

<b>Document Title</b>	Specification of Large Data COM
<b>Document Owner</b>	AUTOSAR
<b>Document Responsibility</b>	AUTOSAR
<b>Document Identification No</b>	655

<b>Document Status</b>	Final
<b>Part of AUTOSAR Standard</b>	Classic Platform
<b>Part of Standard Release</b>	4.4.0

<b>Document Change History</b>			
<b>Date</b>	<b>Release</b>	<b>Changed by</b>	<b>Change Description</b>
2018-10-31	4.4.0	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation</li> </ul>
2017-12-08	4.3.1	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>Minor corrections / clarifications / editorial changes; for details please refer to the ChangeDocumentation</li> </ul>
2016-11-30	4.3.0	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>Introduced reliable TxConfirmation</li> <li>Minor corrections</li> </ul>
2015-07-31	4.2.2	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>Fixed TriggerTransmit for dynamic length PDUs</li> <li>Added PreCompile configuration class for all symbolicNameValue parameters</li> </ul>
2014-10-31	4.2.1	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>Initial Release</li> </ul>

## Disclaimer

This work (specification and/or software implementation) and the material contained in it, as released by AUTOSAR, is for the purpose of information only. AUTOSAR and the companies that have contributed to it shall not be liable for any use of the work.

The material contained in this work is protected by copyright and other types of intellectual property rights. The commercial exploitation of the material contained in this work requires a license to such intellectual property rights.

This work may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only. For any other purpose, no part of the work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The work has been developed for automotive applications only. It has neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

## Table of Contents

1	Introduction and functional overview .....	5
2	Acronyms and abbreviations .....	6
3	Related documentation.....	7
3.1	Input documents .....	7
3.2	Related standards and norms .....	7
3.3	Related specification .....	7
4	Constraints and assumptions .....	8
4.1	Limitations .....	8
4.2	Applicability to car domains .....	8
5	Dependencies to other modules.....	9
5.1	RTE .....	9
5.2	PDU Router .....	9
5.3	Default Error Tracer (DET) .....	9
5.4	File structure.....	9
6	Requirements traceability .....	10
7	Functional specification .....	12
7.1	Initialization.....	12
7.2	De-initialization .....	12
7.3	Overall .....	12
7.4	Transmission .....	13
7.4.1	IF.....	13
7.4.2	TP .....	13
7.5	Reception .....	14
7.5.1	IF.....	14
7.5.2	TP .....	14
7.6	Development Errors.....	14
7.7	Production Errors.....	15
7.8	Extended Production Errors .....	15
7.9	Error notification .....	15
7.10	Error classification .....	15
7.10.1	Runtime Errors .....	15
7.10.2	Transient Faults .....	15
8	API specification.....	16
8.1	Imported types.....	16
8.2	Type definitions .....	16
8.2.1	LdCom_ConfigType .....	16
8.3	Function definitions.....	16
8.3.1	LdCom_Init.....	16
8.3.2	LdCom_DeInit.....	17
8.3.3	LdCom_GetVersionInfo .....	17
8.3.4	LdCom_Transmit .....	18
8.4	Call-back functions and notifications .....	18

8.4.1	LdCom_CopyTxData .....	19
8.4.2	LdCom_TpTxConfirmation .....	20
8.4.3	LdCom_StartOfReception .....	21
8.4.4	LdCom_CopyRxData .....	21
8.4.5	LdCom_TpRxIndication .....	22
8.4.6	LdCom_RxIndication.....	22
8.4.7	LdCom_TxConfirmation .....	23
8.4.8	LdCom_TriggerTransmit.....	23
8.5	Scheduled functions .....	24
8.6	Expected Interfaces.....	24
8.6.1	Mandatory Interfaces .....	24
8.6.2	Optional Interfaces.....	24
8.6.3	Configurable interfaces .....	25
8.7	Service Interfaces.....	29
9	Sequence diagrams .....	30
9.1	Transmission .....	30
9.1.1	TP-API .....	30
9.1.2	IF-API.....	31
9.1.3	TriggerTransmit.....	31
9.2	Reception .....	32
9.2.1	TP-API .....	32
9.2.2	IF-API.....	32
10	Configuration specification.....	33
10.1	Containers and configuration parameters .....	33
10.1.1	LdCom.....	33
10.1.2	LdComConfig.....	34
10.1.3	LdComGeneral .....	34
10.1.4	LdComIPdu.....	35
10.2	Published Information.....	42
11	Not applicable requirements .....	43

## 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

<b><i>Abbreviation / Acronym:</i></b>	<b><i>Description:</i></b>
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\_00050][The LdCom implementation shall include Det.h if LdComDevErrorDetect is enabled.] (SRS\_BSW\_00350)

## 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
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_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 support post-build configuration	SWS_LDCOM_00022, SWS_LDCOM_00052
SRS_BSW_00405	BSW Modules shall support multiple configuration sets	SWS_LDCOM_00022
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_00414	Init functions shall have a	SWS_LDCOM_00022

	pointer to a configuration structure as single parameter	
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\_Delnit 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_LdComCbkJCopyTxData_<sn>	LdComTxCopyTxData
Rte_LdComCbkJTpTxConfirmation_<sn>	LdComTpTxConfirmation
Rte_LdComCbkJRxIndication_<sn>	LdComRxIndication
Rte_LdComCbkJStartOfReception_<sn>	LdComRxStartOfReception
Rte_LdComCbkJCopyRxData_<sn>	LdComRxCopyRxData
Rte_LdComCbkJTpRxIndication_<sn>	LdComTpRxIndication
Rte_LdComCbkJTriggerTransmit_<sn>	LdComTxTriggerTransmit
Rte_LdComCbkJTxConfirmation_<sn>	LdComTxConfirmation

**Table 1: API to Parameter mapping**

**[SWS\_LDCOM\_00009]**[When called by the RTE, LdCom shall use the Signal Id ("id" parameter in the call) as LdComHandleId ([ECUC LdCom 00005](#)) to look-up the correct LdComIPdu in the LdCom configuration. Using the LdComPduRef configuration parameter ([ECUC LdCom 00010](#)) the corresponding PDU Id in the

PduR'S configuration shall be derived. This PDU Id shall then be used when forwarding the call towards the 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\_LdComCbkJRxIndication\_<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\_LdComCbkJStartOfReception\_<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\_LdComCbkJCopyRxData\_<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\_LdComTpRkJIndication\_<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

<i>Type of error</i>	<i>Related error code</i>	<i>Value [hex]</i>
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] [

<i>Module</i>	<i>Header File</i>	<i>Imported Type</i>
ComStack_Types	ComStackTypes.h	BufReq_ReturnType
	ComStackTypes.h	PduIdType
	ComStackTypes.h	PduInfoType
	ComStackTypes.h	PduLengthType
	ComStackTypes.h	RetryInfoType
Std_Types	StandardTypes.h	Std_ReturnType
	StandardTypes.h	Std_VersionInfoType

] (SRS\_BSW\_00384)

### 8.2 Type definitions

#### 8.2.1 LdCom\_ConfigType

[SWS\_LDCOM\_00052] [

<b>Name:</b>	LdCom_ConfigType	
<b>Type:</b>	Structure	
<b>Range:</b>	implementation specific	The contents of the initialization data structure are implementation specific
<b>Description:</b>	This type contains the implementation-specific post build configuration structure.	
<b>Available via:</b>	LdCom.h	

] (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.

Note: All functions in this chapter requires previous initialization (LdCom\_Init), except the following ones:

- LdCom\_Init
- LdCom\_GetVersionInfo

#### 8.3.1 LdCom\_Init

[SWS\_LDCOM\_00022] [



<b>Service name:</b>	LdCom_Init
<b>Syntax:</b>	<pre>void LdCom_Init(     const LdCom_ConfigType* config )</pre>
<b>Service ID[hex]:</b>	0x01
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Non Reentrant
<b>Parameters (in):</b>	config   Pointer to the AUTOSAR LdCom module's configuration data.
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	None
<b>Description:</b>	This service initializes internal and external interfaces and variables of the AUTOSAR LdCom module for the further processing.
<b>Available via:</b>	LdCom.h

] (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] [

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

] (SRS\_BSW\_00336)

### 8.3.3 LdCom\_GetVersionInfo

[SWS\_LDCOM\_00024] [

<b>Service name:</b>	LdCom_GetVersionInfo
<b>Syntax:</b>	<pre>void LdCom_GetVersionInfo(     Std_VersionInfoType* versioninfo )</pre>
<b>Service ID[hex]:</b>	0x03
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Non Reentrant

<b>Parameters (in):</b>	None
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	versioninfo   Pointer to where to store the version information of this module.
<b>Return value:</b>	None
<b>Description:</b>	Returns the version information of this module.
<b>Available via:</b>	LdCom.h

] (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]** [

<b>Service name:</b>	LdCom_Transmit
<b>Syntax:</b>	Std_ReturnType LdCom_Transmit( PduIdType Id, const PduInfoType* InfoPtr )
<b>Service ID[hex]:</b>	0x05
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Reentrant for different Ids. Non reentrant for the same Id.
<b>Parameters (in):</b>	Id   Identifier of the signal to be transmitted. InfoPtr   Length of and pointer to the signal data and pointer to MetaData.
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	Std_ReturnType   E_OK: Transmit request has been accepted.   E_NOT_OK: Transmit request has not been accepted.
<b>Description:</b>	Requests transmission of a signal.
<b>Available via:</b>	LdCom.h

] (SRS\_Com\_02110)

## 8.4 Call-back functions and notifications

This is a list of functions provided for other modules.

**[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\_TpRxIndications 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)

Note: All functions in this chapter requires that the LdCom module is initialized correctly.

### 8.4.1 LdCom\_CopyTxData

**[SWS\_LDCOM\_00027]** [

<b>Service name:</b>	LdCom_CopyTxData	
<b>Syntax:</b>	<pre>BufReq_ReturnType LdCom_CopyTxData (     PduIdType id,     const PduInfoType* info,     const RetryInfoType* retry,     PduLengthType* availableDataPtr )</pre>	
<b>Service ID[hex]:</b>	0x43	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	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_CONFENDING, 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.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	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.
<b>Return value:</b>	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.
<b>Description:</b>	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.	
<b>Available via:</b>	LdCom.h	

] (SRS\_Com\_02109)

## 8.4.2 LdCom\_TpTxConfirmation

[SWS\_LDCOM\_00028] [

<b>Service name:</b>	LdCom_TpTxConfirmation	
<b>Syntax:</b>	<pre>void LdCom_TpTxConfirmation(     PduIdType id,     Std_ReturnType result )</pre>	
<b>Service ID[hex]:</b>	0x48	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	id	Identification of the transmitted I-PDU.
	result	Result of the transmission of the I-PDU.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	None	
<b>Description:</b>	This function is called after the I-PDU has been transmitted on its network, the result indicates whether the transmission was successful or not.	
<b>Available via:</b>	LdCom.h	

] (SRS\_Com\_02109)

### 8.4.3 LdCom\_StartOfReception

#### [SWS\_LDCOM\_00029] [

<b>Service name:</b>	LdCom_StartOfReception	
<b>Syntax:</b>	<pre>BufReq_ReturnType LdCom_StartOfReception (     PduIdType id,     const PduInfoType* info,     PduLengthType TpSduLength,     PduLengthType* bufferSizePtr )</pre>	
<b>Service ID[hex]:</b>	0x46	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	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.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	bufferSizePtr	Available receive buffer in the receiving module. This parameter will be used to compute the Block Size (BS) in the transport protocol module.
<b>Return value:</b>	BufReq_ReturnType	<p>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.</p> <p>BUFREQ_E_NOT_OK: Connection has been rejected; reception is aborted. bufferSizePtr remains unchanged.</p> <p>BUFREQ_E_OVFL: No buffer of the required length can be provided; reception is aborted. bufferSizePtr remains unchanged.</p>
<b>Description:</b>	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.	
<b>Available via:</b>	LdCom.h	

] (SRS\_Com\_02109)

### 8.4.4 LdCom\_CopyRxData

#### [SWS\_LDCOM\_00030] [

<b>Service name:</b>	LdCom_CopyRxData	
<b>Syntax:</b>	<pre>BufReq_ReturnType LdCom_CopyRxData (     PduIdType id,     const PduInfoType* info,     PduLengthType* bufferSizePtr )</pre>	
<b>Service ID[hex]:</b>	0x44	
<b>Sync/Async:</b>	Synchronous	

<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	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.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	bufferSizePtr	Available receive buffer after data has been copied.
<b>Return value:</b>	BufReq_ReturnType	BUFREQ_OK: Data copied successfully BUFREQ_E_NOT_OK: Data was not copied because an error occurred.
<b>Description:</b>	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 buffer is written to the position indicated by bufferSizePtr.	
<b>Available via:</b>	LdCom.h	

] (SRS\_Com\_02109)

#### 8.4.5 LdCom\_TpRxIndication

[SWS\_LDCOM\_00031] [

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

] (SRS\_Com\_02109)

#### 8.4.6 LdCom\_RxIndication

[SWS\_LDCOM\_00032] [

<b>Service name:</b>	LdCom_RxIndication	
<b>Syntax:</b>	<pre>void LdCom_RxIndication(     PduIdType RxPduId,</pre>	

	const PduInfoType* PduInfoPtr )	
<b>Service ID[hex]:</b>	0x42	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant for different PduIds. Non reentrant for the same PduId.	
<b>Parameters (in):</b>	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.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	None	
<b>Description:</b>	Indication of a received PDU from a lower layer communication interface module.	
<b>Available via:</b>	LdCom.h	

] (SRS\_Com\_02110)

#### 8.4.7 LdCom\_TxConfirmation

[SWS\_LDCOM\_00056] [

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

] (SRS\_Com\_02110)

#### 8.4.8 LdCom\_TriggerTransmit

[SWS\_LDCOM\_00033] [

<b>Service name:</b>	LdCom_TriggerTransmit	
<b>Syntax:</b>	Std_ReturnType LdCom_TriggerTransmit( PduIdType TxPduId, PduInfoType* PduInfoPtr )	

<b>Service ID[hex]:</b>	0x41	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant for different PduIds. Non reentrant for the same PduId.	
<b>Parameters (in):</b>	TxPduId	ID of the SDU that is requested to be transmitted.
<b>Parameters (inout):</b>	PduInfoPtr	Contains a pointer to a buffer (SduDataPtr) to where the SDU data shall be copied, and the available buffer size in SduLength. On return, the service will indicate the length of the copied SDU data in SduLength.
<b>Parameters (out):</b>	None	
<b>Return value:</b>	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.
<b>Description:</b>	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.	
<b>Available via:</b>	LdCom.h	

] (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

None.

### 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	Header File	Description
Det_ReportError	Det.h	Service to report development errors.



PduR_LdComTransmit	PduR_LdCom.h	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\_LdComCbkJCopyTxData\_<sn>

[SWS\_LDCOM\_00036] [

<b>Service name:</b>	Rte_LdComCbkJCopyTxData_<sn>	
<b>Syntax:</b>	<pre>BufReq_ReturnType Rte_LdComCbkJCopyTxData_&lt;sn&gt;(     const PduInfoType* info,     const RetryInfoType* retry,     PduLengthType* availableDataPtr )</pre>	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant for same sn, otherwise Reentrant	
<b>Parameters (in):</b>	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.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	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.
<b>Return value:</b>	BufReq_ReturnType	<p>BUFREQ_OK: Data has been copied to the transmit buffer completely as requested.</p> <p>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.</p> <p>BUFREQ_E_NOT_OK: Data has not been copied. Request failed.</p>
<b>Description:</b>	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
<b>Available via:</b>	Rte_LdCom.h

] (SRS\_Com\_02109)

### 8.6.3.2 Rte\_LdComCbkJpTxConfirmation\_<sn>

[SWS\_LDCOM\_00037] [

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

] (SRS\_Com\_02109)

### 8.6.3.3 Rte\_LdComCbkJStartOfReception\_<sn>

[SWS\_LDCOM\_00038] [

<b>Service name:</b>	Rte_LdComCbkJStartOfReception_<sn>
<b>Syntax:</b>	BufReq_ReturnType Rte_LdComCbkJStartOfReception_<sn>( const PduInfoType* info, PduLengthType TpSduLength, PduLengthType* bufferSizePtr )
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Non Reentrant for same sn, otherwise Reentrant
<b>Parameters (in):</b>	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.
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	bufferSizePtr Available receive buffer in the receiving module. This parameter will be used to compute the Block Size (BS) in the transport protocol module.
<b>Return value:</b>	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.
<b>Description:</b>	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.	
<b>Available via:</b>	Rte_LdCom.h	

] (SRS\_Com\_02109)

### 8.6.3.4 Rte\_LdComCbkJCopyRxData\_<sn>

[SWS\_LDCOM\_00039] [

<b>Service name:</b>	Rte_LdComCbkJCopyRxData_<sn>	
<b>Syntax:</b>	BufReq_ReturnType Rte_LdComCbkJCopyRxData_<sn>( const PduInfoType* info, PduLengthType* bufferSizePtr )	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant for same sn, otherwise Reentrant	
<b>Parameters (in):</b>	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.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	bufferSizePtr	Available receive buffer after data has been copied.
<b>Return value:</b>	BufReq_ReturnType	BUFREQ_OK: Data copied successfully BUFREQ_E_NOT_OK: Data was not copied because an error occurred.
<b>Description:</b>	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.	
<b>Available via:</b>	Rte_LdCom.h	

] (SRS\_Com\_02109)

### 8.6.3.5 Rte\_LdComCbkJTpRxIndication\_<sn>

[SWS\_LDCOM\_00040] [

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

] (SRS\_Com\_02109)

### 8.6.3.6 Rte\_LdComCbkJxIndication\_<sn>

[SWS\_LDCOM\_00041] [

<b>Service name:</b>	Rte_LdComCbkJxIndication_<sn>	
<b>Syntax:</b>	<pre>void Rte_LdComCbkJxIndication_&lt;sn&gt;(     const PduInfoType* PduInfoPtr )</pre>	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant for same sn, otherwise Reentrant	
<b>Parameters (in):</b>	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.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	None	
<b>Description:</b>	Indication of a received PDU from a lower layer communication interface module.	
<b>Available via:</b>	Rte_LdCom.h	

] (SRS\_Rte\_00246, SRS\_Com\_02110)

### 8.6.3.7 Rte\_LdComCbkJriggerTransmit\_<sn>

[SWS\_LDCOM\_00042] [

<b>Service name:</b>	Rte_LdComCbkJriggerTransmit_<sn>	
<b>Syntax:</b>	<pre>Std_ReturnType Rte_LdComCbkJriggerTransmit_&lt;sn&gt;(     PduInfoType* PduInfoPtr )</pre>	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant for same sn, otherwise Reentrant	
<b>Parameters (in):</b>	None	
<b>Parameters (inout):</b>	PduInfoPtr	Contains a pointer to a buffer (SduDataPtr) to where the SDU data shall be copied, and the available buffer size in SduLength. On return, the service will indicate the length of the copied SDU data in SduLength.
<b>Parameters (out):</b>	None	
<b>Return value:</b>	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.
<b>Description:</b>	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.	
<b>Available via:</b>	Rte_LdCom.h	

] (SRS\_Com\_02111)

### 8.6.3.8 Rte\_LdComCbkJxConfirmation\_<sn>

[SWS\_LDCOM\_00053] [

<b>Service name:</b>	Rte_LdComCbkJxConfirmation_<sn>	
----------------------	---------------------------------	--

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

] (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

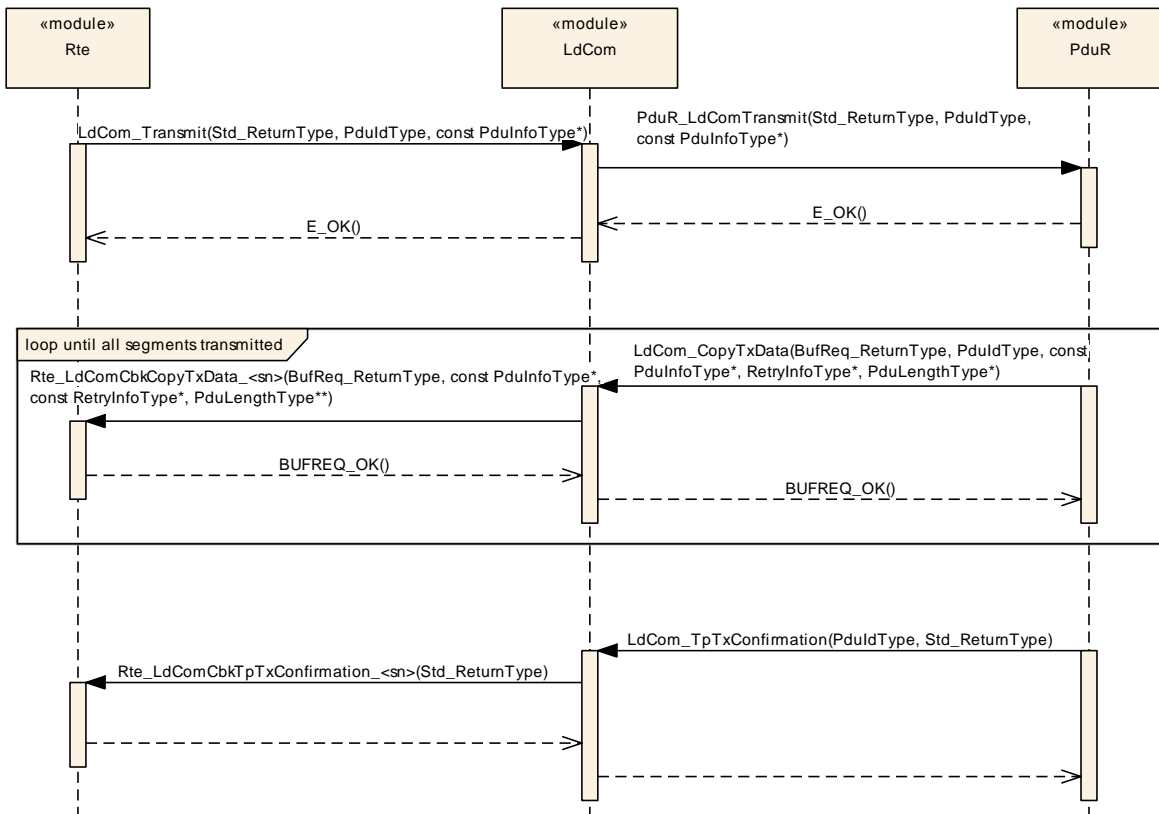
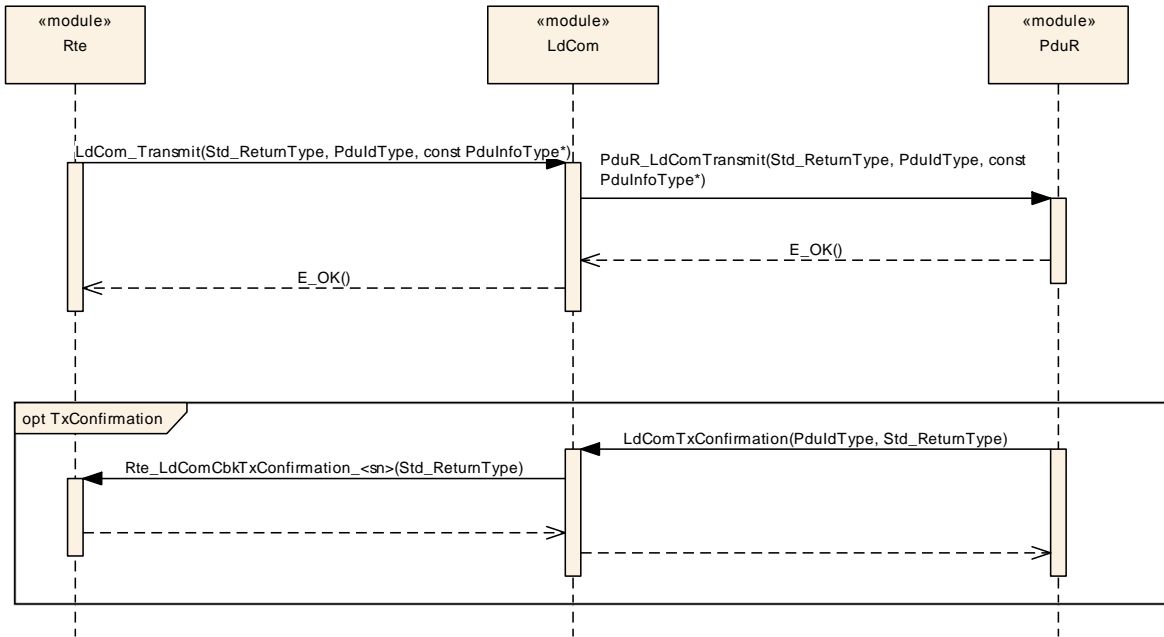


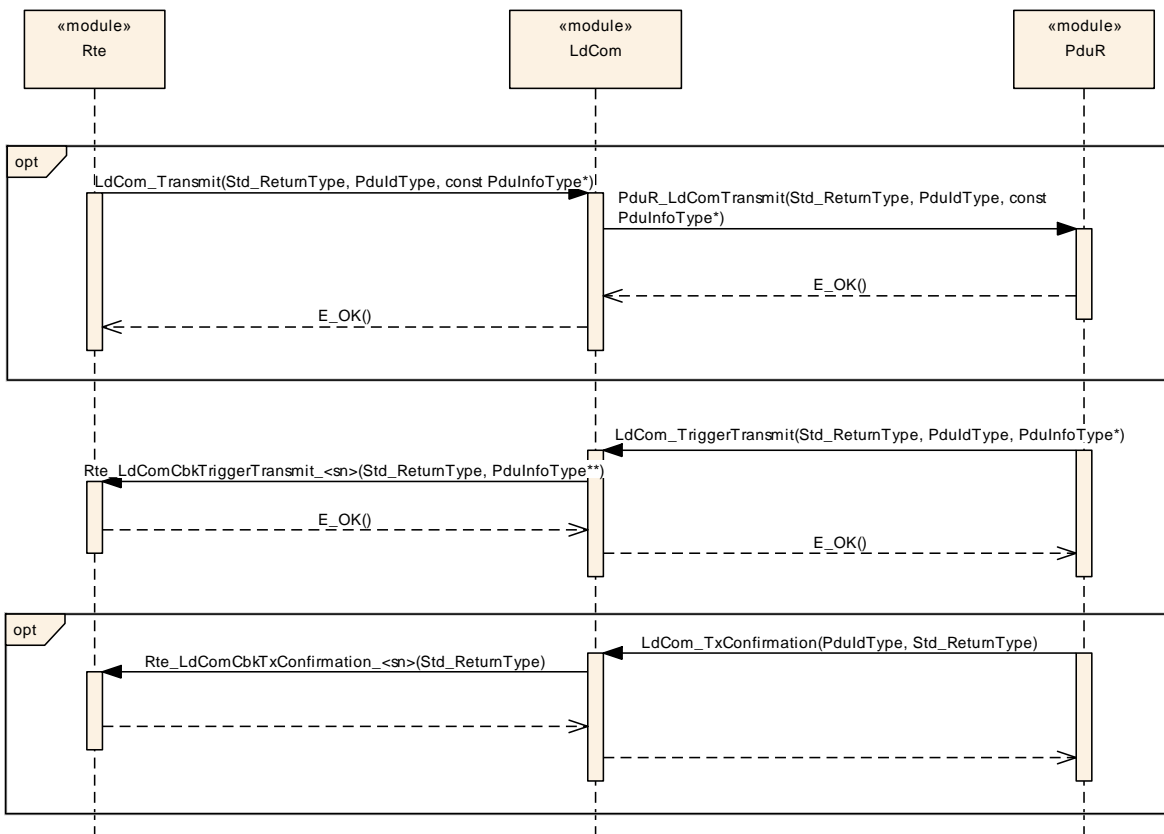
Figure 1 - Transmission via TP-API

**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

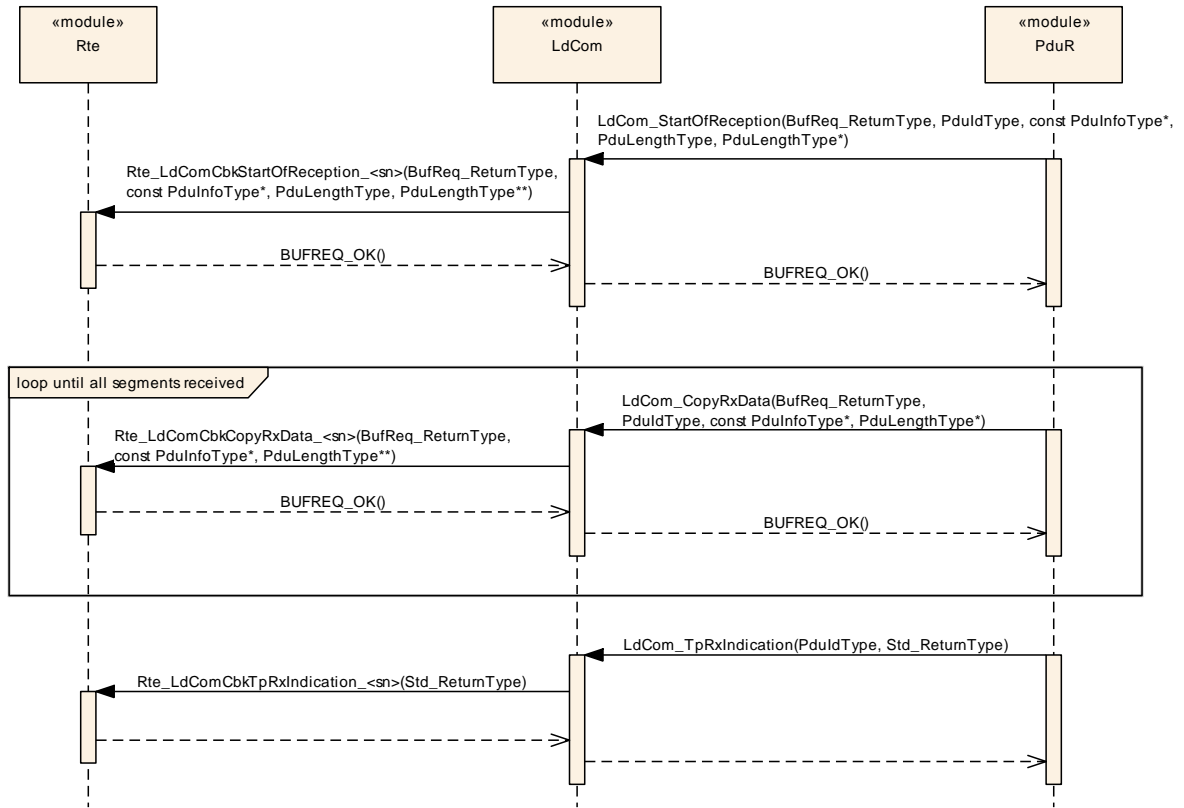


Figure 4 - Reception via TP-API

### 9.2.2 IF-API

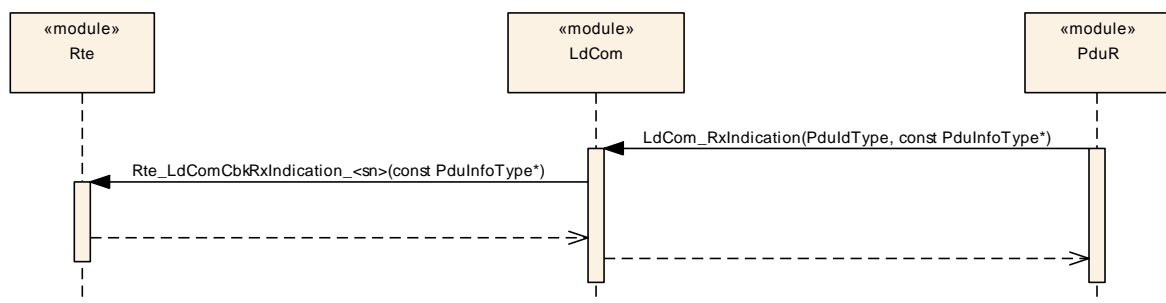


Figure 5 - Reception via IF-API



## 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

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

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
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.

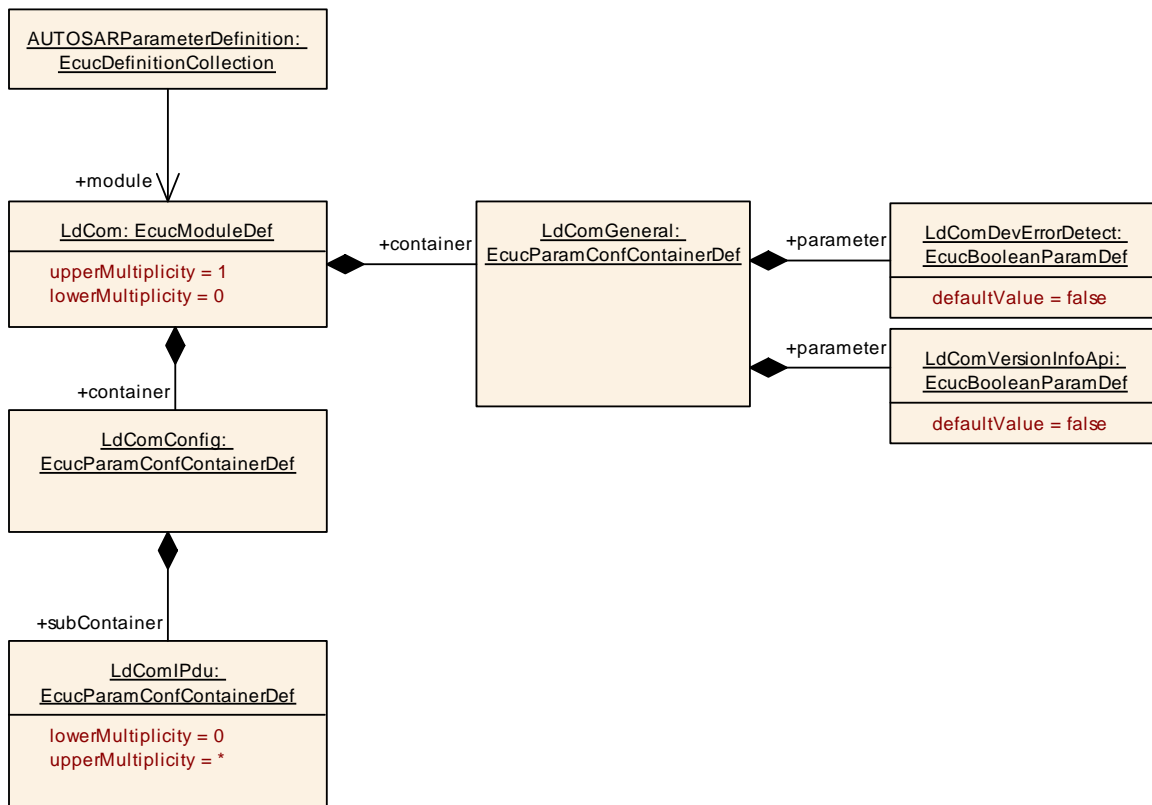


Figure 6 : Configuration LdCom

### 10.1.2 LdComConfig

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

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

### 10.1.3 LdComGeneral

<b>SWS Item</b>	<b>ECUC_LdCom_00004 :</b>
<b>Container Name</b>	LdComGeneral
<b>Description</b>	Contains the general configuration parameters of the LdCom module.
<b>Configuration Parameters</b>	

<b>SWS Item</b>	<b>ECUC_LdCom_00020 :</b>		
<b>Name</b>	LdComDevErrorDetect		
<b>Parent Container</b>	LdComGeneral		
<b>Description</b>	Switches the development error detection and notification on or off. <ul style="list-style-type: none"> <li>• true: detection and notification is enabled.</li> <li>• false: detection and notification is disabled.</li> </ul>		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_LdCom_00012 :</b>		
<b>Name</b>	LdComVersionInfoApi		
<b>Parent Container</b>	LdComGeneral		
<b>Description</b>	Activate/Deactivate the version information API (LdCom_GetVersionInfo). <ul style="list-style-type: none"> <li>• True: version information API activated</li> <li>• False: version information API deactivated</li> </ul>		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		

<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.1.4 LdComIPdu

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

<b>SWS Item</b>	<b>ECUC_LdCom_00002 :</b>		
<b>Name</b>	LdComApiType		
<b>Parent Container</b>	LdComIPdu		
<b>Description</b>	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.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	LDCOM_IF	sent or received via interface API.	
	LDCOM_TP	sent or received via transport protocol API.	
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_LdCom_00005 :</b>		
<b>Name</b>	LdComHandleId		
<b>Parent Container</b>	LdComIPdu		
<b>Description</b>	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.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
<b>Range</b>	0 .. 65535		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	

	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

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

<b>SWS Item</b>	<b>ECUC_LdCom_00013 :</b>		
<b>Name</b>	LdComRxCopyRxData		
<b>Parent Container</b>	LdComIPdu		
<b>Description</b>	Only on receiver side: Name of Rte_LdComCbxCopyRxData callback function to be called.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucFunctionNameDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_LdCom_00014 :</b>		
<b>Name</b>	LdComRxIndication		
<b>Parent Container</b>	LdComIPdu		
<b>Description</b>	Only on receiver side: Name of Rte_LdComCbRxIndication callback function to be called.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucFunctionNameDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		

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

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

<b>SWS Item</b>	<b>ECUC_LdCom_00016 :</b>		
<b>Name</b>	LdComTpRxIndication		
<b>Parent Container</b>	LdComIPdu		
<b>Description</b>	Only on receiver side: Name of Rte_LdComCbKTpRxIndication callback function to be called.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucFunctionNameDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE

	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

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

<b>SWS Item</b>	<b>ECUC_LdCom_00021 :</b>		
<b>Name</b>	LdComTxConfirmation		
<b>Parent Container</b>	LdComIPdu		
<b>Description</b>	Only on sender side: Name of Rte_LdComCbKTxConfirmation callback function to be called.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucFunctionNameDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_LdCom_00018 :</b>		
<b>Name</b>	LdComTxCopyTxData		
<b>Parent Container</b>	LdComIPdu		

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

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

<b>SWS Item</b>	<b>ECUC_LdCom_00010 :</b>		
<b>Name</b>	LdComPduRef		
<b>Parent Container</b>	LdComIPdu		
<b>Description</b>	Reference to the global Pdu.		
<b>Multiplicity</b>	1		
<b>Type</b>	Reference to [ Pdu ]		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD

	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		
<b>SWS Item</b>	<b>ECUC_LdCom_00011 :</b>		
<b>Name</b>	LdComSystemTemplateSignalRef		
<b>Parent Container</b>	LdComIPdu		
<b>Description</b>	Reference to the ISignalToIPduMapping that contains a reference to the ISignal (System Template).		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Foreign reference to [ I-SIGNAL-TO-I-PDU-MAPPING ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		
<b>No Included Containers</b>			



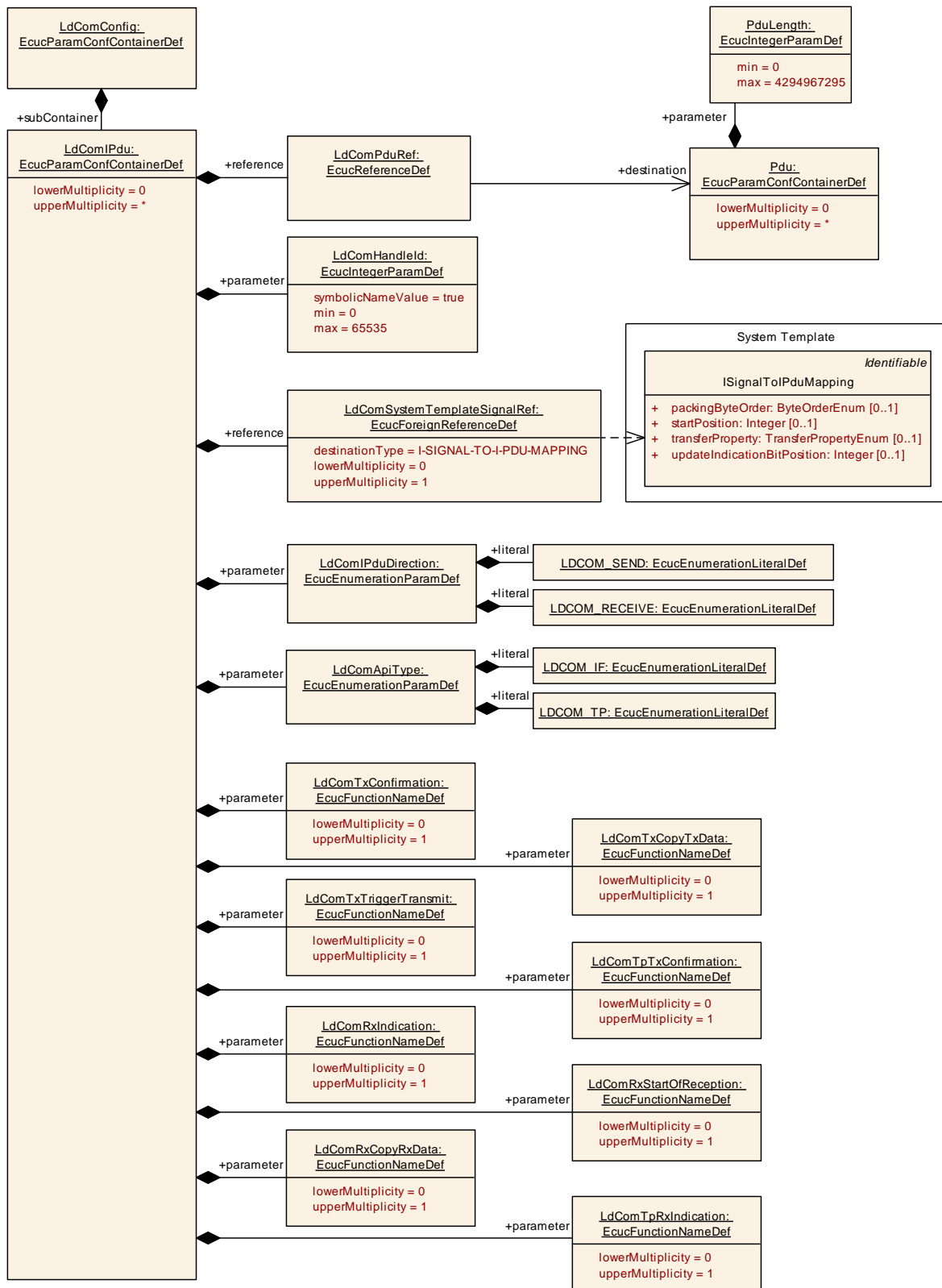


Figure 7 : Configuration LdComIPdu

## 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.