Document Title	Specification of Large Data COM
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	655

Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R21-11

Document Change History

Date	Release	Changed by	Change Description
2021-11-25	R21-11	AUTOSAR Release Management	• Introduced the support for “Software Clusters”. Therefore extend the LdComUser configuration and introduced handle-id-based call-back functions • Minor corrections
2020-11-30	R20-11	AUTOSAR Release Management	• Clean up error section • Changed Document Status from Final to published
2019-11-28	R19-11	AUTOSAR Release Management	• Clean up diagrams in chapter 10 • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• Header File Cleanupminor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2017-12-08	4.3.1	AUTOSAR Release Management	• Minor corrections / clarifications / editorial changes; for details please refer to the ChangeDocumentation
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

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 / SwCluC_LdComProxy 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 its users (the RTE, SwCluC_LdComProxy)
- Provision of received signals to its users (RTE, SwCluC_LdComProxy)
- 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

[7] Specification of Default Error Tracer
AUTOSAR_SWS_DefaultErrorTracer.pdf

[8] Specification of Software Cluster Connection
AUTOSAR_SWS_SoftwareClusterConnection.pdf

[9] Specification of ECU Configuration
AUTOSAR_TPS_ECUConfiguration.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

Large data 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.
Large data 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
- No TP Fan-out

4.2 Applicability to car domains

No restrictions.

5 Dependencies to other modules

5.1 LdCom Users

5.1.1 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].

5.1.2 SwCluC

For SwCluC the AUTOSAR LdCom module is also an additional mean to send and receive signals. In AUTOSAR, the SwCluC_LdComProxy (LowProxy) is the higher layer (in the HOST SW Cluster) above the LdCom module responsible for dispatching the Callback invocations from the LdCom towards the applicative SW Clusters. For further information, see [8].

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 (for further information, see [7]).

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	SWS_LDCOM_00024, SWS_LDCOM_00045

	information of a dedicated module implementation	
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, SWS_LDCOM_00061, SWS_LDCOM_91008
SRS_Com_02108	Support of Large Data COM	SWS_LDCOM_00005, SWS_LDCOM_00009, SWS_LDCOM_00035, SWS_LDCOM_00046, SWS_LDCOM_00057, SWS_LDCOM_00058, SWS_LDCOM_00061
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, SWS_LDCOM_00063, SWS_LDCOM_00065, SWS_LDCOM_00066, SWS_LDCOM_00067, SWS_LDCOM_91001, SWS_LDCOM_91002, SWS_LDCOM_91003, SWS_LDCOM_91004, SWS_LDCOM_91005, SWS_LdCom_CONSTR_00009, SWS_LdCom_CONSTR_00010, SWS_LdCom_CONSTR_00011
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, SWS_LDCOM_00061, SWS_LDCOM_00064, SWS_LDCOM_91006
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, SWS_LDCOM_00060, SWS_LDCOM_91007
SRS_Com_02114	AUTOSAR COM and LargeDataCOM shall support independent development of CP Software Clusters	SWS_LDCOM_00057, SWS_LDCOM_00058, SWS_LDCOM_00060, SWS_LDCOM_00061, SWS_LDCOM_00063, SWS_LDCOM_00064, SWS_LDCOM_00065, SWS_LDCOM_00066, SWS_LDCOM_00067, SWS_LDCOM_91001, SWS_LDCOM_91002, SWS_LDCOM_91003, SWS_LDCOM_91004, SWS_LDCOM_91005, SWS_LDCOM_91006, SWS_LDCOM_91007, SWS_LDCOM_91008, SWS_LdCom_CONSTR_00001, SWS_LdCom_CONSTR_00002, SWS_LdCom_CONSTR_00003,

		SWS_LdCom_CONSTR_00004, SWS_LdCom_CONSTR_00005, SWS_LdCom_CONSTR_00006, SWS_LdCom_CONSTR_00007, SWS_LdCom_CONSTR_00008, SWS_LdCom_CONSTR_00009, SWS_LdCom_CONSTR_00010, SWS_LdCom_CONSTR_00011
SRS_Rte_00246	Support of Efficient COM for large data	SWS_LDCOM_00041, SWS_LDCOM_91006

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] {OBSOLETE}⌈When called by PduR LdCom shall use the passed PDU Id as Handle Id (LdComHandleIdECUC_LdCom_00005), to derive the actualAPI from configuration and use it when passing the call towards RTE.⌋ (SRS_Com_02108)

[SWS_LDCOM_00057]⌈ When called by its users (e.g. RTE, SwCluC LdCom Proxy), 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, SRS_Com_02114)

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. The LdCom user callback handle Id (LdComUserCbkHandleId) parameter identifies the corresponding Signal/PDU.

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

<LdComUser_LdComCbkCopyTxData> {DRAFT}	LdComUserCallbackName with LdComUserCallbackType set to LDCOM_TP_COPY_TX_DATA {DRAFT}
<LdComUser_LdComCbkTpTxConfirmation>{DRAFT}	LdComUserCallbackName with LdComUserCallbackType set to LDCOM_TP_COPY_TX_CONFIRMATION{DRAFT}
<LdComUser_LdComCbkRxIndication>{DRAFT}	LdComUserCallbackName with LdComUserCallbackType set to LDCOM_TP_COPY_RX_INDICATION{DRAFT}
<LdComUser_LdComCbkStartOfReception>{DRAFT}	LdComUserCallbackName with LdComUserCallbackType set to LDCOM_RX_START_OF_RECEPTION {DRAFT}
<LdComUser_LdComCbkCopyRxData>{DRAFT}	LdComUserCallbackName with LdComUserCallbackType set to LDCOM_TP_COPY_RX_DATA {DRAFT}
<LdComUser_LdComCbkTpRxIndication>{DRAFT}	LdComUserCallbackName with LdComUserCallbackType set to LDCOM_TP_COPY_RX_INDICATION {DRAFT}
<LdComUser_LdComCbkTriggerTransmit>{DRAFT}	LdComUserCallbackName with LdComUserCallbackType set to LDCOM_TP_COPY_TX_TRIGGER_TRANSMIT {DRAFT}
<LdComUser_LdComCbkTxConfirmation>{DRAFT}	LdComUserCallbackName with LdComUserCallbackType set to LDCOM_TX_CONFIRMATION{DRAFT}

Table 1: API to Parameter mapping

[SWS_LDCOM_00009] {OBSOLETE} ⌈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)

[SWS_LDCOM_00058] {DRAFT} ⌈ When called by its users (e.g. RTE, SwCluC LdCom Proxy), 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, SRS_Com_02114)

Even if the concept of LdCom user provides a lot of flexibility to support access by multiple users including their notifications some limitations needs to be considered.

In general, multiple writers can cause race conditions if the writers are not coordinated. In addition, neither the behavior of TriggerTransmit interfacing nor the TP interfacing does support notification towards multiple users for the same IPdu.

[SWS_LdCom_CONSTR_00001] {DRAFT} ⌈Sent IPdus shall be owned by at most one LdCom user. ⌋(SRS_Com_02114)

Nevertheless, reading an IPdu by several LdCom Users in the same or different Software Clusters is possible but the partititon assignment of the IPdus needs to be respected.

[SWS_LdCom_CONSTR_00002] {DRAFT} ⌈All LdCom users registering notifications for IPdus shall reside on the EcucPartition on which the LdCom module handles the related IPdu.⌋(SRS_Com_02114)

7.4 Transmission

Transmission is initiated by the LdCom user (e.g. RTE, SwCluC_LdComProxy) by invoking 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] {OBSOLETE} ⌈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_00060] {DRAFT} ⌈When LdCom_TriggerTransmit is invoked, LdCom shall use the passed PDU Id as Handle Id (LdComHandleId ECUC_LdCom_00005) to derive the corresponding <LdComUser_LdComCbkTriggerTransmit> user notification callback and call it with the according LdCom user callback handle Id.⌋(SRS_Com_02111, SRS_Com_02114)

[SWS_LdCom_CONSTR_00003]{DRAFT}⌈Only a single LdCom user can be notified with <LdComUser_LdComCbkTriggerTransmit>.⌋(SRS_Com_02114)

[SWS_LDCOM_00046] {OBSOLETE} ⌈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)

[SWS_LDCOM_00061] {DRAFT} ⌈When LdCom_TxConfirmation is invoked, LdCom shall use the passed PDU Id as Handle Id (LdComHandleId ECUC_LdCom_00005) to derive the corresponding <LdComUser_LdComCbkTxConfirmation> user notification callback and call it with the according LdCom user callback handle Id. ⌋(SRS_Com_02044, SRS_Com_02108, SRS_Com_02110, SRS_Com_02114)

[SWS_LdCom_CONSTR_00004] {DRAFT} ⌈Only a single LdCom user can be notified with <LdComUser_LdComCbkTriggerTransmit>.⌋(SRS_Com_02114)

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] {OBSOLETE} ⌈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)

[SWS_LDCOM_00063] {DRAFT} ⌈When LdCom_CopyTxData and LdCom_TpTxConfirmation are invoked, LdCom shall use the passed PDU Id as Handle Id (LdComHandleId ECUC_LdCom_00005) to derive the corresponding <LdComUser_LdComCbkCopyTxData> or <LdComUser_LdComCbkTpTxConfirmation> user notification callback and call it with the according LdCom user callback handle Id.⌋(SRS_Com_02109, SRS_Com_02114)

[SWS_LdCom_CONSTR_00005] {DRAFT} ⌈Only a single LdCom user can be notified with <LdComUser_LdComCbkCopyTxData> or <LdComUser_LdComCbkTpTxConfirmation>.⌋(SRS_Com_02114)

7.5 Reception

7.5.1 IF

[SWS_LDCOM_00014] {OBSOLETE} ⌈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)

[SWS_LDCOM_00064] {DRAFT} ⌈When LdCom_RxIndication is invoked, LdCom shall use the passed PDU Id as Handle Id (LdComHandleId ECUC_LdCom_00005) to derive the corresponding <LdComUser_LdComCbkRxIndication> user notification callbacks and call them with the according LdCom user callback handle Id.⌋ (SRS_Com_02110, SRS_Com_02114)

7.5.2 TP

[SWS_LDCOM_00015] {OBSOLETE} ⌈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_00065] {DRAFT} ⌈When LdCom_StartOfReception is invoked, LdCom shall use the passed PDU Id as Handle Id (LdComHandleId

ECUC_LdCom_00005) to derive the corresponding <LdComUser_LdComCbkStartOfReception> user notification callback and call it with the according LdCom user callback handle Id.](SRS_Com_02109,SRS_Com_02114)

[SWS_LdCom_CONSTR_00006] {DRAFT} [Only a single LdCom user can be notified with <LdComUser_LdComCbkStartOfReception>.](SRS_Com_02114)

[SWS_LDCOM_00016] {OBSOLETE} [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_00066] {DRAFT} [When LdCom_CopyRxData is invoked, LdCom shall use the passed PDU Id as Handle Id (LdComHandleId ECUC_LdCom_00005) to derive the corresponding <LdComUser_LdComCbkCopyRxData> user notification callback and call it with the according LdCom user callback handle Id.](SRS_Com_02109, SRS_Com_02114)

[SWS_LdCom_CONSTR_00007] {DRAFT} [Only a single LdCom user can be notified with <LdComUser_LdComCbkCopyRxData>.](SRS_Com_02114)

[SWS_LDCOM_00017] {OBSOLETE} [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)

[SWS_LDCOM_00067] {DRAFT} [When LdCom_TpRxIndication is invoked, LdCom shall use the passed PDU Id as Handle Id (LdComHandleId ECUC_LdCom_00005) to derive the corresponding <LdComUser_LdComTpRxIndication> user notification callback and call it with the according LdCom user callback handle Id.](SRS_Com_02109, SRS_Com_02114)

[SWS_LdCom_CONSTR_00008] {DRAFT} [Only a single LdCom user can be notified with < LdComUser_LdComTpRxIndication>.](SRS_Com_02114)

7.6 Error classification

Section 7.2 "Error Handling" of the document "General Specification of Basic Software Modules" describes the error handling of the Basic Software in detail. Above all, it constitutes a classification scheme consisting of five error types which may occur in BSW modules.

Based on this foundation, the following section specifies particular errors arranged in the respective subsections below.

7.6.1 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)

7.6.2 Runtime Errors

There are no runtime errors.

7.6.3 Transient Faults

There are no transient faults.

7.6.4 Production Errors

There are no production errors.

7.6.5 Extended Production Errors

There are no extended production errors.

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	*Header File*	*Imported Type*
ComStack_Types	ComStack_Types.h	BufReq_ReturnType
	ComStack_Types.h	CbkHandleIdType (draft)
	ComStack_Types.h	PduIdType
	ComStack_Types.h	PduInfoType
	ComStack_Types.h	PduLengthType
	ComStack_Types.h	RetryInfoType
	ComStack_Types.h	TpDataStateType
Std	Std_Types.h	Std_ReturnType
	Std_Types.h	Std_VersionInfoType

⌋(SRS_BSW_00384)

8.2 Type definitions

8.2.1 LdCom_ConfigType

[SWS_LDCOM_00052]⌈

Name	LdCom_ConfigType	
Kind	Structure	
Elements	implementation specific	
	Type	--
	Comment	The contents of the initialization data structure are implementation specific
Description	This type contains the implementation-specific post build configuration structure.	
Available via	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]⌈

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.	
Available via	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][

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.
Available via	LdCom.h

](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.	

Available via	LdCom.h

](SRS_BSW_00407, SRS_BSW_00003)

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

8.3.4 LdCom_Transmit

[SWS_LDCOM_00026][

Service Name	LdCom_Transmit	
Syntax	`Std_ReturnType LdCom_Transmit ( PduIdType Id, const PduInfoType* InfoPtr )`	
Service ID [hex]	0x49	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different Ids. Non reentrant for the same Id.	
Parameters (in)	Id	Identifier of the signal to be transmitted.
	InfoPtr	Length of and pointer to the signal data and pointer to Meta Data.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Transmit request has been accepted. E_NOT_OK: Transmit request has not been accepted.
Description	Requests transmission of a signal.	
Available via	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_TpRxIndicationshall 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][

Service Name	LdCom_CopyTxData	
Syntax	`BufReq_ReturnType LdCom_CopyTxData ( PduIdType id, const PduInfoType* info, const 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)	available DataPtr	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_-Return-Type	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.	
Available via	LdCom.h	

](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	E_OK: The PDU was transmitted. E_NOT_OK: Transmission of the PDU failed.
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.	
Available via	LdCom.h	

](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.
	TpSdu Length	Total length of the N-SDU to be received.
Parameters (inout)	None	
Parameters	buffer	Available receive buffer in the receiving module. This parameter will be

(out)	SizePtr	used to compute the Block Size (BS) in the transport protocol module.
Return value	BufReq_-Return-Type	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.	
Available via	LdCom.h	

](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)	bufferSize Ptr	Available receive buffer after data has been copied.
Return value	BufReq_-Return-Type	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 buffer is written to the position indicated by bufferSizePtr.
Available via	LdCom.h

](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	E_OK: The PDU was received. E_NOT_OK: Reception of the PDU failed.
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.	
Available via	LdCom.h	

](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.	
Available via	LdCom.h	

](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	None	

(inout)	
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.
Available via	LdCom.h

](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_-Return-Type	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.	
Available via	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_LdComCbkCopyTxData_<sn>

[SWS_LDCOM_00036]{OBSOLETE} [

Service Name	Rte_LdComCbkCopyTxData_<sn> (obsolete)	

Syntax	`BufReq_ReturnType Rte_LdComCbkCopyTxData_<sn> (`
	`  const PduInfoType* info,`
	`  const 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)	available DataPtr	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_-Return-Type	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 **Tags:** atp.Status=obsolete	
Available via	Rte_Cbk.h	

](SRS_Com_02109)

Please note that [SWS_LDCOM_00036] is set to OBSOLETE and will be replaced by **[SWS_LDCOM_91001]** to support the LdCom user approach (using callback handle Ids) introduced by a concept, which will be validated after the release of R21-11.

8.6.3.2 LdComCbkCopyTxData

[SWS_LDCOM_91001]{DRAFT} [

Service Name	<LdComUser_LdComCbkCopyTxData> (draft)	

Syntax	`BufReq_ReturnType <LdComUser_LdComCbkCopyTxData> (`
	`  CbkHandleIdType LdComUserCbkHandleId,`
	`  const PduInfoType* info,`
	`  const RetryInfoType* retry,`
	`  PduLengthType* availableDataPtr`
	`)`	

Service ID [hex]	0x4a	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same LdComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	LdCom UserCbk HandleId	LdCom user callback handle Id corresponding to 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	Will not be handled by LdCom and its upper layer.
Parameters (inout)	None	
Parameters (out)	available DataPtr	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_-Return-Type	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->Tp DataState 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 **Tags:** atp.Status=draft	
Available via	LdComUserHeaderInclude ([ECUC_LdCom_xxx04])	

](SRS_Com_02109, SRS_Com_02114)

8.6.3.3 Rte_LdComCbkTpTxConfirmation_<sn>

[SWS_LDCOM_00037]{OBSOLETE} [

Service Name	Rte_LdComCbkTpTxConfirmation_<sn> (obsolete)	
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. **Tags:** atp.Status=obsolete	
Available via	Rte_Cbk.h	

](SRS_Com_02109)

Please note that [SWS_LDCOM_00037] is set to OBSOLETE and will be replaced by **[SWS_LDCOM_91002]** to support the LdCom user approach (using callback handle Ids) introduced by a concept, which will be validated after the release of R21-11.

8.6.3.4 LdComCbkTpTxConfirmation

[SWS_LDCOM_91002]{DRAFT} [

Service Name	<LdComUser_LdComCbkTpTxConfirmation> (draft)	
Syntax	`void <LdComUser_LdComCbkTpTxConfirmation> ( CbkHandleIdType LdComUserCbkHandleId, Std_ReturnType result )`	
Service ID [hex]	0x4b	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same LdComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	LdComUserCbk HandleId	LdCom user callback handle Id corresponding to the transmitted I-PDU
	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. **Tags:** atp.Status=draft
Available via	LdComUserHeaderInclude ([ECUC_LdCom_xxx04])

](SRS_Com_02109, SRS_Com_02114)

8.6.3.5 Rte_LdComCbkStartOfReception_<sn>

[SWS_LDCOM_00038]{OBSOLETE} [

Service Name	Rte_LdComCbkStartOfReception_<sn> (obsolete)	

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.
	TpSdu Length	Total length of the N-SDU to be received.
Parameters (inout)	None	
Parameters (out)	buffer SizePtr	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_-Return-Type	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.	

	Tags: atp.Status=obsolete
Available via	Rte_Cbk.h

](SRS_Com_02109)

Please note that [SWS_LDCOM_00038] is set to OBSOLETE and will be replaced by **[SWS_LDCOM_91003]** to support the LdCom user approach (using Callback handle Ids) introduced by a concept, which will be validated after the release of R21-11.

8.6.3.6 LdComCbkStartOfReception

[SWS_LDCOM_91003]{DRAFT} [

Service Name	<LdComUser_LdComCbkStartOfReception> (draft)	

Syntax	`BufReq_ReturnType <LdComUser_LdComCbkStartOfReception> (`
	`  CbkHandleIdType LdComUserCbkHandleId,`
	`  const PduInfoType* info,`
	`  PduLengthType TpSduLength,`
	`  PduLengthType* bufferSizePtr`
	`)`	

Service ID [hex]	0x4c	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same LdComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	LdCom UserCbk HandleId	LdCom user callback handle Id corresponding to 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.
	TpSdu Length	Total length of the N-SDU to be received.
Parameters (inout)	None	
Parameters (out)	bufferSize Ptr	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_- Return- Type	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. **Tags:** atp.Status=draft
Available via	LdComUserHeaderInclude ([ECUC_LdCom_xxx04])

](SRS_Com_02109, SRS_Com_02114)

8.6.3.7 Rte_LdComCbkCopyRxData_<sn>

[SWS_LDCOM_00039]{OBSOLETE} [

Service Name	Rte_LdComCbkCopyRxData_<sn> (obsolete)	

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_-Return-Type	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. **Tags:** atp.Status=obsolete	
Available via	Rte_Cbk.h	

](SRS_Com_02109)

Please note that [SWS_LDCOM_00039] is set to OBSOLETE and will be replaced by **[SWS_LDCOM_91004]** to support the LdCom user approach (using callback handle Ids) introduced by a concept, which will be validated after the release of R21-11.

8.6.3.8 LdComCbkCopyRxData

[SWS_LDCOM_91004]{DRAFT} [

Service	<LdComUser_LdComCbkCopyRxData> (draft)

Name		

Syntax	`BufReq_ReturnType <LdComUser_LdComCbkCopyRxData> (`
	`  CbkHandleIdType LdComUserCbkHandleId,`
	`  const PduInfoType* info,`
	`  PduLengthType* bufferSizePtr`
	`)`	

Service ID [hex]	0x4d	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same LdComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	LdCom UserCbk HandleId	LdCom user callback handle Id corresponding to 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)	bufferSize Ptr	Available receive buffer after data has been copied.
Return value	BufReq_-Return-Type	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. **Tags:** atp.Status=draft	
Available via	LdComUserHeaderInclude ([ECUC_LdCom_xxx04])	

](SRS_Com_02109, SRS_Com_02114)

8.6.3.9 Rte_LdComCbkTpRxIndication_<sn>

[SWS_LDCOM_00040]{OBSOLETE} [

Service Name	Rte_LdComCbkTpRxIndication_<sn> (obsolete)	

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. **Tags:** atp.Status=obsolete
Available via	Rte_Cbk.h

](SRS_Com_02109)

Please note that [SWS_LDCOM_00040] is set to OBSOLETE and will be replaced by **[SWS_LDCOM_91005]** to support the LdCom user approach (using callback handle Ids) introduced by a concept, which will be validated after the release of R21-11.

8.6.3.10 LdComCbkTpRxIndication

[SWS_LDCOM_91005]{DRAFT} [

Service Name	<LdComUser_LdComCbkTpRxIndication> (draft)	

Syntax	`void <LdComUser_LdComCbkTpRxIndication> (`
	`  CbkHandleIdType LdComUserCbkHandleId,`
	`  Std_ReturnType result`
	`)`	

Service ID [hex]	0x4e	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same LdComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	LdComUserCbk HandleId	LdCom user callback handle Id corresponding to 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. **Tags:** atp.Status=draft	
Available via	LdComUserHeaderInclude ([ECUC_LdCom_xxx04])	

](SRS_Com_02109, SRS_Com_02114)

8.6.3.11 Rte_LdComCbkRxIndication_<sn>

[SWS_LDCOM_00041]{OBSOLETE} [

Service Name	Rte_LdComCbkRxIndication_<sn> (obsolete)	

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

Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same sn, otherwise Reentrant	
Parameters (in)	Pdu InfoPtr	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. **Tags:** atp.Status=obsolete	
Available via	Rte_Cbk.h	

](SRS_Rte_00246, SRS_Com_02110)

Please note that [SWS_LDCOM_00041] is set to OBSOLETE and will be replaced by **[SWS_LDCOM_91006]** to support the LdCom user approach (using callback handle Ids) introduced by a concept, which will be validated after the release of R21-11.

8.6.3.12 LdComCbkRxIndication

[SWS_LDCOM_91006]{DRAFT} [

Service Name	<LdComUser_LdComCbkRxIndication> (draft)	

Syntax	`void <LdComUser_LdComCbkRxIndication> (`
	`  CbkHandleIdType LdComUserCbkHandleId,`
	`  const PduInfoType* PduInfoPtr`
	`)`	

Service ID [hex]	0x4f	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant for same LdComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	LdComUser CbkHandleId	LdCom user callback handle Id corresponding to received I-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. **Tags:** atp.Status=draft
Available via	LdComUserHeaderInclude ([ECUC_LdCom_xxx04])

](SRS_Rte_00246, SRS_Com_02110, SRS_Com_02114)

8.6.3.13 Rte_LdComCbkTriggerTransmit_<sn>

[SWS_LDCOM_00042]{OBSOLETE} [

Service Name	Rte_LdComCbkTriggerTransmit_<sn> (obsolete)	

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 Sdu Length.
Parameters (out)	None	
Return value	Std_-Return-Type	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. **Tags:** atp.Status=obsolete	
Available via	Rte_Cbk.h	

](SRS_Com_02111)

Please note that [SWS_LDCOM_00042] is set to OBSOLETE and will be replaced by **[SWS_LDCOM_91007]** to support the LdCom user approach (using callback handle Ids) introduced by a concept, which will be validated after the release of R21-11.

8.6.3.14 LdComCbkTriggerTransmit

[SWS_LDCOM_91007]{DRAFT} [

Service Name	<LdComUser_LdComCbkTriggerTransmit> (draft)	
Syntax	`Std_ReturnType <LdComUser_LdComCbkTriggerTransmit> ( CbkHandleIdType LdComUserCbkHandleId, PduInfoType* PduInfoPtr )`	
Service ID [hex]	0x50	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same LdComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	LdComUser CbkHandleId	LdCom user callback handle Id corresponding to the ID of the SDU that is requested to be transmitted
	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 Sdu Length.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	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. **Tags:** atp.Status=draft	
Available via	LdComUserHeaderInclude ([ECUC_LdCom_xxx04])	

](SRS_Com_02111, SRS_Com_02114)

8.6.3.15 Rte_LdComCbkTxConfirmation_<sn>

[SWS_LDCOM_00053]{OBSOLETE} [

Service Name	Rte_LdComCbkTxConfirmation_<sn> (obsolete)
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. **Tags:** atp.Status=obsolete	
Available via	Rte_Cbk.h	

](SRS_Com_02044)

Please note that [SWS_LDCOM_00053] is set to OBSOLETE and will be replaced by **[SWS_LDCOM_91008]** to support the LdCom user approach (using callback handle Ids) introduced by a concept, which will be validated after the release of R21-11.

8.6.3.16 LdComCbkTxConfirmation

[SWS_LDCOM_91008]{DRAFT} [

Service Name	<LdComUser_LdComCbkTxConfirmation> (draft)	

Syntax	`void <LdComUser_LdComCbkTxConfirmation> (`
	`  CbkHandleIdType LdComUserCbkHandleId,`
	`  Std_ReturnType result`
	`)`	

Service ID [hex]	0x51	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same LdComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	LdComUserCbk HandleId	LdCom user callback handle Id corresponding to 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. **Tags:** atp.Status=draft	
Available via	LdComUserHeaderInclude ([ECUC_LdCom_xxx04])	

](SRS_Com_02044, SRS_Com_02114)

8.7 Service Interfaces

None.

9 Sequence diagrams

This chapter contains sequence charts showing the involvement of LdCom into interactions between its users (e.g. RTE, SwCluC LdCom Proxy) and PduR.

9.1 Transmission

9.1.1 TP-API

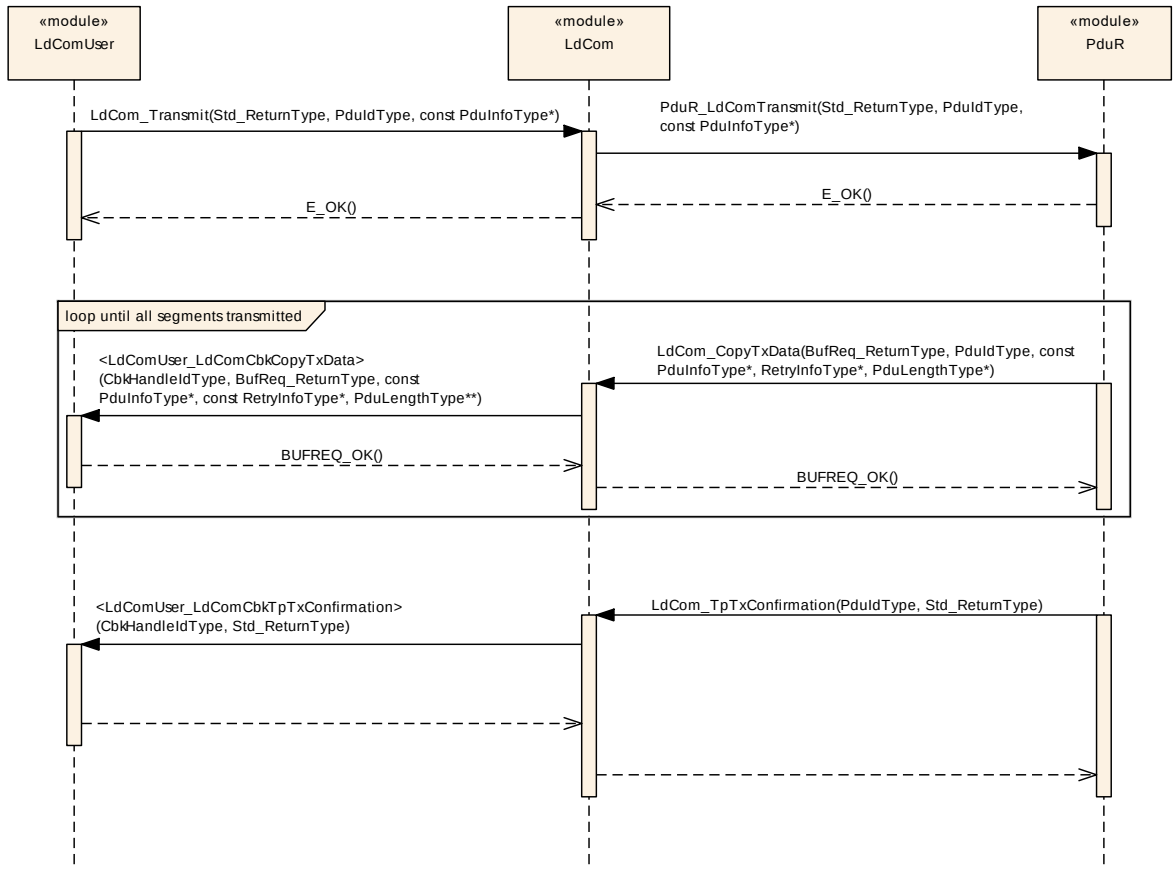

Figure 1 - Transmission via TP-API

9.1.2 IF-API

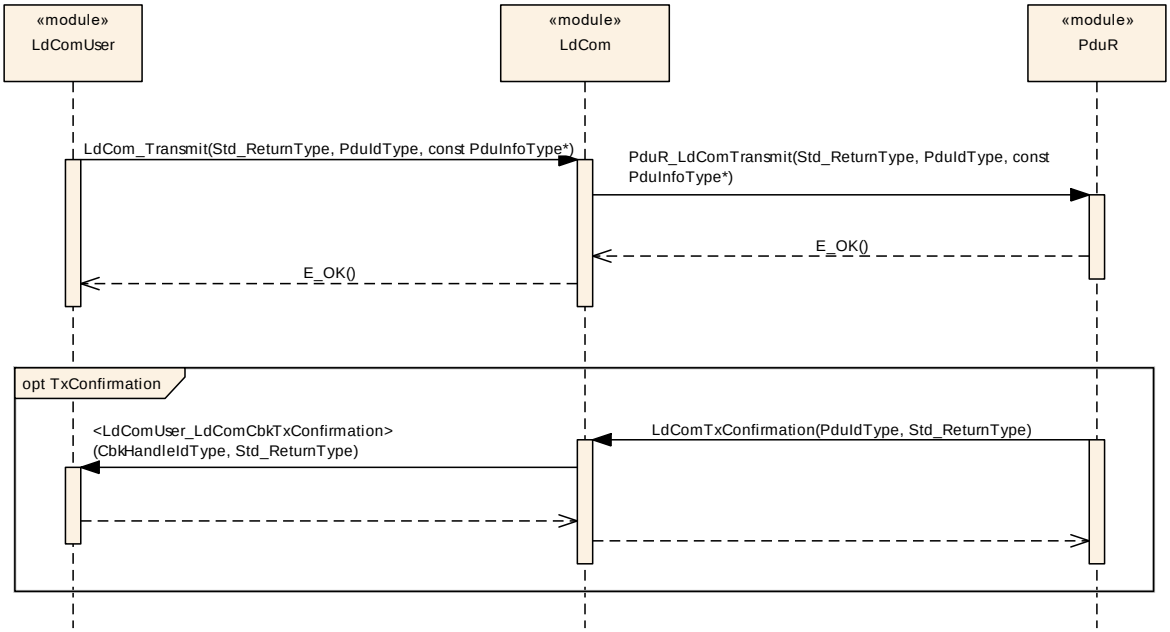

Figure 2 - Transmission via IF-API

9.1.3 TriggerTransmit

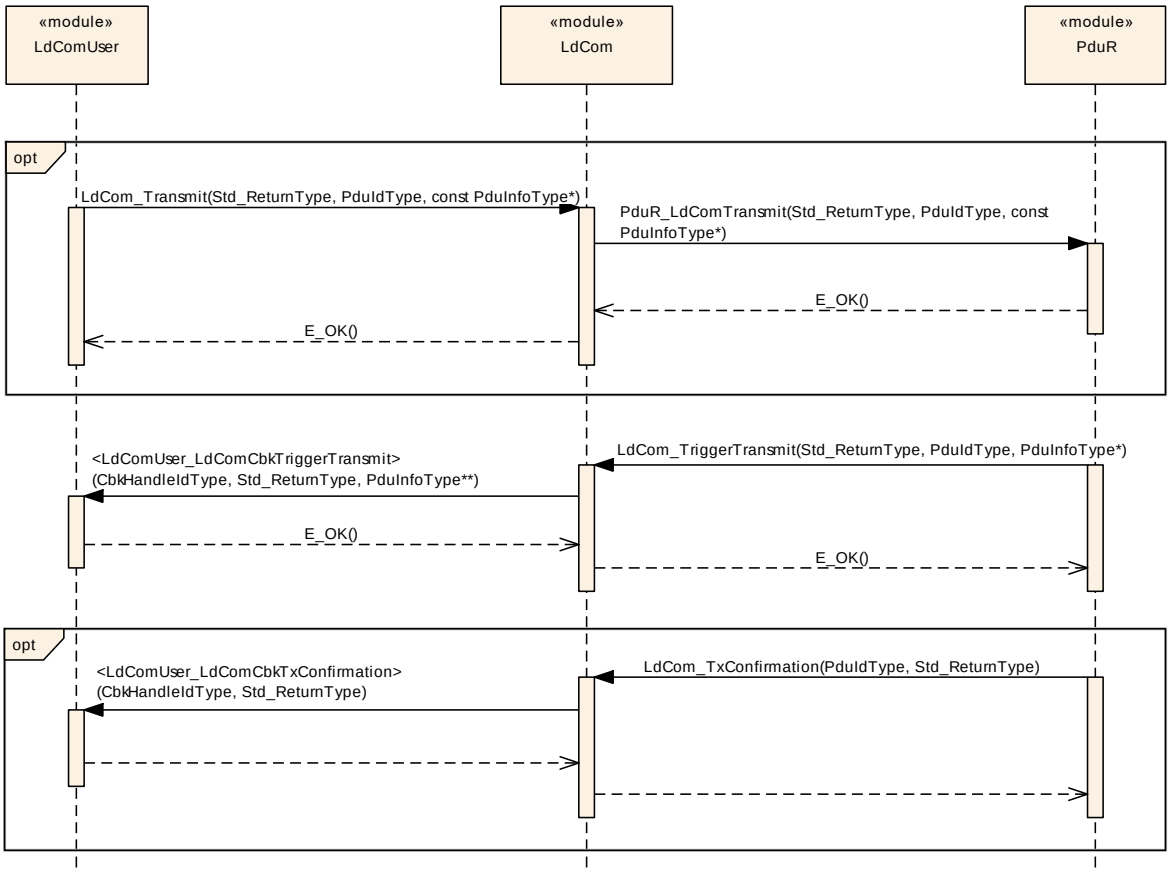

Figure 3 – TriggerTransmit

9.2 Reception

9.2.1 TP-API

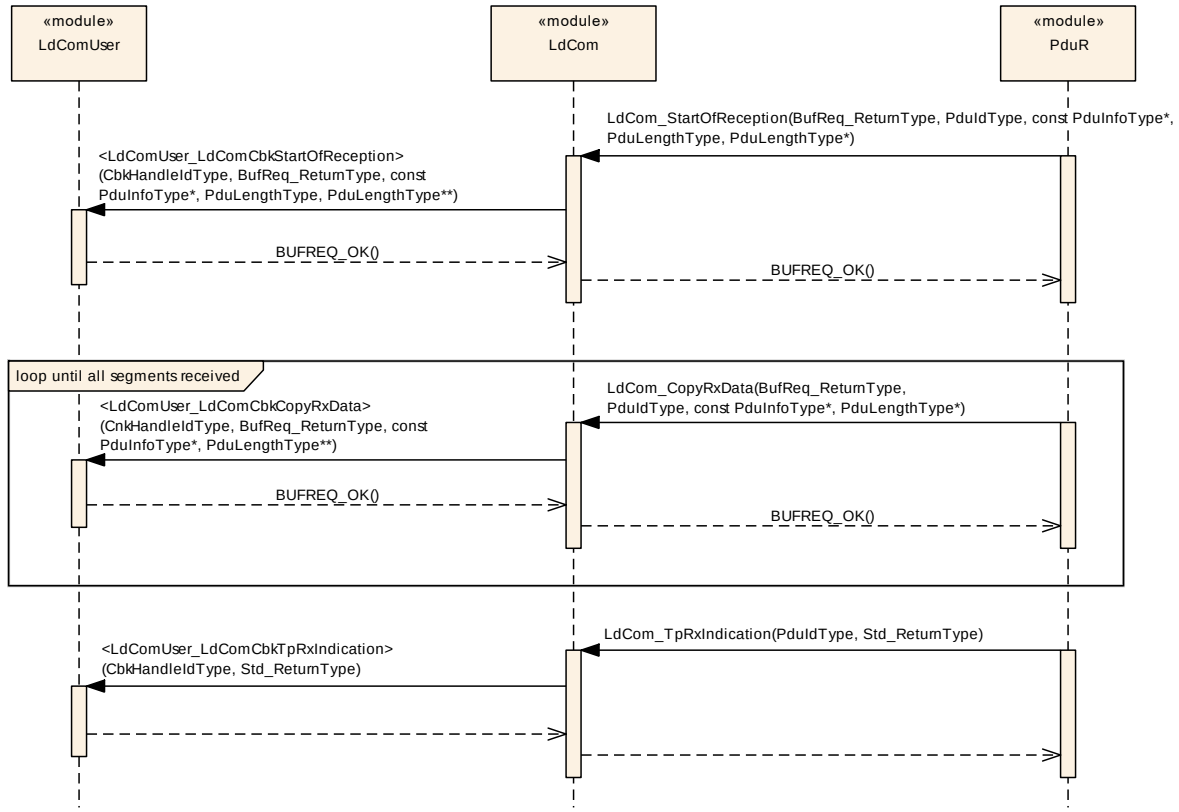

Figure 4 - Reception via TP-API

9.2.2 IF-API

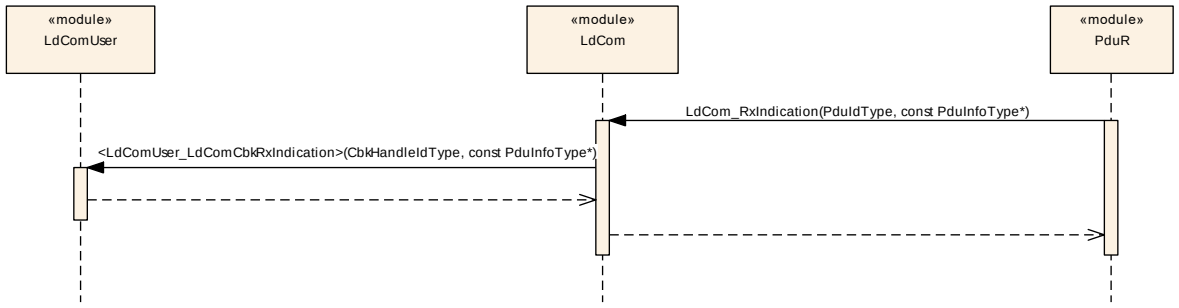

Figure 5 - Reception via IF-API

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 Chapters0 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.

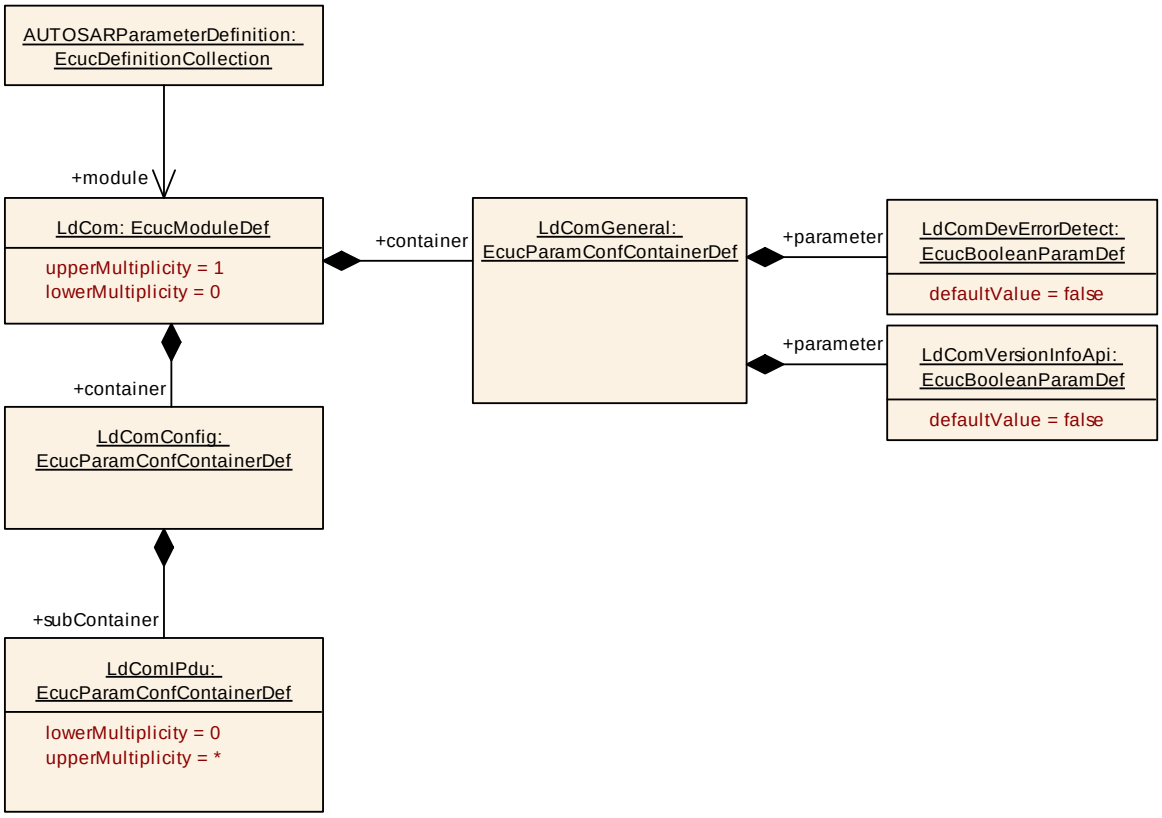

Figure 6 : Configuration LdCom

10.1.2 LdComConfig

SWS Item	**ECUC_LdCom_00003 :**
Container Name	LdComConfig
Parent Container	LdCom
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 LdCom's signal (IPdu) inside LdCom.
LdComUserModule	0..*	Contains the configuration parameters of the LdCom user modules. **Tags:** atp.Status=draft

10.1.3 LdComGeneral

SWS Item	**ECUC_LdCom_00004 :**
Container Name	LdComGeneral
Parent Container	LdCom

Description	Contains the general configuration parameters of the LdCom module.
Configuration Parameters	

SWS Item	**ECUC_LdCom_00020 :**		
Name	LdComDevErrorDetect		
Parent Container	LdComGeneral		
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		
Parent Container	LdComGeneral		
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		
Parent Container	LdComConfig		
Description	Contains the configuration parameters of the LdCom's signal (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		
Parent Container	LdComIPdu		
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		
Parent Container	LdComIPdu		
Description	This is the ID used by the LdCom users (e.g. RTE) to invoke LdCom. A corresponding shortName is created, which is used for the invocations of the users (e.g. 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		
Parent Container	LdComIPdu		
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 : (Obsolete)		
Name	LdComRxCopyRxData		
Parent Container	LdComIPdu		
Description	Only on receiver side: Name of Rte_LdComCbkCopyRxData callback function to be called. **Tags:** atp.Status=obsolete		
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 : (Obsolete)		
Name	LdComRxIndication		
Parent Container	LdComIPdu		
Description	Only on receiver side: Name of Rte_LdComCbkRxIndication callback function to be called. **Tags:** atp.Status=obsolete		
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 : (Obsolete)
Name	LdComRxStartOfReception
Parent Container	LdComIPdu
Description	Only on receiver side: Name of Rte_LdComCbkStartOfReception callback

	function to be called. **Tags:** atp.Status=obsolete		
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 : (Obsolete)**		
Name	LdComTpRxIndication		
Parent Container	LdComIPdu		
Description	Only on receiver side: Name of Rte_LdComCbkTpRxIndication callback function to be called. **Tags:** atp.Status=obsolete		
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 : (Obsolete)**
Name	LdComTpTxConfirmation
Parent Container	LdComIPdu
Description	Only on sender side: Name of Rte_LdComCbkTpTxConfirmation callback function to be called. **Tags:** atp.Status=obsolete
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 : (Obsolete)**		
Name	LdComTxConfirmation		
Parent Container	LdComIPdu		
Description	Only on sender side: Name of Rte_LdComCbkTxConfirmation callback function to be called. **Tags:** atp.Status=obsolete		
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 : (Obsolete)**
Name	LdComTxCopyTxData
Parent Container	LdComIPdu
Description	Only on sender side: Name of Rte_LdComCbkCopyTxData callback function to be called. **Tags:** atp.Status=obsolete
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 : (Obsolete)**		
Name	LdComTxTriggerTransmit		
Parent Container	LdComIPdu		
Description	Only on sender side: Name of Rte_LdComCbkTriggerTransmit callback function to be called. If defined TriggerTransmit has to be supported for this signal. **Tags:** atp.Status=obsolete		
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		
Parent Container	LdComIPdu		
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

Parent Container	LdComIPdu		
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

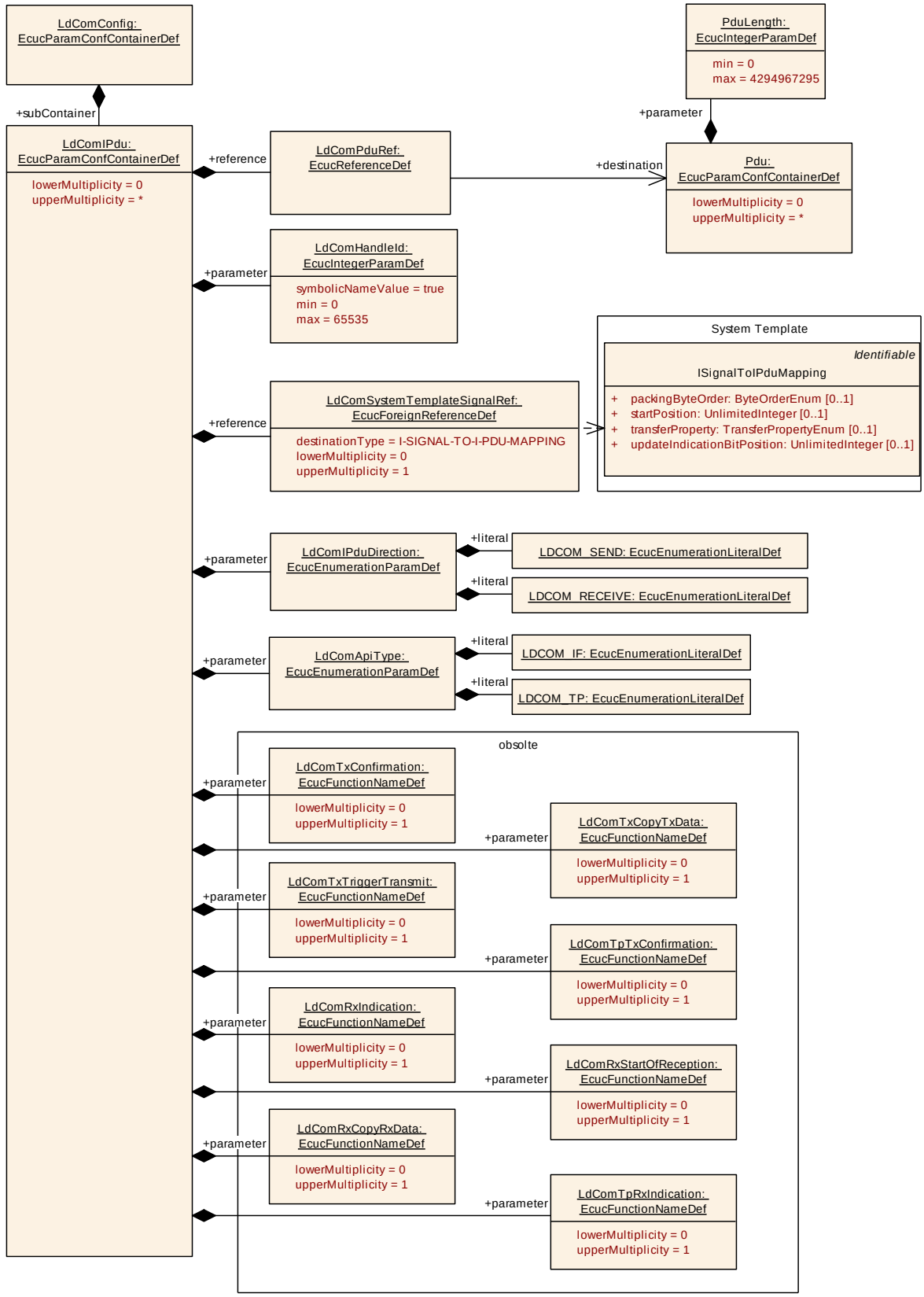

Figure 7 : Configuration LdComIPdu

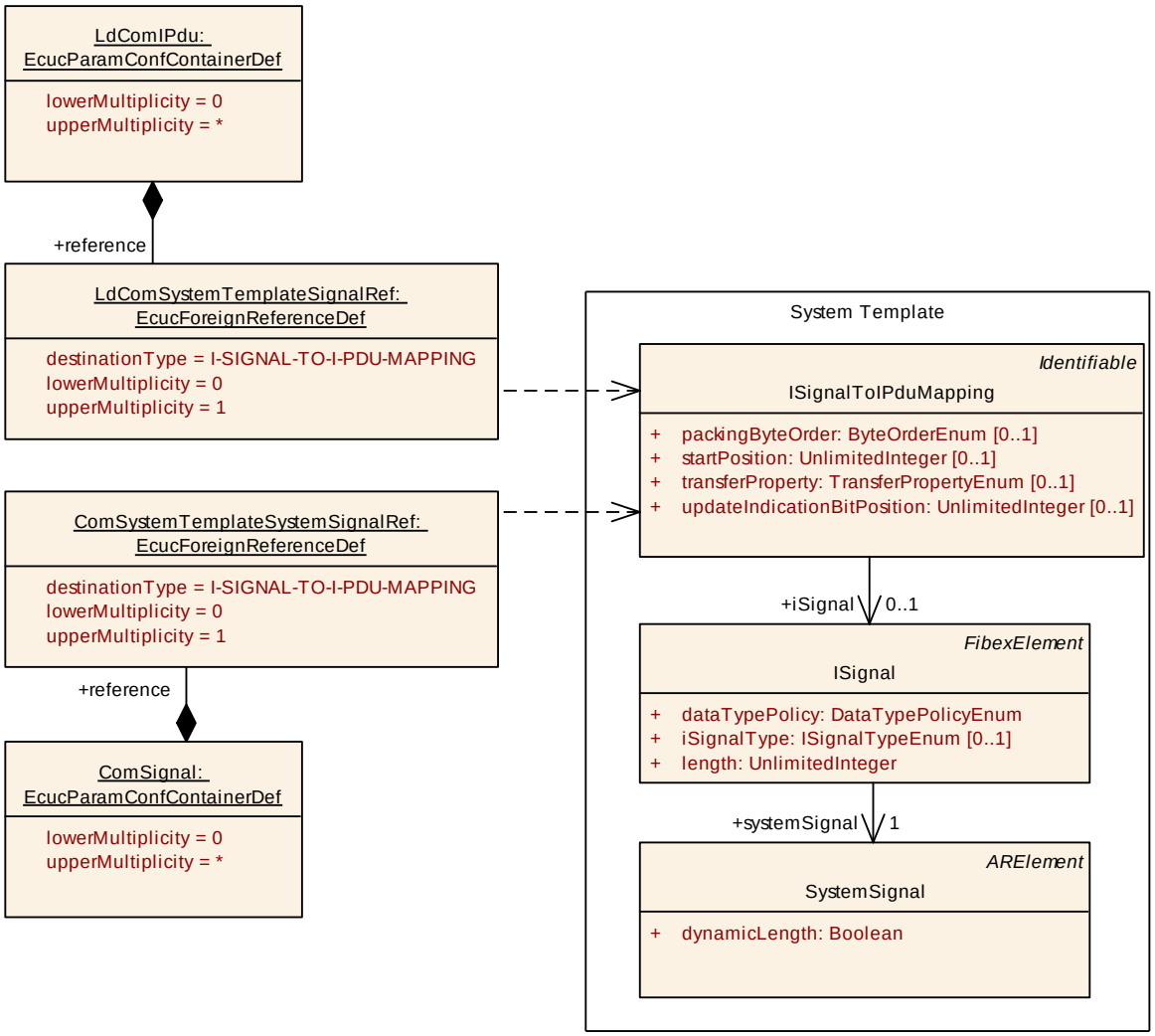

Figure 8 Configuration LdComIPdu – choice if ISignal is handled by LdCom (LdComSystemTemplateSignalRef) or by Com (ComSystemTemplateSystemSignalRef)

10.1.5 LdComUserModule

SWS Item	**ECUC_LdCom_00029 :**		
Container Name	LdComUserModule		
Parent Container	LdComConfig		
Description	Contains the configuration parameters of the LdCom user modules. **Tags:** atp.Status=draft		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Configuration Parameters			

SWS Item	**ECUC_LdCom_00032 :**
Name	LdComUserModuleCnfRef
Parent Container	LdComUserModule
Description	Reference to the LdCom user module configuration.

	Tags: atp.Status=draft		
Multiplicity	1		
Type	Reference to destinationUri [LdComUser]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: ECU		

No Included Containers

The concept of “Software Clusters” enables the splitting of the software of an AUTOSAR Classic Platform Architecture into smaller units has an impact on the LdCom module as well. In fact, the LdCom module can now have an arbitrary of users (RTE, SwCluC, and CDD), and therefore relies on the usage of URI References (See [9], Section URI Reference) to link the LdCom to its user(s) in the model.
To guaranty the compatibility between configurations of the LdCom module and its users, the LdComUserUriDefSet (see **ECUC_LdCom_00034 :**) defines the required parameters and containers. This means, an LdCom user shall configure LdComUserModuleCnf container (including its sub-containers), which holds the configuration of the LdCom IPdus it transmits and receives (via dedicated notification callbacks).

An LdCom user may span over one or multiple ECUC partitions. However, it is an implementation specific decision of the respective LdCom user how this can be achieved. Two different architecture patterns therefore apply:

1. **ECUC Partition specific LCom user**

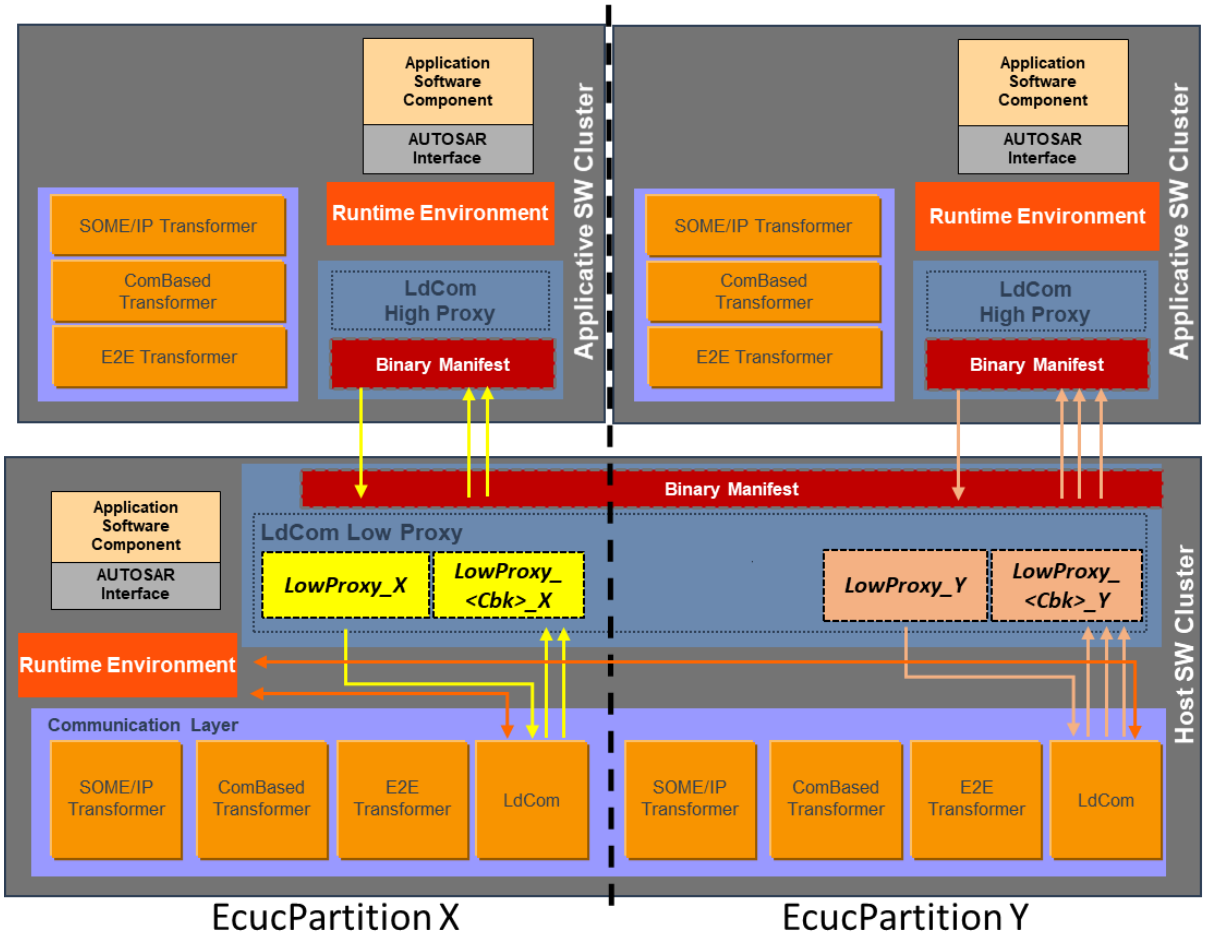

Figure 9 ECUC Partition specific LdCom user Overview

With this approach, the LdCom user module provides dedicated instances for each configured partition, on which LdCom (notification callback) invocations shall take place. However, this mandates that the LdCom user provides multiple main functions, each one bound to the relevant partition. The LdCom user's notification callbacks are invoked in the context of one partition only. Identification of the partition context can be done with a simple "callback function → partition" lookup table.

2. ECUC Partition agnostic LdCom user

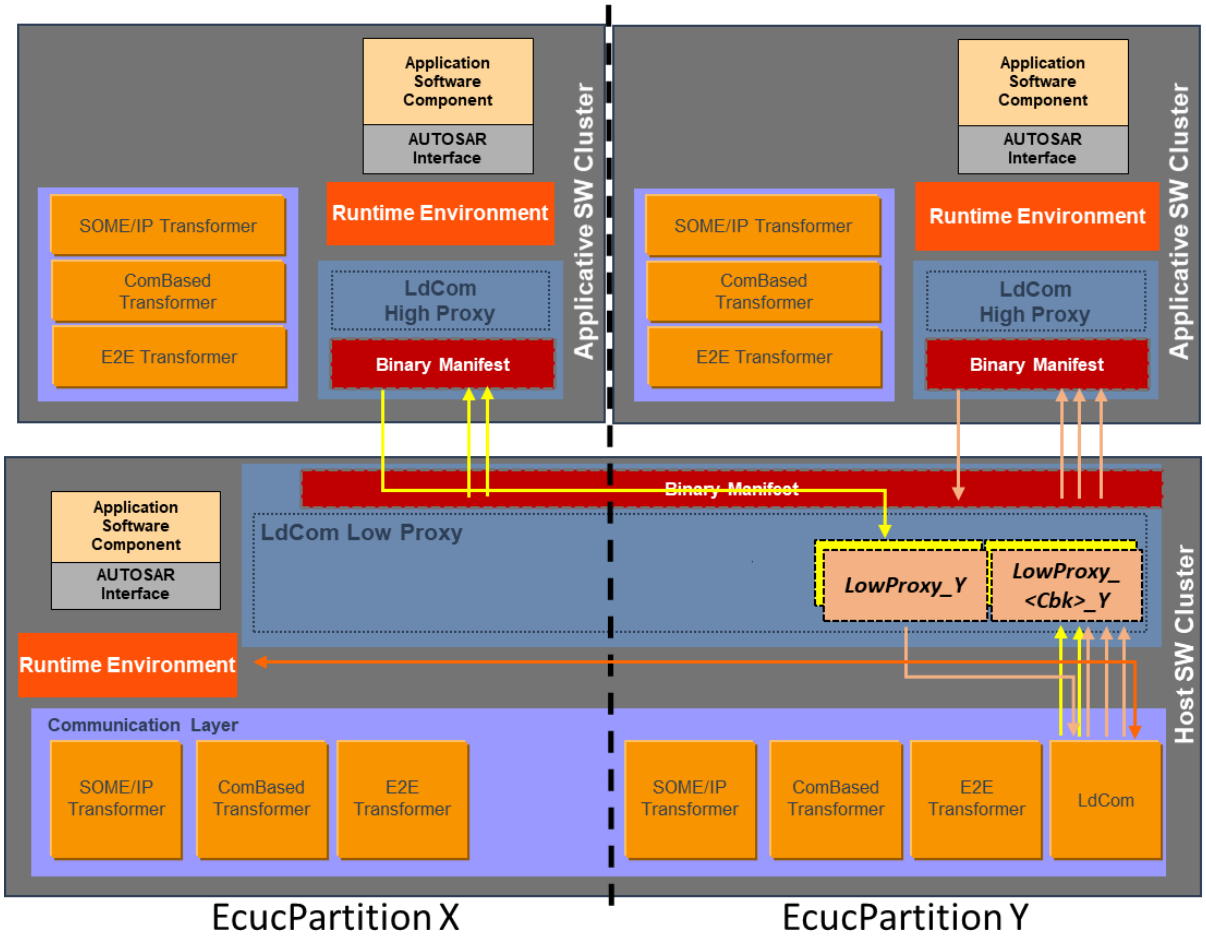

Figure 10 ECUC Partition agnostic LCom user Overview

In this architecture pattern, the LdCom user is partition independent and therefore has to provide one common set of notification callbacks, which are invoked in the context of different partitions. Furthermore, it shall provide a reentrant implementation of the notification callbacks for different LdComIPdus on different ECUC partitions.

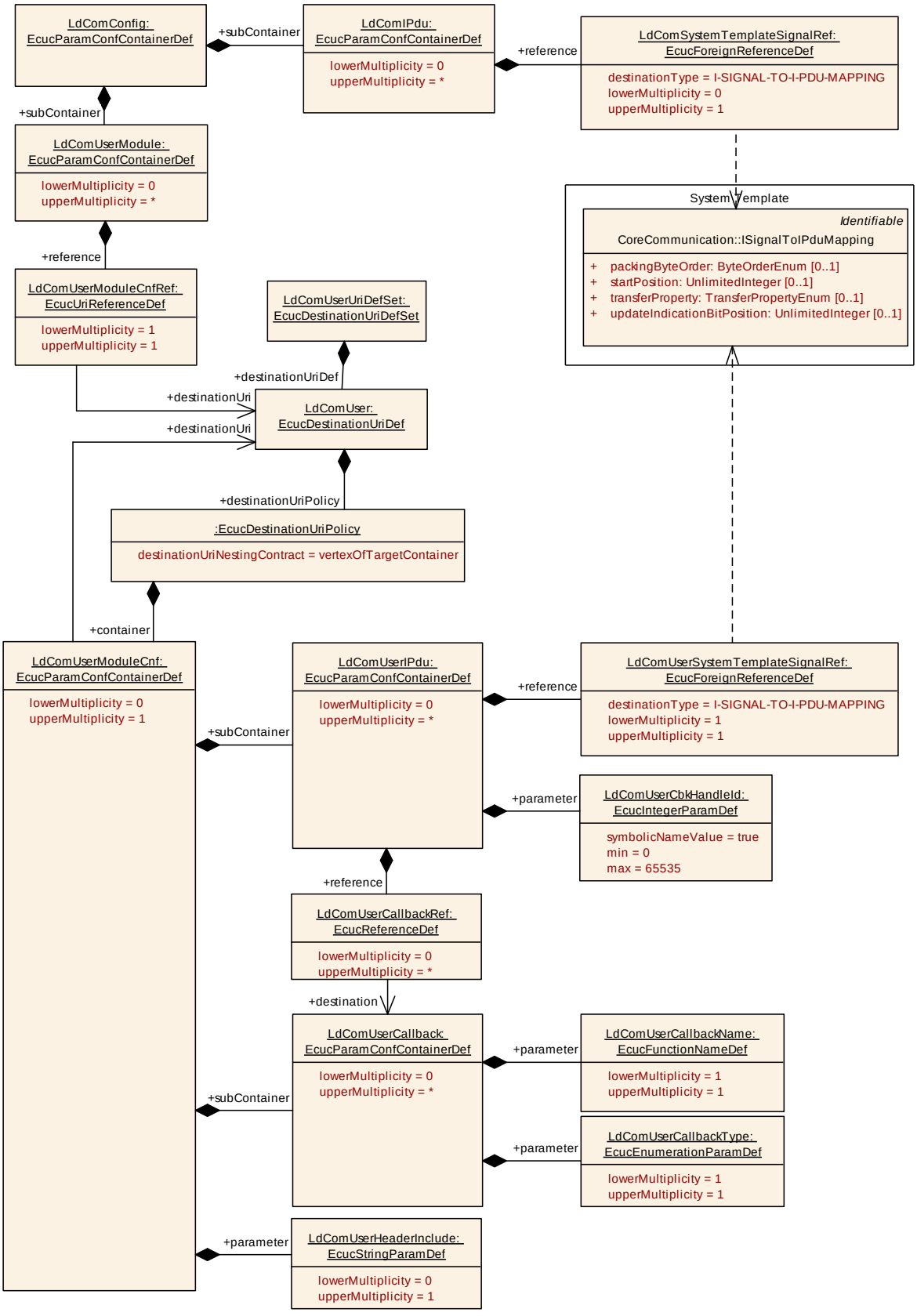

Figure 11 Configuration of the LdCom User Module

10.1.6 LdComUserUriDefSet

SWS Item	**ECUC_LdCom_00034 :**
EcucDestinationUriDefSet Name	*LdComUserUriDefSet*
Description	Defines the set of DestinationUriDefs for the LdCom module.
Included EcucDestinationUriDefs	
Name	***Description***
LdComUser	Defines the configuration container content of the LdCom user modules relevant settings.

SWS Item	**ECUC_LdCom_00035 :**
EcucDestinationUriDef Name	LdComUser
Destination Uri Definition Set	LdComUserUriDefSet
Description	Defines the configuration container content of the LdCom user modules relevant settings.
destinationUri NestingConstract	vertexOfTargetContainer
Configuration Parameters	

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
LdComUserModuleCnf	0..1	Contains the configuration parameters of the LdCom user module. **Tags:** atp.Status=draft

10.1.7 LdComUserModuleCnf

SWS Item	**ECUC_LdCom_00030 :**
Container Name	LdComUserModuleCnf
Parent Container	RteLdComUser, SwCluCLdComProxyBaseSocket
Destination Uri Definition	LdComUser
Description	Contains the configuration parameters of the LdCom user module. **Tags:** atp.Status=draft
Post-Build Variant Multiplicity	false
Configuration Parameters	

SWS Item	**ECUC_LdCom_00027 :**
Name	LdComUserHeaderInclude
Parent Container	LdComUserModuleCnf
Description	Defines the header file where the LdLom user provides the function declarations for configured callbacks. **Tags:** atp.Status=draft
Multiplicity	0..1
Type	EcucStringParamDef
Default value	--
maxLength	--

minLength	--
regularExpression	--
Post-Build Variant Value	false
Scope / Dependency	scope: local

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
LdComUserCallback	0..*	This container defines a LdCom callback function for a LdComIPdu. **Tags:** atp.Status=draft
LdComUserIPdu	0..*	Contains the configuration parameters for the LdCom's signal (LdComIPdu) inside a LdCom user module. **Tags:** atp.Status=draft

Note:
For SwCluC, a LdCom user is represented by one or several SwCluCLdComBaseSockets. A Base Socket is required for each partition, in which the LdCom user

1. requires direct access to the LdCom APIs initiating transmission requests
2. provides notification callbacks w.r.t transmission and reception

Effectively, a Base Socket links a fixed set of notification callbacks in the LdCom to a specific ECUC partition in the Applicative SW Cluster. As consequence, this means:
The LdCom LowProxy has to map each LdComUserIPdu via LdComUserSystemTemplateSignalRef to an LdComIPdu. There is one LdComUserModuleCnf associated to a SwCluCLdComBaseSocket per EcucPartition. This having the effect that there is also a dedicated range of Handle IDs per EcucPartition, easing the check that IDs are uniquely configured for LdComIPdus.

- The LdCom shall provide its APIs for transmission requests of the relevant LdCom IPdus on the ECUC partition configured in the Base Socket. (Please note that a bottom-up approach, where the LdCom configures on which ECUC partitions which LdCom IPdus are provided, is also possible).

- The LdCom High Proxy shall provide a compatible configuration structure and content for the RTE. It derives its configuraton of LdCom IPdus from the LdCom. For the partition assignment, the LdCom High Proxy creates “virtual” main functions (Rx/Tx) and maps the LdCom IPdus to them. These main functions exist only in the configuration but do not have an implementation.

The system must provide the required ECUC partitions in the Application and the Host Cluster. A requirement, which must be considered during system design

10.1.8 LdComUserCallback

SWS Item	**ECUC_LdCom_00022 :**
Container Name	LdComUserCallback

Parent Container	LdComUserModuleCnf
Description	This container defines a LdCom callback function for a LdComIPdu. **Tags:** atp.Status=draft
Post-Build Variant Multiplicity	true
Configuration Parameters	

SWS Item	**ECUC_LdCom_00023 :**
Name	LdComUserCallbackName
Parent Container	LdComUserCallback
Description	The name of the callback function to be called. **Tags:** atp.Status=draft
Multiplicity	1
Type	EcucFunctionNameDef
Default value	--
maxLength	--
minLength	--
regularExpression	--
Post-Build Variant Multiplicity	false
Post-Build Variant Value	false
Scope / Dependency	scope: ECU

SWS Item	**ECUC_LdCom_00025 :**	
Name	LdComUserCallbackType	
Parent Container	LdComUserCallback	
Description	The type of the LdCom callback **Tags:** atp.Status=draft	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	LDCOM_RX_INDICATION	LdComCbkRxIndication callback indicates a received PDU from a lower layer communication interface module. **Tags:** atp.Status=draft
	LDCOM_RX_START_OF_-RECEPTION	LdComCbkStartOfReception callback called at the start of receiving an N-SDU. **Tags:** atp.Status=draft
	LDCOM_TP_COPY_RX_DATA	LdComCbkCopyRxData callback to provide the received data of an I-PDU segment (N-PDU) to the upper layer. **Tags:** atp.Status=draft
	LDCOM_TP_COPY_TX_DATA	LdComCbkCopyTxData callback to acquire the transmit data of an I-PDU segment. **Tags:** atp.Status=draft
	LDCOM_TP_RX_INDICATION	LdComCbkTpRxIndication callback called after an I-PDU has been received via the TP API **Tags:** atp.Status=draft
	LDCOM_TP_TX_CONFIRMATION	LdComCbkTpTxConfirmation callback called after a Signal has been transmitted via the

		TP-API on its network. **Tags:** atp.Status=draft
	LDCOM_TX_CONFIRMATION	LdComCbkTxConfirmation callback which is called when the lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU. **Tags:** atp.Status=draft
	LDCOM_TX_TRIGGER_TRANSMIT	LdComCbkTxConfirmation callback which is called when the lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU. **Tags:** atp.Status=draft
Post-Build Variant Multiplicity	false	
Post-Build Variant Value	false	
Scope / Dependency	scope: ECU	

No Included Containers

10.1.9 LdComUserIPdu

SWS Item	**ECUC_LdCom_00028 :**
Container Name	LdComUserIPdu
Parent Container	LdComUserModuleCnf
Description	Contains the configuration parameters for the LdCom's signal (LdComIPdu) inside a LdCom user module. **Tags:** atp.Status=draft
Post-Build Variant Multiplicity	true
Configuration Parameters	

SWS Item	**ECUC_LdCom_00026 :**
Name	LdComUserCbkHandleId
Parent Container	LdComUserIPdu
Description	The numerical value used as the LdCom user callback handle Id. This is the ID used by LdCom to invoke callbacks of a LdCom user (Rte, ScCluC LdCom Low Proxy or CDDs) using LdComUserCbkHandleId parameter respectively. A corresponding symbolic name reference is created, which may be used for the invocations of the user. **Tags:** atp.Status=draft
Multiplicity	1
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)
Range	0 .. 65535
Default value	--

Post-Build Variant Value	false
Scope / Dependency	scope: ECU

SWS Item	**ECUC_LdCom_00024 :**
Name	LdComUserCallbackRef
Parent Container	LdComUserIPdu
Description	Reference(s) to all callback(s) of this LdComIPdu. **Tags:** atp.Status=draft
Multiplicity	0..*
Type	Reference to [LdComUserCallback]
Post-Build Variant Multiplicity	true
Post-Build Variant Value	true
Scope / Dependency	scope: ECU

SWS Item	**ECUC_LdCom_00033 :**
Name	LdComUserSystemTemplateSignalRef
Parent Container	LdComUserIPdu
Description	Reference to the ISignalToIPduMapping that contains a reference to the ISignal (System Template). **Tags:** atp.Status=draft
Multiplicity	1
Type	Foreign reference to [I-SIGNAL-TO-I-PDU-MAPPING]
Post-Build Variant Multiplicity	true
Post-Build Variant Value	true
Scope / Dependency	scope: ECU

No Included Containers

[SWS_LdCom_CONSTR_00009]{DRAFT}⌈If there exists a LdComUserIPdu with the LdComIPduDirection set to LDCOM_SEND and LdComApiType set to LDCOM_TP which references an ISignal, the respective
- <LdComUser_LdComCbkCopyTxData> and
- <LdComUser_LdComCbkTpTxConfirmation>

Notification callbacks shall be configured too.⌋(SRS_Com_02109, SRS_Com_02114)

[SWS_LdCom_CONSTR_00010]{DRAFT}⌈ If there exists a LdComUserIPdu with the LdComIPduDirection set to LDCOM_RECEIVE and LdComApiType set to LDCOM_TP, the respective
- <LdComUser_LdComCbkStartOfReception>,
- <LdComUser_LdComCbkCopyRxData> and
- <LdComUser_LdComTpRxIndication>

Notification callbacks shall be configured too.⌋(SRS_Com_02109, SRS_Com_02114)

[SWS_LdCom_CONSTR_00011]{DRAFT}⌈ If there exists a LdComUserIPdu with the LdComIPduDirection set to LDCOM_RECEIVE and LdComApiType set to LDCOM_IF, the respective
- <LdComUser_LdComCbkRxIndication>

Notification callback shall be configured too.⌋(SRS_Com_02109, SRS_Com_02114)

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.