataPtr) containing the PDU, and the MetaData related to this PDU.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Indication of a received PDU from a lower layer communication interface module.	

] (SRS_Com_02110)

8.4.7 LdCom_TxConfirmation

[SWS_LDCOM_00056] [

Service name:	LdCom_TxConfirmation

Syntax:	`void LdCom_TxConfirmation(`
	`  PduIdType TxPduId,`
	`  Std_ReturnType result`
	`)`	

Service ID[hex]:	0x40	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in):	TxPduId	ID of the PDU that has been transmitted.
	result	E_OK: The PDU was transmitted. E_NOT_OK: Transmission of the PDU failed.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.	

] (SRS_Com_02110)

8.4.8 LdCom_TriggerTransmit

[SWS_LDCOM_00033] [

Service name:	LdCom_TriggerTransmit	

Syntax:	`Std_ReturnType LdCom_TriggerTransmit(`
	`  PduIdType TxPduId,`
	`  PduInfoType* PduInfoPtr`
	`)`	

Service ID[hex]:	0x41	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in):	TxPduId	ID of the SDU that is requested to be transmitted.
Parameters (inout):	PduInfoPtr	Contains a pointer to a buffer (SduDataPtr) to where the SDU data shall be copied, and the available buffer size in SduLengh. On return, the service will indicate the length of the copied SDU data in SduLength.
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: SDU has been copied and SduLength indicates the number of copied bytes. E_NOT_OK: No SDU data has been copied. PduInfoPtr must not be used since it may contain a NULL pointer or point to invalid data.
Description:	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.	

] (SRS_Com_02111)

[SWS_LDCOM_00047]
[LdCom_TriggerTransmit shall only be available if at least one LdComIPdu has LdComTxTriggerTransmit configured.] (SRS_Com_02111)

8.5 Scheduled functions

None.

8.6 Expected Interfaces

In this chapter all external interfaces required from other modules are listed.

8.6.1 Mandatory Interfaces

8.6.2 Optional Interfaces

This chapter defines all external interfaces which are required to fulfill an optional functionality of the module.

[SWS_LDCOM_00035] [

API function	*Description*
Det_ReportError	Service to report development errors.
PduR_LdComTransmit	Requests transmission of a PDU.

] (SRS_BSW_00384, SRS_Com_02108, SRS_Com_02109, SRS_Com_02110)

8.6.3 Configurable interfaces

In this chapter all interfaces are listed where the target function could be configured. The target function is usually a call-back function. The names of these kind of interfaces are not fixed because they are configurable.

The following Callbacks can be configured for each signal .

See Table 1: API to Parameter mapping for the configuration of the actual API names.

8.6.3.1 Rte_LdComCbkCopyTxData_<sn>

[SWS_LDCOM_00036] [

Service name:	Rte_LdComCbkCopyTxData_<sn>	
Syntax:	`BufReq_ReturnType Rte_LdComCbkCopyTxData_<sn>( const PduInfoType* info, RetryInfoType* retry, PduLengthType* availableDataPtr )`	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant for same sn, otherwise Reentrant	
Parameters (in):	info	Provides the destination buffer (SduDataPtr) and the number of bytes to be copied (SduLength). If not enough transmit data is available, no data is copied by

		the upper layer module and BUFREQ_E_BUSY is returned. The lower layer module may retry the call. An SduLength of 0 can be used to indicate state changes in the retry parameter or to query the current amount of available data in the upper layer module. In this case, the SduDataPtr may be a NULL_PTR.
	retry	Will not be handled by LdCom and its upper layer.
Parameters (inout):	None	
Parameters (out):	availableDataPtr	Indicates the remaining number of bytes that are available in the upper layer module's Tx buffer. availableDataPtr can be used by TP modules that support dynamic payload lengths (e.g. FrIsoTp) to determine the size of the following CFs.
Return value:	BufReq_ReturnType	BUFREQ_OK: Data has been copied to the transmit buffer completely as requested. BUFREQ_E_BUSY: Request could not be fulfilled, because the required amount of Tx data is not available. The lower layer module may retry this call later on. No data has been copied. BUFREQ_E_NOT_OK: Data has not been copied. Request failed.
Description:	This function is called to acquire the transmit data of an I-PDU segment (N-PDU). Each call to this function provides the next part of the I-PDU data unless retry->TpDataState is TP_DATARETRY. In this case the function restarts to copy the data beginning at the offset from the current position indicated by retry->TxTpDataCnt. The size of the remaining data is written to the position indicated by availableDataPtr	

] (SRS_Com_02109)

8.6.3.2 Rte_LdComCbkTpTxConfirmation_<sn>

[SWS_LDCOM_00037] [

Service name:	Rte_LdComCbkTpTxConfirmation_<sn>	

Syntax:	`void Rte_LdComCbkTpTxConfirmation_<sn>(`
	`  Std_ReturnType result`
	`)`	

Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant for same sn, otherwise Reentrant	
Parameters (in):	result	E_OK - transmission successful E_NOT_OK - transmission not successful
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	This function is called after a Signal has been transmitted via the TP-API on its network.	

] (SRS_Com_02109)

8.6.3.3 Rte_LdComCbkStartOfReception_<sn>

[SWS_LDCOM_00038] [

Service name:	Rte_LdComCbkStartOfReception_<sn>

Syntax:	`BufReq_ReturnType Rte_LdComCbkStartOfReception_<sn>(`
	`  const PduInfoType* info,`
	`  PduLengthType TpSduLength,`
	`  PduLengthType* bufferSizePtr`
	`)`

Sync/Async:	Synchronous

Reentrancy:	Non Reentrant for same sn, otherwise Reentrant	
Parameters (in):	info	Pointer to a PduInfoType structure containing the payload data (without protocol information) and payload length of the first frame or single frame of a transport protocol I-PDU reception, and the MetaData related to this PDU. If neither first/single frame data nor MetaData are available, this parameter is set to NULL_PTR.
	TpSduLength	Total length of the N-SDU to be received.
Parameters (inout):	None	
Parameters (out):	bufferSizePtr	Available receive buffer in the receiving module. This parameter will be used to compute the Block Size (BS) in the transport protocol module.
Return value:	BufReq_ReturnType	BUFREQ_OK: Connection has been accepted. bufferSizePtr indicates the available receive buffer; reception is continued. If no buffer of the requested size is available, a receive buffer size of 0 shall be indicated by bufferSizePtr. BUFREQ_E_NOT_OK: Connection has been rejected; reception is aborted. bufferSizePtr remains unchanged. BUFREQ_E_OVFL: No buffer of the required length can be provided; reception is aborted. bufferSizePtr remains unchanged.
Description:	This function is called at the start of receiving an N-SDU. The N-SDU might be fragmented into multiple N-PDUs (FF with one or more following CFs) or might consist of a single N-PDU (SF). The service shall provide the currently available maximum buffer size when invoked with TpSduLength equal to 0.	

] (SRS_Com_02109)

8.6.3.4 Rte_LdComCbkCopyRxData_<sn>

[SWS_LDCOM_00039] [

Service name:	Rte_LdComCbkCopyRxData_<sn>	

Syntax:	`BufReq_ReturnType Rte_LdComCbkCopyRxData_<sn>(`
	`  const PduInfoType* info,`
	`  PduLengthType* bufferSizePtr`
	`)`	

Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant for same sn, otherwise Reentrant	
Parameters (in):	info	Provides the source buffer (SduDataPtr) and the number of bytes to be copied (SduLength). An SduLength of 0 can be used to query the current amount of available buffer in the upper layer module. In this case, the SduDataPtr may be a NULL_PTR.
Parameters (inout):	None	
Parameters (out):	bufferSizePtr	Available receive buffer after data has been copied.
Return value:	BufReq_ReturnType	BUFREQ_OK: Data copied successfully BUFREQ_E_NOT_OK: Data was not copied because an error occurred.
Description:	This function is called to provide the received data of an I-PDU segment (N-PDU) to the upper layer. Each call to this function provides the next part of the I-PDU data. The size of the remaining data is written to the position indicated by bufferSizePtr.	

] (SRS_Com_02109)

8.6.3.5 Rte_LdComCbkTpRxIndication_<sn>

[SWS_LDCOM_00040] [

Service name:	Rte_LdComCbkTpRxIndication_<sn>	

Syntax:	`void Rte_LdComCbkTpRxIndication_<sn>(`
	`  Std_ReturnType result`
	`)`	

Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant for same sn, otherwise Reentrant	
Parameters (in):	result	Result of the reception.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Called after an I-PDU has been received via the TP API, the result indicates whether the transmission was successful or not.	

] (SRS_Com_02109)

8.6.3.6 Rte_LdComCbkRxIndication_<sn>

[SWS_LDCOM_00041] [

Service name:	Rte_LdComCbkRxIndication_<sn>	

Syntax:	`void Rte_LdComCbkRxIndication_<sn>(`
	`  const PduInfoType* PduInfoPtr`
	`)`	

Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant for same sn, otherwise Reentrant	
Parameters (in):	PduInfoPtr	Contains the length (SduLength) of the received PDU, a pointer to a buffer (SduDataPtr) containing the PDU, and the MetaData related to this PDU.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Indication of a received PDU from a lower layer communication interface module.	

] (SRS_Rte_00246, SRS_Com_02110)

8.6.3.7 Rte_LdComCbkTriggerTransmit_<sn>

[SWS_LDCOM_00042] [

Service name:	Rte_LdComCbkTriggerTransmit_<sn>	

Syntax:	`Std_ReturnType Rte_LdComCbkTriggerTransmit_<sn>(`
	`  PduInfoType* PduInfoPtr`
	`)`	

Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant for same sn, otherwise Reentrant	
Parameters (in):	None	
Parameters (inout):	PduInfoPtr	Contains a pointer to a buffer (SduDataPtr) to where the SDU data shall be copied, and the available buffer size in SduLengh. On return, the service will indicate the length of the copied SDU data in SduLength.
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: SDU has been copied and SduLength indicates the number of copied bytes.

		E_NOT_OK: No SDU data has been copied. PduInfoPtr must not be used since it may contain a NULL pointer or point to invalid data.
Description:	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.	

] (SRS_Com_02111)

8.6.3.8 Rte_LdComCbkTxConfirmation_<sn>

[SWS_LDCOM_00053] [

Service name:	Rte_LdComCbkTxConfirmation_<sn>	

Syntax:	`void Rte_LdComCbkTxConfirmation_<sn>(`
	`  Std_ReturnType result`
	`)`	

Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant for same sn, otherwise Reentrant	
Parameters (in):	result	E_OK: The PDU was transmitted. E_NOT_OK: Transmission of the PDU failed.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.	

] (SRS_Com_02044)

8.7 Service Interfaces

None.

9 Sequence diagrams

This chapter contains sequence charts showing the involvement of LdCom into interactions between RTE and PduR.

9.1 Transmission

9.1.1 TP-API

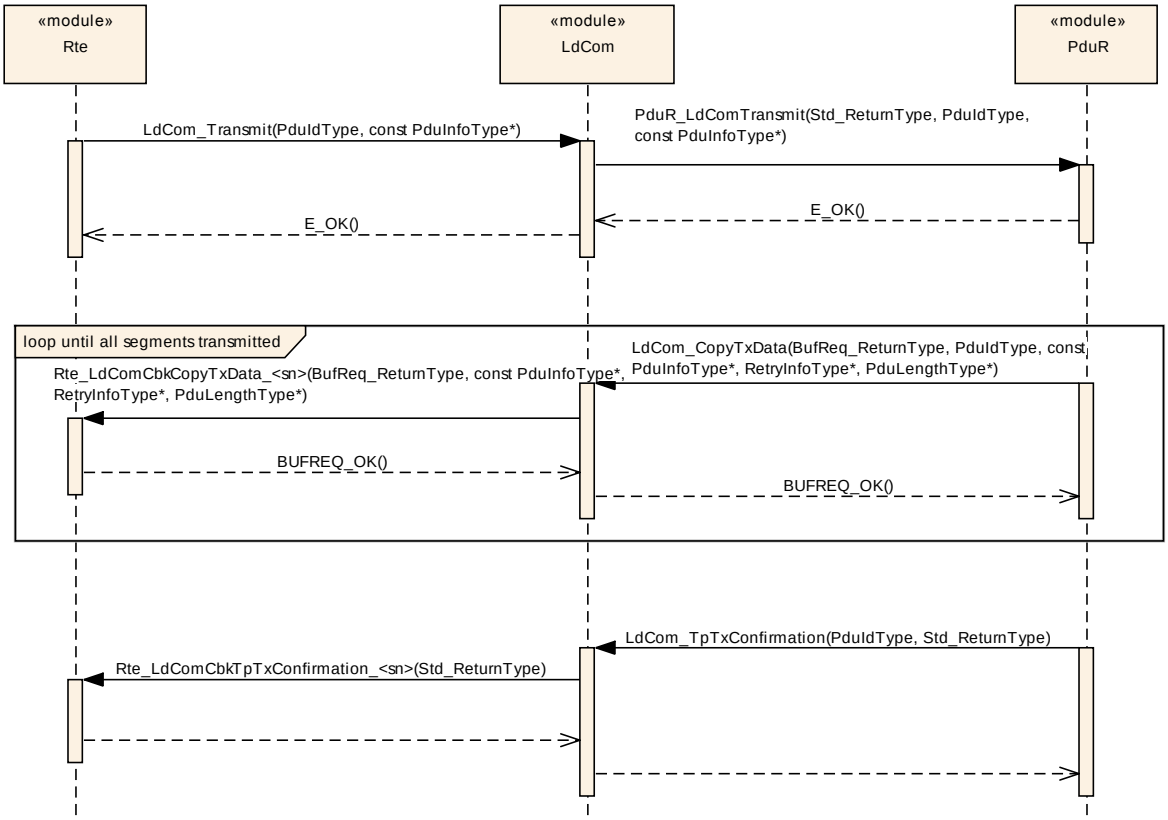

Figure 1 - Transmission via TP-API

9.1.2 IF-API

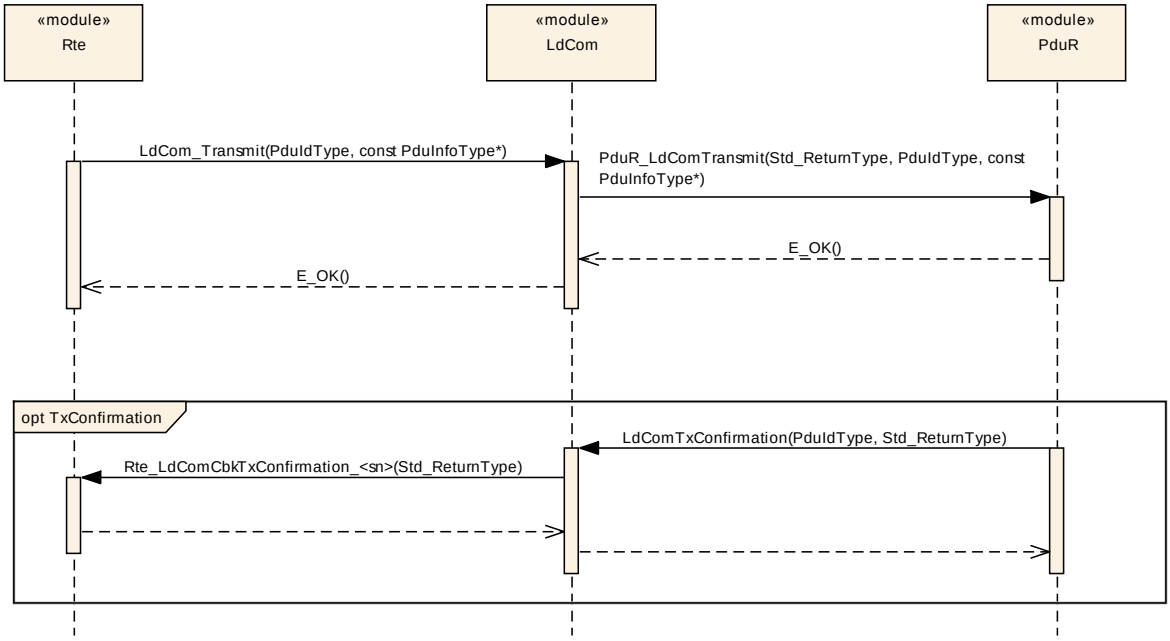

Figure 2 - Transmission via IF-API

9.1.3 TriggerTransmit

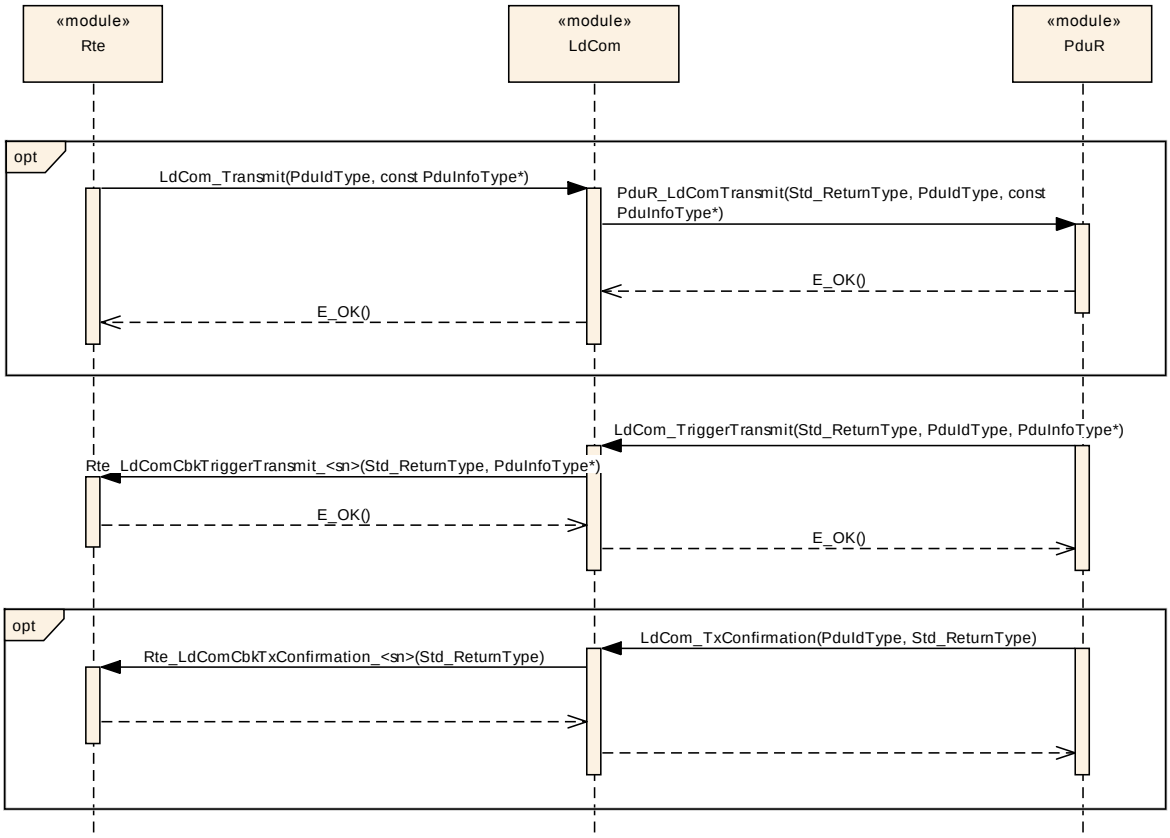

Figure 3 – TriggerTransmit

9.2 Reception

9.2.1 TP-API

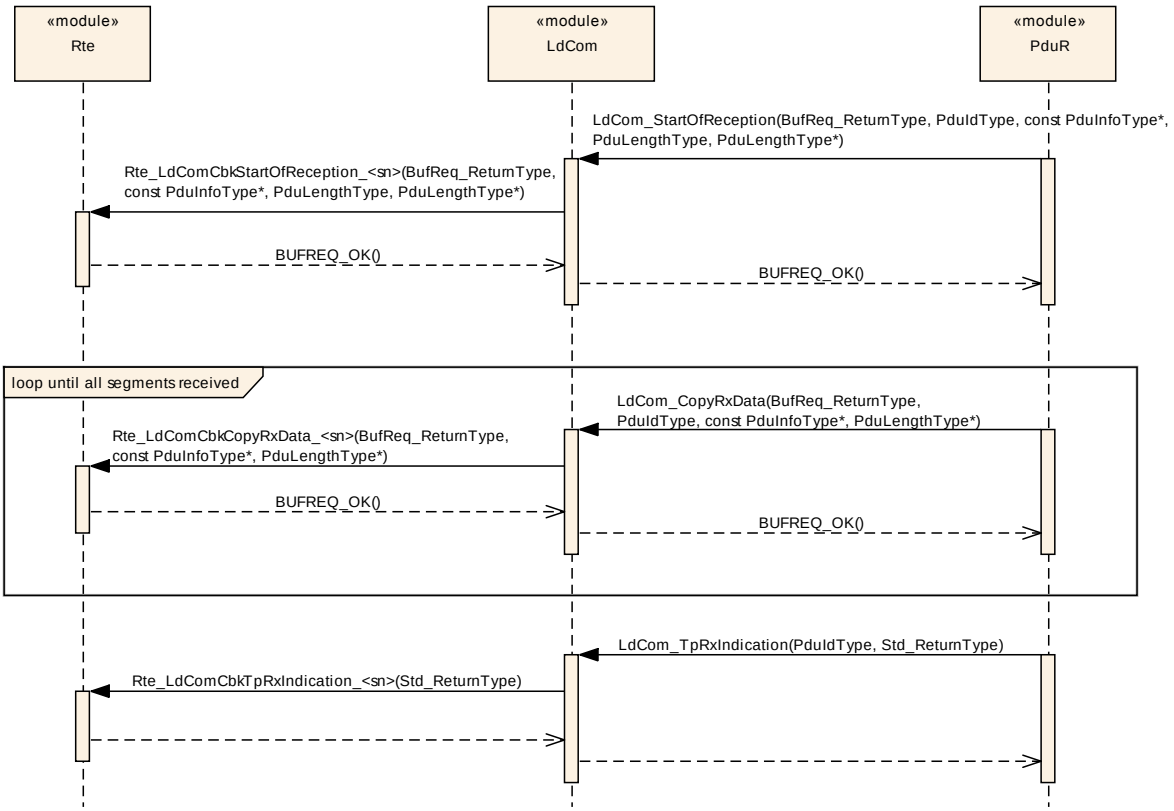

Figure 4 - Reception via TP-API

9.2.2 IF-API

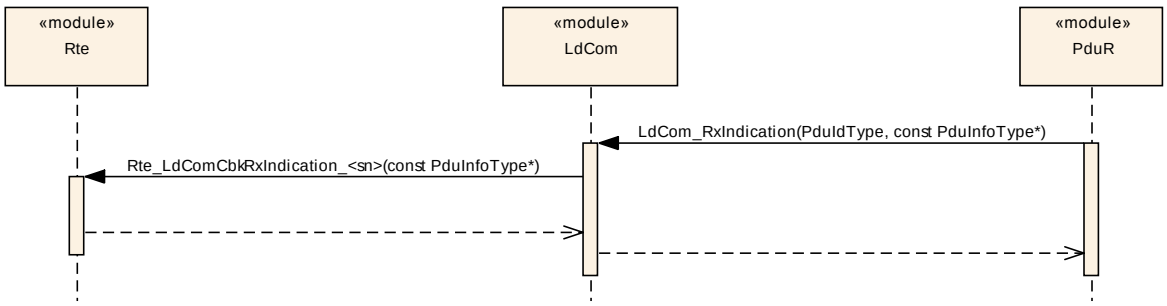

Figure 5 - Reception via IF-API

10 Configuration specification

Chapter 10.1 specifies the structure (containers) and the parameters of LdCom.

Chapter 10.2 specifies additionally published information of LdCom.

10.1 Containers and configuration parameters

The following chapters summarize all configuration parameters. The detailed meanings of the parameters describe Chapters 0 and Chapter 8.

10.1.1 LdCom

SWS Item	**ECUC_LdCom_00001 :**
Module Name	*LdCom*
Module Description	Configuration of the AUTOSAR LdCom module.
Post-Build Variant Support	true
Supported Config Variants	VARIANT-LINK-TIME, VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
LdComConfig	1	This container contains the configuration parameters and sub containers of the AUTOSAR LdCom module.
LdComGeneral	1	Contains the general configuration parameters of the LdCom module.

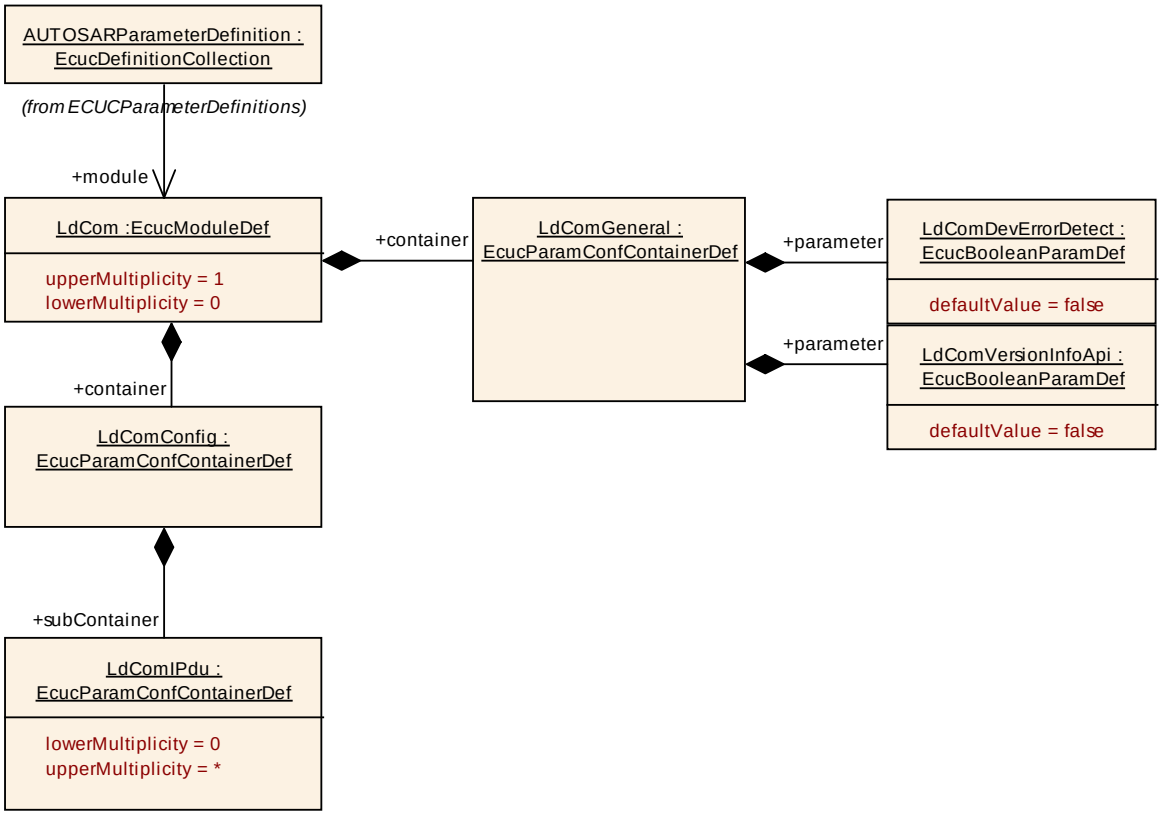

Figure 6 : Configuration LdCom

10.1.2 LdComConfig

SWS Item	**ECUC_LdCom_00003 :**
Container Name	LdComConfig
Description	This container contains the configuration parameters and sub containers of the AUTOSAR LdCom module.
Configuration Parameters	

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
LdComIPdu	0..*	Contains the configuration parameters of the IPdu inside LdCom.

10.1.3 LdComGeneral

SWS Item	**ECUC_LdCom_00004 :**
Container Name	LdComGeneral
Description	Contains the general configuration parameters of the LdCom module.
Configuration Parameters	

SWS Item	**ECUC_LdCom_00020 :**
Name	LdComDevErrorDetect
Description	Switches the development error detection and notification on or off.

	• true: detection and notification is enabled. • false: detection and notification is disabled.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_LdCom_00012 :**		
Name	LdComVersionInfoApi		
Description	Activate/Deactivate the version information API (LdCom_GetVersionInfo). • True: version information API activated • False: version information API deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

10.1.4 LdComIPdu

SWS Item	**ECUC_LdCom_00006 :**		
Container Name	LdComIPdu		
Description	Contains the configuration parameters of the IPdu inside LdCom.		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Configuration Parameters			

SWS Item	**ECUC_LdCom_00002 :**	
Name	LdComApiType	
Description	Defines if this I-PDU is a normal I-PDU that shall be sent unfragmented or if this is a large I-PDU that shall be sent via the Transport Protocol of the underlying bus. This setting is used by RTE to invoke the proper API.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	LDCOM_IF	sent or received via interface API.
	LDCOM_TP	sent or received via transport protocol

		API.	
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_LdCom_00005 :**		
Name	LdComHandleId		
Description	This is the ID used by RTE to invoke LdCom. A corresponding shortName is created, which is used for the invocations of the RTE. The same ID is used for invocations by PduR.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_LdCom_00007 :**		
Name	LdComIPduDirection		
Description	The direction defines if this IPdu, and therefore the contributing signal, shall be sent or received.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	LDCOM_RECEIVE	--	
	LDCOM_SEND	--	
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_LdCom_00013 :**
Name	LdComRxCopyRxData
Description	Only on receiver side: Name of Rte_LdComCbkCopyRxData callback function to be called.
Multiplicity	0..1
Type	EcucFunctionNameDef
Default value	--
maxLength	--
minLength	--
regularExpression	--
Post-Build Variant Multiplicity	false
Post-Build Variant Value	false

Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_LdCom_00014 :**		
Name	LdComRxIndication		
Description	Only on receiver side: Name of Rte_LdComCbkRxIndication callback function to be called.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_LdCom_00015 :**		
Name	LdComRxStartOfReception		
Description	Only on receiver side: Name of Rte_LdComCbkStartOfReception callback function to be called.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_LdCom_00016 :**		
Name	LdComTpRxIndication		
Description	Only on receiver side: Name of Rte_LdComCbkTpRxIndication callback function to be called.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_LdCom_00017 :**		
Name	LdComTpTxConfirmation		
Description	Only on sender side: Name of Rte_LdComCbkTpTxConfirmation callback function to be called.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_LdCom_00021 :**
Name	LdComTxConfirmation
Description	Only on sender side: Name of Rte_LdComCbkTxConfirmation callback function to be called.
Multiplicity	0..1
Type	EcucFunctionNameDef
Default value	--
maxLength	--
minLength	--
regularExpression	--

Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_LdCom_00018 :**		
Name	LdComTxCopyTxData		
Description	Only on sender side: Name of Rte_LdComCbkCopyTxData callback function to be called.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_LdCom_00019 :**		
Name	LdComTxTriggerTransmit		
Description	Only on sender side: Name of Rte_LdComCbkTriggerTransmit callback function to be called. If defined TriggerTransmit has to be supported for this signal.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-

			BUILD
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_LdCom_00010 :**		
Name	LdComPduRef		
Description	Reference to the global Pdu.		
Multiplicity	1		
Type	Reference to [Pdu]		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_LdCom_00011 :**		
Name	LdComSystemTemplateSignalRef		
Description	Reference to the ISignalToIPduMapping that contains a reference to the ISignal (System Template).		
Multiplicity	0..1		
Type	Foreign reference to [I-SIGNAL-TO-I-PDU-MAPPING]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

No Included Containers

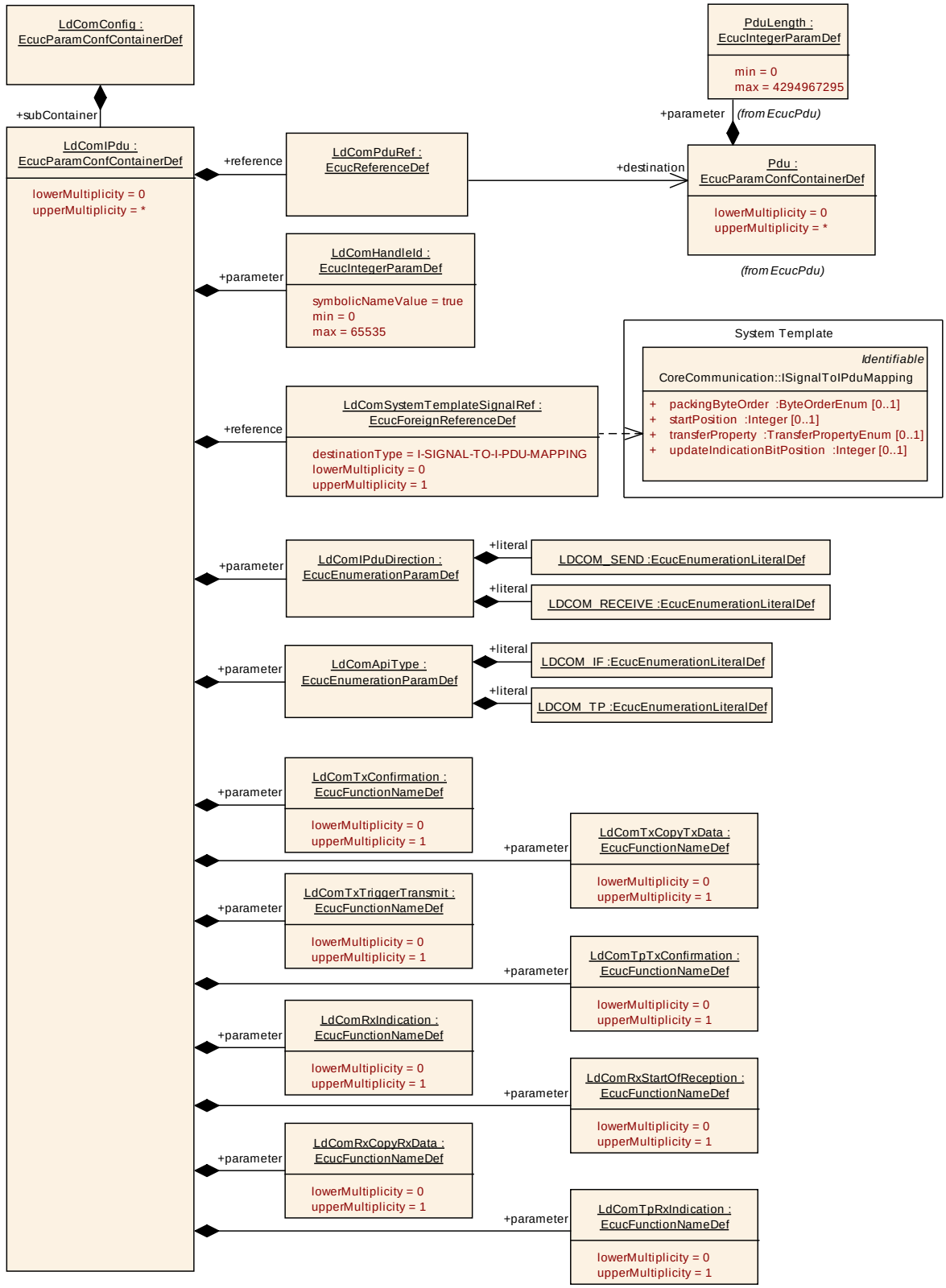

Figure 7 : Configuration LdComIPdu

10.2 Published Information

Published information contains data defined by the implementer of the SW module that does not change when the module is adapted (i.e. configured) to the actual HW/SW environment. It thus contains version and manufacturer information.

11 Not applicable requirements

None at this point in time.