

Document Title	Specification of Communication Manager
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	79
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	<ul style="list-style-type: none"> • Introduced dedicated APIs to synchronize the PNC status with Nm and set the usage of ComSignals to obsolete • Introduced ComMChannelPerTxOnlyPnc to support transmission-only PNCs • Set requirements to valid which relates to forward an wake up request if an PNC is actively requested • Re-worked the service interfaces to support the Pn learning phase
2020-11-30	R20-11	AUTOSAR Release Management	<ul style="list-style-type: none"> • Added optional feature for dynamic PNC-to-channel mapping • Added optional handling to transfer kind of communication request (either active or passive) to lower layers • Extend ComM service interface ComM_GetCurrentComMode, to obtain the PNC state of the mapped ComMUser • Added restriction for ComM users according the assignment of managed and managing channels

Document Change History			
Date	Release	Changed by	Change Description
2019-11-28	R19-11	AUTOSAR Release Management	<ul style="list-style-type: none"> • Introduce handling of PNC coordinator if several ComM channels have the same PNC assignment but PncGatewayTypeEnum is set to "none." • Enabled ComM to be used for BSW distribution (multicore use case) • Minor corrections • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	<ul style="list-style-type: none"> • Introduce "managing" and "managed" ComM channels • Remove relations to EcuMfixed completely • Minor corrections
2017-12-08	4.3.1	AUTOSAR Release Management	<ul style="list-style-type: none"> • Clarification regarding communication inhibition and bus wake up inhibition
2016-11-30	4.3.0	AUTOSAR Release Management	<ul style="list-style-type: none"> • Added the possibility to switch ethernet switch ports according to ComM channel request / release • Added the wake up handling in case of a ECU which is controlling a Ethernet switch and using PNCs. • Minor corrections
2015-07-31	4.2.2	AUTOSAR Release Management	<ul style="list-style-type: none"> • Chapter added to explain partial network usecase • Minor corrections
2014-10-31	4.2.1	AUTOSAR Release Management	<ul style="list-style-type: none"> • Release of PNC related FULL_COM request already upon leaving PNC_REQUESTED • Several clarifications • Minor corrections

Document Change History			
Date	Release	Changed by	Change Description
2014-03-31	4.1.3	AUTOSAR Release Management	<ul style="list-style-type: none"> • Max. number of supported PNCs by ComM now 56 • ComM supports VariantPostBuild instead of VariantPostBuildSelectable • Restrictions for PNCs with ComMChannels of ComMNmVariant "PASSIVE"
2013-10-31	4.1.2	AUTOSAR Release Management	<ul style="list-style-type: none"> • Introduced modeling of Service Interfaces in Chapt. 8 • Repair the reset after forcing NO_COM Feature • Editorial changes • Removed chapter(s) on change documentation
2013-03-15	4.1.1	AUTOSAR Administration	<ul style="list-style-type: none"> • ComM allows configuration of arbitrary bus names for Bus SMs • Nm Variant Passive not configurable on individual channels anymore • Assignment of ComMPncId to Nm UserData bits specified
2011-12-22	4.0.3	AUTOSAR Administration	<ul style="list-style-type: none"> • Partial Network Cluster Management • Improved/Corrected illustration of start-up sequences (chap 9) • Forbid assigning ComM users to channels with NmVariant=PASSIVE • Removed re-request of unchanged communication mode in case of mismatch with BusStateManager (ComM901) • Removed remains of DEM error reporting

Document Change History			
Date	Release	Changed by	Change Description
2009-12-18	4.0.1	AUTOSAR Administration	<ul style="list-style-type: none"> • Table for interaction between ComM and NM added • Production error COMM_E_NET_START_IND_CHANNEL removed • Lower range of configuration parameter "ComMMainFunctionPeriod" modified
2010-02-02	3.1.4	AUTOSAR Administration	<ul style="list-style-type: none"> • Changed interaction between ComM and ECU State Manager (EcuM) • Changed interaction between ComM and Diagnostic Communication Manager (DCM) • Added dependencies to new modules Basic Software Mode Manager (BswM) and Ethernet State Manager • Legal disclaimer revised
2008-08-13	3.1.1	AUTOSAR Administration	<ul style="list-style-type: none"> • Legal disclaimer revised
2007-07-24	2.1.18	AUTOSAR Administration	<ul style="list-style-type: none"> • Bus specific error handling (e.g. bus off handling) removed • Control of the actual bus states removed • PDU group handling removed • Initialization of Communication stack removed • Document meta information extended • Small layout adaptations made

Document Change History			
Date	Release	Changed by	Change Description
2007-01-24	2.1.19	AUTOSAR Administration	<ul style="list-style-type: none"> • Changed features • Restart (silent com. -> full com.) now possible even if mode limitation is active • Channel state machine changed • Sequence diagrams changed • New services to upper layers • Mode indication API to RTE changed • New calls to other modules • Usage of channel specific API (EcuM and ComM) to indicate that a communication channel has been woken up and has gone to sleep • API for NM control canged (Nm_PassiveStartUp, Nm_NetworkRequest, Nm_NetworkRelease) • Legal disclaimer revised • Release Notes added • “Advice for users” revised • “Revision Information” added
2005-05-31	1.0	AUTOSAR Administration	<ul style="list-style-type: none"> • Initial Release

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	11
2	Acronyms and definitions.....	12
3	Related documentation	14
3.1	Input documents.....	14
3.2	Related standards and norms.....	16
3.3	Related specification.....	16
4	Constraints and assumptions	17
4.1	Limitations	17
4.2	Applicability to car domains	17
5	Dependencies to other modules	18
5.1	File structure	18
5.2	AUTOSAR Runtime Environment (RTE).....	18
5.3	ECU State Manager (EcuM)	18
5.4	Basic Software Mode Manager (BswM)	19
5.5	NVRAM Manager.....	19
5.6	Diagnostic Communication Manager (DCM).....	19
5.7	LIN State Manager.....	19
5.8	CAN State Manager.....	20
5.9	FlexRay State Manager	20
5.10	Ethernet State Manager.....	20
5.11	Network Management (NM).....	20
5.12	Default Error Tracer (DET)	20
5.13	Communication (COM) – {OBSOLETE}.....	20
6	Requirements traceability	21
7	Functional specification.....	30
7.1	Partial Network Cluster Management.....	33
7.1.1	Overview.....	33
7.1.2	Partial Network Cluster Management Functionality.....	34
7.1.3	ComM PNC state machine.....	36
7.1.4	PNC Gateway.....	51
7.1.5	Dynamic PNC-to-channel-mapping (optional)	56
7.1.6	Partial Networking Configuration Hints	58
7.2	ComM channel state machine	60
7.2.1	ComM managed and managing channels	64
7.2.2	Behavior in state COMM_NO_COMMUNICATION	65
7.2.3	Behaviour in state COMM_SILENT_COMMUNICATION.....	68
7.2.4	Behaviour in state COMM_FULL_COMMUNICATION.....	69
7.3	ComM User to PNC Relations	74
7.4	Extended functionality.....	76
7.4.1	Communication inhibition	77
7.5	Bus communication management	80
7.6	Network management dependencies	81
7.7	Bus error management	82

7.7.1	Network Start Indication	82
7.8	Test support requirements.....	82
7.8.1	Inhibited Full Communication Request Counter	82
7.9	Error classification.....	83
7.9.1	Development errors	83
7.9.2	Runtime Errors	84
7.9.3	Transient Faults.....	84
7.9.4	Production Errors.....	84
7.9.5	Extended Production Errors	84
7.10	Communication Manager Module Services.....	84
7.10.1	Architecture.....	84
7.10.2	Use Cases	85
7.10.3	Specification of Ports and Port Interfaces.....	89
7.10.4	Runnables and Entry points	93
7.11	Multicore Distribution	96
7.12	Non functional requirements.....	96
8	API specification.....	97
8.1	Imported types	97
8.1.1	Standard types	97
8.2	Type definitions	98
8.2.1	ComM_InitStatusType	98
8.2.2	ComM_PncModeType.....	98
8.2.3	ComM_StateType.....	99
8.2.4	ComM_ConfigType.....	99
8.3	Function definitions	100
8.3.1	ComM_Init	100
8.3.2	ComM_DelInit.....	101
8.3.3	ComM_GetStatus	101
8.3.4	ComM_GetInhibitionStatus	102
8.3.5	ComM_RequestComMode.....	103
8.3.6	ComM_GetMaxComMode.....	104
8.3.7	ComM_GetRequestedComMode.....	105
8.3.8	ComM_GetCurrentComMode	105
8.3.9	ComM_GetCurrentPNCComMode.....	106
8.3.10	ComM_GetPncToChannelMapping	108
8.3.11	ComM_UpdatePncToChannelMapping	109
8.3.12	ComM_ResetPncToChannelMapping	111
8.3.13	ComM_PnLearningRequest.....	112
8.3.14	ComM_UpdatePncMembership	113
8.3.15	ComM_PreventWakeUp.....	114
8.3.16	ComM_LimitChannelToNoComMode	115
8.3.17	ComM_LimitECUToNoComMode	116
8.3.18	ComM_ReadInhibitCounter.....	117
8.3.19	ComM_ResetInhibitCounter.....	117
8.3.20	ComM_SetECUGroupClassification	118
8.3.21	ComM_GetVersionInfo.....	119
8.4	Callback notifications	119
8.4.1	AUTOSAR Network Management Interface	119
8.4.2	AUTOSAR Diagnostic Communication Manager Interface	127

8.4.3	AUTOSAR ECU State Manager Interface	128
8.4.4	AUTOSAR ECU State Manager and Basic Software Mode Manager Interface	129
8.4.5	Bus State Manager Interface.....	129
8.4.6	COM Interface (OBSOLETE)	131
8.5	Scheduled functions.....	131
8.5.1	ComM_MainFunction	132
8.6	Expected interfaces	132
8.6.1	Mandatory Interfaces.....	132
8.6.2	Optional Interfaces	135
8.6.3	Configurable Interfaces	136
8.7	Service Interfaces	136
8.7.1	Sender-Receiver-interfaces.....	136
8.7.2	Client-Server-interfaces.....	138
8.7.3	Mode-Switch-Interfaces.....	147
8.7.4	Implementation Data Types	147
8.7.5	Ports	150
8.7.6	ModeDeclarationGroups	153
9	Sequence diagrams	154
9.1	Transmission and Reception start (CAN)	154
9.2	Passive Wake-up (CAN)	155
9.3	Network shutdown (CAN)	156
9.4	Communication request	158
9.5	Synchronized PNC shutdown	159
10	Configuration specification.....	164
10.1	How to read this chapter	164
10.2	Containers and configuration parameters	165
10.2.1	ComM	166
10.2.2	ComMGeneral	166
10.2.3	ComMConfigSet	173
10.2.4	ComMUser.....	174
10.2.5	ComMChannel.....	176
10.2.6	ComMNetworkManagement.....	183
10.2.7	ComMUserPerChannel	185
10.2.8	ComMPnc	186
10.2.9	ComMPncComSignal	189
10.3	Published information	191
11	Not applicable requirements	192

List of Figures

Figure 1: Communication Manager Module context view.....	18
Figure 2: PNC State Machine	38
Figure 3: PNC-to-channel-mapping	51
Figure 4: Example for a PNC gateway with not coordinated ComMChannels (see Node2)	52
Figure 5: Example for a partial network (PN) topology that reflect the hierarchy.....	54
Figure 6: Example for a valid PN with multiple top-level PNC coordinators.....	55
Figure 7: ComM channel state machine	61
Figure 8: User to Partial network and channel Mapping Use Cases.....	74
Figure 9: ARPackages of the Communication Manager Module	85
Figure 10: SW-C requests state changes to the Communication Manager Module.....	86
Figure 11: SW-C requires state changes within the Communication Manager Module and reads out current communication state	87
Figure 12: Interaction between BswM and the ComM module.....	89
Figure 13: Starting transmission and reception on CAN	154
Figure 14: Reaction on a wake-up indicated by the ECU State Manager module.....	155
Figure 15: Network shutdown (CAN)	157
Figure 16: Request Communication	158
Figure 17: Request for a synchronized PNC shutdown in the role of a top-level PNC coordinator (TLPC).....	161
Figure 18: Request to forward a synchronized PNC shutdown in the role of an intermediate PNC coordinator	163
Figure 19: Configuration ComM	166
Figure 20: Configuration ComMGeneral	173
Figure 21: Configuration ComMUser	176
Figure 22: Configuration ComMChannel	182
Figure 23: Configuration ComMNetworkManagement	185
Figure 24: Configuration ComMUserPerChannel and ComUserPerPNC	186
Figure 25: Configuration ComMPnc.....	189

1 Introduction and functional overview

The Communication Manager Module (COM Manager, ComM) is a component of the Basic Software (BSW). It is a Resource Manager, which encapsulates the control of the underlying communication services. The ComM module controls basic software modules relating to communication and not software components or runnable entities. The ComM module collects the bus communication access requests from communication requestors (see definition of term “User” in Chapter 2) and coordinates the bus communication access requests.

The purpose of the ComM module is:

Simplifying the usage of the bus communication stack for the user. This includes a simplified network management handling.

Coordinating the availability of the bus communication stack (allow sending and receiving of signals) of multiple independent software components on one ECU.

Comment: A user should not have any knowledge about the hardware (e.g. on which channel to communicate). A user simply requests a "Communication Mode" and ComM module switches the communication capability of the corresponding channel on/off.

Offer an API to disable sending of signals to prevent the ECU from (actively) waking up the communication bus.

Comment: On CAN every message wakes up the bus, on FlexRay it is only possible to wake up the bus with a so called wake-up pattern.

Controlling of more than one communication bus channel of an ECU by implementing a channel state machine for every channel.

Comment: The ComM module requests a Communication Mode from the corresponding Bus State Manager module. The actual bus states are controlled by the corresponding Bus State Manager module.

Offering the possibility to force an ECU that keeps the bus awake to the ‘No Communication’ mode (see Section 7.4.1.2 for details).

Simplifying the resource management by allocating all resources necessary for the requested Communication Mode.

Comment: E.g. check if communication is allowed when a user requests ‘Full Communication’ mode, and prevent the ECU from shutdown during communication.

Further, the PNC extension allows users to request and keep awake a logical group of ECUs all over the network, a so-called “partial network cluster”. The “PNC gateway” allows to span these (logical) network clusters over different, hierarchically structured physical busses and networks

2 Acronyms and definitions

Abbreviation / Acronym:	Description:
BSW	Basic Software
BswM	Basic Software Mode Manager
ComM	Communication Manager
DCM	Diagnostic Communication Manager
Det	Default Error Tracer
EcuM	ECU State Manager module
I-PDU	Information Protocol Data Unit
NM	Network Management
PDU	Protocol Data Unit
SW-C	Software Component
VMM	Vehicle Message Matrix
OA TC10	Open Alliance TC10 specification (see [33])
IRA	Internal Request Array. This is a bit vector which contains the aggregated internal PNC requests per channel. (see also chapter 8.6.2 "Nm_UpdateIRA")
EIRA	External and Internal Request Array. This is a bit vector which contains the aggregated external and internal PNC requests
ERA	External Request Array. This is a bit vector which contains the aggregated external PNC requests. Each ComMChannel which has a ComMGatewayType set is has one corresponding ERA
ERAn	All External Request Arrays which are available in ComM, i.e. "n" ComMChannels were ComMGatewayType is set, result in "n" External Request Arrays in ComM

Term:	Description:
DCM_ActiveDiagnostic indication	The DCM module indicates an active diagnostic session. DCM need "full communication" = COMM_FULL_COMMUNICATION for diagnostic purpose
Active wake-up	Wake-up caused by the hosting ECU e.g. by a sensor.
Application signal scheduling	Sending of application signals according to the VMM. Scheduling of CAN application signals is performed by the Communication Module, scheduling of LIN application I-PDUs (a PDU containing signals) is performed by the LIN interface and scheduling of FlexRay application PDUs is performed by the FlexRay Interface module.
Bus sleep	No activity required on the communication bus (e.g. CAN bus sleep).
Bus communication messages	Bus communication messages are all messages that are sent on the communication bus. This can be either a diagnostic message or an application message.
COM Inhibition status	Defines whether full communication, silent communication or wake-up is allowed or not.
Communication Channel	The medium used to convey information from a sender (or transmitter) to a receiver.
Communication Mode	Mode determining which kind of communication are allowed: "full communication" = COMM_FULL_COMMUNICATION "no communication" = COMM_NO_COMMUNICATION "silent communication" = COMM_SILENT_COMMUNICATION <i>Note: COMM_SILENT_COMMUNICATION can not be requested by a user. Internal mode for synchronizing network at shutdown</i>
Diagnostic PDU scheduling	Sending of diagnostic PDUs. Scheduling of CAN diagnostic PDUs is performed by the diagnostic module, scheduling of LIN diagnostic PDUs is performed by the diagnostic module and the LIN interface and scheduling of FlexRay diagnostic PDUs is performed by the diagnostic module and the FlexRay Interface module.
ECU shut down	See ECU State Manager specification [6].
Fan-out	Same message/indication are sent to multiple destinations/receivers

Independent software component	A separately developed software component performing a coherent set of functions with a minimum amount of interfaces to other software applications on an ECU. This can be e.g. a basic software component or an application software component.
Passive wake-up	Wake-up by another ECU and propagated (e.g. by bus or wake-up-line) to the ECU currently in focus.
System User	An administration functionality (a specific "user", which is generated within the internal context of the ComM) for making a default request and for overriding the user requests.
User	Concept for requestors of the ECU State Manager module and of the Communication Manager Module. A user may be the BswM, a runnable entity, a SW-C or a group of SW-Cs, which act as a single unit towards the ECU State Manager module and the Communication Manager Module.
User Request	A User can request different Communication Modes from ComM
Managed channel	A ComM channel that is referenced exclusively from one other channel by ECUC parameter ComMManageReference (see ECUC ComM 00893).
Managing channel	A ComM channel that references 1..n other channels by ECUC parameter ComMManageReference (see ECUC ComM 00893).

3 Related documentation

3.1 Input documents

- [1] List of Basic Software Modules
AUTOSAR_TR_BSWModuleList.pdf
- [2] Layered Software Architecture
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf
- [3] General Requirements on Basic Software Modules
AUTOSAR_SRS_BSWGeneral.pdf
- [4] Requirements on Mode Management
AUTOSAR_SRS_ModeManagement.pdf
- [5] Specification of ECU Configuration
AUTOSAR_TPS_ECUConfiguration.pdf
- [6] Specification of ECU State Manager
AUTOSAR_SWS_ECUStateManager.pdf
- [7] Specification of NVRAM Manager
AUTOSAR_SWS_NVRAMManager.pdf
- [8] Specification of RTE Software
AUTOSAR_SWS_RTE.pdf
- [9] Specification of Generic Network Management Interface
AUTOSAR_SWS_NetworkManagementInterface.pdf
- [11] Specification of Diagnostic Communication Manager
AUTOSAR_SWS_DiagnosticCommunicationManager.pdf
- [12] Specification of LIN Interface
AUTOSAR_SWS_LINInterface.pdf
- [13] Specification of FlexRay Interface
AUTOSAR_SWS_FlexRayInterface.pdf
- [14] Specification of Default Error Tracer
AUTOSAR_SWS_DefaultErrorTracer.pdf
- [16] Specification of CAN Transceiver Driver
AUTOSAR_SWS_CANTransceiverDriver.pdf
- [17] Specification of CAN Interface
AUTOSAR_SWS_CANInterface.pdf

[18] Specification of FlexRay Transceiver Driver
AUTOSAR_SWS_FlexRayTransceiver.pdf

[19] Specification of PDU Router
AUTOSAR_SWS_PDURouter.pdf

[20] Requirements on IPDU Multiplexer
AUTOSAR_SWS_IPDUM.pdf

[21] Specification of System Services Mode Management
AUTOSAR_SystemServices_ModeManagement.pdf

[22] Specification of C Implementation Rules
AUTOSAR_Tr_CImplementationRules.pdf

[23] Specification of LIN State Manager
AUTOSAR_SWS_LINStateManager.pdf

[24] Specification of CAN State Manager
AUTOSAR_SWS_CANStateManager.pdf

[25] Specification of FlexRay State Manager
AUTOSAR_SWS_FlexRayStateManager.pdf

[26] Basic Software Module Description Template,
AUTOSAR_TPS_BSWModuleDescriptionTemplate.pdf

[27] Glossary,
AUTOSAR_TR_Glossary.pdf

[28] Specification of Ethernet State Manager
AUTOSAR_SWS_EthernetStateManager.pdf

[29] Specification of Basic Software Mode Manager
AUTOSAR_SWS_BSWModeManager.pdf

[30] General Specification of Basic Software Modules
AUTOSAR_SWS_BSWGeneral.pdf

[31] Specification of System Template
AUTOSAR_TPS_SystemTemplate

[32] Specification of Guide to BSW Distribution
AUTOSAR_EXP_BSWDistributionGuide

3.2 Related standards and norms

[33] OPEN ALIANCE Sleep/Wake-up Specification Version 2.0 (Rel Feb 21, 2017), <http://www.opensig.org/Automotive-Ethernet-Specifications/>

3.3 Related specification

AUTOSAR provides a General Specification on Basic Software modules [30] (SWS BSW General), which is also valid for COM Manager.

Thus, the specification SWS BSW General shall be considered as additional and required specification for COM Manager.

4 Constraints and assumptions

4.1 Limitations

No limitations.

4.2 Applicability to car domains

No restrictions.

5 Dependencies to other modules

A context view which shows the Communication Manager Module and the dependencies to other modules is shown in Figure 1:

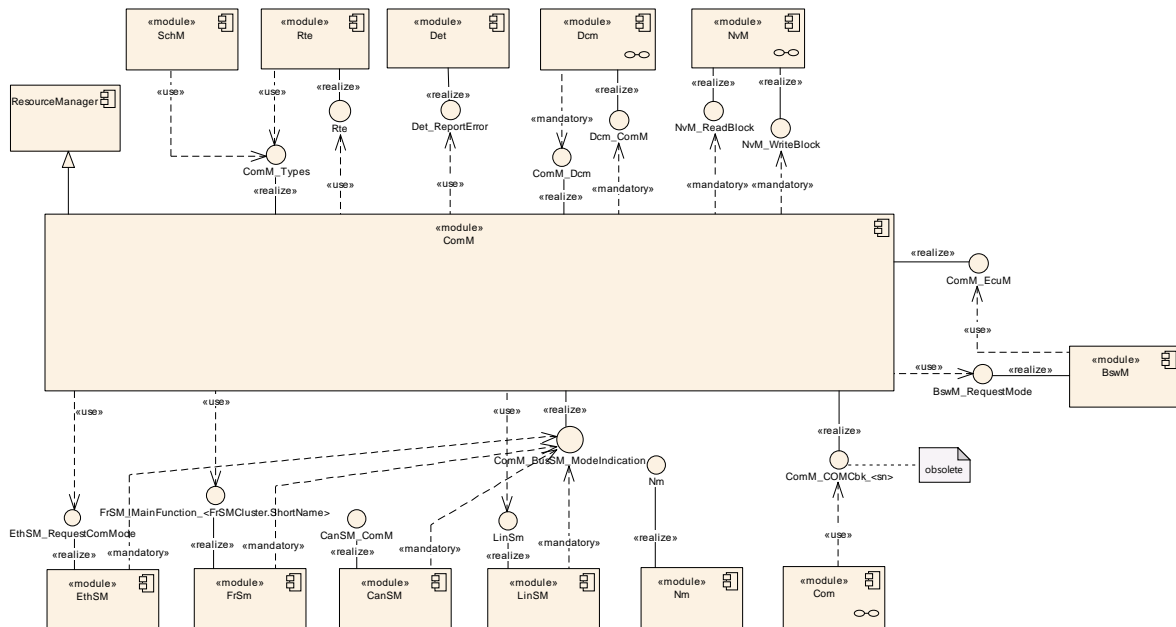


Figure 1: Communication Manager Module context view

The Communication Manager Module requests the communication capabilities, requested from the users, from the Bus State Manager modules.

5.1 File structure

5.2 AUTOSAR Runtime Environment (RTE)

Every user can request a Communication Mode. The RTE propagates the user request to the ComM module and the Communication Mode indications from the ComM to the users (for details refer to [8]).

5.3 ECU State Manager (EcuM)

EcuM is responsible to validate wake-up events and send an indication to ComM if a wake-up is validated.

Communication allowed and shutdown of ECU is handled by EcuM together with BswM. (see [6] for details)

5.4 Basic Software Mode Manager (BswM)

The BswM realizes two functionalities Mode Arbitration and Mode Control to allow the application of an Application Mode Management and a Vehicle Mode Management.

The BswM propagates user requests to the ComM module, if configured in the action lists of BswM to be able to request ComM modes via BswM.

The BswM controls the PDU Groups in the AUTOSAR Communication Module (COM), if the call of `Com_IpduGroupControl` is configured in the action list.

[SWS_ComM_00976] [ComM indicates all channel main state changes and all PNC state changes to the BswM.](SRS_ModeMgm_09251)

If EcuM-Flex is used, BswM will indicate to ComM if communication is allowed or not.

5.5 NVRAM Manager

The ComM module uses the NVRAM Manager to store and read non-volatile data. For details on initial values of the NVRAM data refer to Chapter 10.

Comment: The NVRAM Manager must be initialized after a power up or reset of the ECU. It must be initialized before ComM, as when ComM is initialized, ComM assumes that NVRAM is ready to be used, and that it can read back non-volatile configuration data. When ComM is de-initialized, it writes non-volatile data to NVRAM.

5.6 Diagnostic Communication Manager (DCM)

The DCM performs the scheduling of diagnostic PDUs. The DCM acts as a user by requesting Communication Mode `COMM_FULL_COMMUNICATION` via a “DCM_ActiveDiagnostic” indication if diagnostics shall be performed. The DCM does not provide an API to start/stop sending and receiving but guarantees that the communication capabilities are according to the ComM module Communication Modes.

5.7 LIN State Manager

The LIN State Manager controls the actual states of the LIN bus that correspond to a Communication Mode of the ComM module. The ComM module requests a Communication Mode from the LIN State Manager and the LIN State Manager maps the Communication Mode to a bus state.

5.8 CAN State Manager

The CAN State Manager controls the actual states of the CAN bus that correspond to a Communication Mode of the ComM module. The ComM module requests a Communication Mode from the CAN State Manager and the CAN State Manager maps the Communication Mode to a bus state.

5.9 FlexRay State Manager

The FlexRay State Manager controls the actual states of the FlexRay bus that correspond to a Communication Mode of the ComM module. The ComM module requests a Communication Mode from the FlexRay State Manager and the FlexRay State Manager maps the Communication Mode to a bus state.

5.10 Ethernet State Manager

The Ethernet State Manager controls the actual states of the Ethernet bus that correspond to a Communication Mode of the ComM module. The ComM module requests a Communication Mode from the Ethernet State Manager and the Ethernet State Manager maps the Communication Mode to a bus state.

5.11 Network Management (NM)

The ComM module uses the NM to synchronize the control of communication capabilities across the network (synchronous start-up and shutdown). Additionally the status information about PNCs is exchanged via dedicated APIs between ComM and Nm.

5.12 Default Error Tracer (DET)

The DET provides services for reporting development, runtime, and transient errors. (see Section 7.9)

5.13 Communication (COM) – {OBSOLETE}

[SWS_ComM_00975] {OBSOLETE} [The AUTOSAR Communication module (COM) shall be used to distribute the status information about PNCs using COM signals.]()

6 Requirements traceability

Requirement	Description	Satisfied by
SRS_BSW_00004	All Basic SW Modules shall perform a pre-processor check of the versions of all imported include files	SWS_ComM_00418
SRS_BSW_00005	Modules of the μ C Abstraction Layer (MCAL) may not have hard coded horizontal interfaces	SWS_ComM_00499
SRS_BSW_00009	All Basic SW Modules shall be documented according to a common standard.	SWS_ComM_00499
SRS_BSW_00010	The memory consumption of all Basic SW Modules shall be documented for a defined configuration for all supported platforms.	SWS_ComM_00499
SRS_BSW_00101	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	SWS_ComM_00146
SRS_BSW_00158	-	SWS_ComM_00464
SRS_BSW_00161	The AUTOSAR Basic Software shall provide a microcontroller abstraction layer which provides a standardized interface to higher software layers	SWS_ComM_00499
SRS_BSW_00162	The AUTOSAR Basic Software shall provide a hardware abstraction layer	SWS_ComM_00499
SRS_BSW_00164	The Implementation of interrupt service routines shall be done by the Operating System, complex drivers or modules	SWS_ComM_00499
SRS_BSW_00167	All AUTOSAR Basic Software Modules shall provide configuration rules and constraints to enable plausibility checks	SWS_ComM_00419

SRS_BSW_00168	SW components shall be tested by a function defined in a common API in the Basis-SW	SWS_ComM_00499
SRS_BSW_00170	The AUTOSAR SW Components shall provide information about their dependency from faults, signal qualities, driver demands	SWS_ComM_00499
SRS_BSW_00314	All internal driver modules shall separate the interrupt frame definition from the service routine	SWS_ComM_00499
SRS_BSW_00323	All AUTOSAR Basic Software Modules shall check passed API parameters for validity	SWS_ComM_00234
SRS_BSW_00325	The runtime of interrupt service routines and functions that are running in interrupt context shall be kept short	SWS_ComM_00499
SRS_BSW_00327	Error values naming convention	SWS_ComM_00234
SRS_BSW_00336	Basic SW module shall be able to shutdown	SWS_ComM_00147
SRS_BSW_00337	Classification of development errors	SWS_ComM_00234
SRS_BSW_00341	Module documentation shall contains all needed informations	SWS_ComM_00499
SRS_BSW_00342	It shall be possible to create an AUTOSAR ECU out of modules provided as source code and modules provided as object code, even mixed	SWS_ComM_00459
SRS_BSW_00343	The unit of time for specification and configuration of Basic SW modules shall be preferably in physical time unit	SWS_ComM_00499
SRS_BSW_00344	BSW Modules shall support link-time configuration	SWS_ComM_00499
SRS_BSW_00348	All AUTOSAR standard types and constants	SWS_ComM_00820

	shall be placed and organized in a standard type header file	
SRS_BSW_00353	All integer type definitions of target and compiler specific scope shall be placed and organized in a single type header	SWS_ComM_00499
SRS_BSW_00357	For success/failure of an API call a standard return type shall be defined	SWS_ComM_00820
SRS_BSW_00358	The return type of init() functions implemented by AUTOSAR Basic Software Modules shall be void	SWS_ComM_00146
SRS_BSW_00361	All mappings of not standardized keywords of compiler specific scope shall be placed and organized in a compiler specific type and keyword header	SWS_ComM_00499
SRS_BSW_00373	The main processing function of each AUTOSAR Basic Software Module shall be named according the defined convention	SWS_ComM_00429
SRS_BSW_00375	Basic Software Modules shall report wake-up reasons	SWS_ComM_00499
SRS_BSW_00378	AUTOSAR shall provide a boolean type	SWS_ComM_00499
SRS_BSW_00385	List possible error notifications	SWS_ComM_00234
SRS_BSW_00386	The BSW shall specify the configuration for detecting an error	SWS_ComM_00234
SRS_BSW_00398	The link-time configuration is achieved on object code basis in the stage after compiling and before linking	SWS_ComM_00499
SRS_BSW_00404	BSW Modules shall support post-build configuration	SWS_ComM_00499
SRS_BSW_00405	BSW Modules shall support multiple configuration sets	SWS_ComM_00499

SRS_BSW_00406	A static status variable denoting if a BSW module is initialized shall be initialized with value 0 before any APIs of the BSW module is called	SWS_ComM_00242, SWS_ComM_00612, SWS_ComM_00858
SRS_BSW_00407	Each BSW module shall provide a function to read out the version information of a dedicated module implementation	SWS_ComM_00370
SRS_BSW_00413	An index-based accessing of the instances of BSW modules shall be done	SWS_ComM_00499
SRS_BSW_00414	Init functions shall have a pointer to a configuration structure as single parameter	SWS_ComM_00146
SRS_BSW_00416	The sequence of modules to be initialized shall be configurable	SWS_ComM_00499
SRS_BSW_00417	Software which is not part of the SW-C shall report error events only after the DEM is fully operational.	SWS_ComM_00499
SRS_BSW_00422	Pre-de-bouncing of error status information is done within the DEM	SWS_ComM_00499
SRS_BSW_00423	BSW modules with AUTOSAR interfaces shall be describable with the means of the SW-C Template	SWS_ComM_00499
SRS_BSW_00424	BSW module main processing functions shall not be allowed to enter a wait state	SWS_ComM_00499
SRS_BSW_00425	The BSW module description template shall provide means to model the defined trigger conditions of schedulable objects	SWS_ComM_00499
SRS_BSW_00426	BSW Modules shall ensure data consistency of data which is shared between BSW modules	SWS_ComM_00499
SRS_BSW_00427	ISR functions shall be defined and documented in the BSW	SWS_ComM_00499

	module description template	
SRS_BSW_00428	A BSW module shall state if its main processing function(s) has to be executed in a specific order or sequence	SWS_ComM_00499
SRS_BSW_00429	Access to OS is restricted	SWS_ComM_00499
SRS_BSW_00432	Modules should have separate main processing functions for read/receive and write/transmit data path	SWS_ComM_00499
SRS_BSW_00433	Main processing functions are only allowed to be called from task bodies provided by the BSW Scheduler	SWS_ComM_00499
SRS_BSW_00437	Memory mapping shall provide the possibility to define RAM segments which are not to be initialized during startup	SWS_ComM_00499
SRS_BSW_00438	Configuration data shall be defined in a structure	SWS_ComM_00499
SRS_BSW_00439	Enable BSW modules to handle interrupts	SWS_ComM_00499
SRS_BSW_00441	Naming convention for type, macro and function	SWS_ComM_00863
SRS_BSW_00459	It shall be possible to concurrently execute a service offered by a BSW module in different partitions	SWS_ComM_01019, SWS_ComM_01020, SWS_ComM_01021, SWS_ComM_01059
SRS_ModeMgm_00049	The Communication Manager shall initiate the wake-up and keep awake physical channels	SWS_ComM_00869, SWS_ComM_00870, SWS_ComM_00929, SWS_ComM_01069, SWS_ComM_01071, SWS_ComM_01086
SRS_ModeMgm_09071	It shall be possible to limit communication modes independently for each physical channel	SWS_ComM_00066, SWS_ComM_00215, SWS_ComM_00303, SWS_ComM_00355, SWS_ComM_00740, SWS_ComM_00744, SWS_ComM_00745, SWS_ComM_00752, SWS_ComM_00800, SWS_ComM_00801, SWS_ComM_00842
SRS_ModeMgm_09078	The Communication Manager shall coordinate multiple communication requests	SWS_ComM_00582, SWS_ComM_00686, SWS_ComM_00736, SWS_ComM_00744, SWS_ComM_00745, SWS_ComM_00848

SRS_ModeMgm_09080	Each physical channel shall be controlled by an independent communication mode	SWS_ComM_00051, SWS_ComM_00744, SWS_ComM_00745
SRS_ModeMgm_09081	The Communication Manager shall provide an API allowing collecting communication requests	SWS_ComM_00110
SRS_ModeMgm_09083	The Communication Manager shall support two communication modes for each physical channel	SWS_ComM_00485, SWS_ComM_00845, SWS_ComM_00846, SWS_ComM_00866, SWS_ComM_00867, SWS_ComM_00868, SWS_ComM_00879, SWS_ComM_00880, SWS_ComM_00881, SWS_ComM_00897
SRS_ModeMgm_09084	The Communication Manager shall provide an API which allows application to query the current communication mode	SWS_ComM_00083, SWS_ComM_00734, SWS_ComM_00744, SWS_ComM_00745
SRS_ModeMgm_09085	The Communication Manager shall provide an indication of communication mode changes	SWS_ComM_00091, SWS_ComM_00472, SWS_ComM_00663, SWS_ComM_00733, SWS_ComM_00778, SWS_ComM_00847
SRS_ModeMgm_09087	The Minimum duration of communication request after wakeup shall be configurable	SWS_ComM_00893, SWS_ComM_00894
SRS_ModeMgm_09089	The Communication Manager shall be able to prevent waking up physical channels	SWS_ComM_00157, SWS_ComM_00302, SWS_ComM_00747, SWS_ComM_00799
SRS_ModeMgm_09090	Relationship between users and physical channels shall be configurable at pre compile time	SWS_ComM_00662, SWS_ComM_00795, SWS_ComM_00796, SWS_ComM_00798, SWS_ComM_00995, SWS_ComM_01025
SRS_ModeMgm_09132	It shall be possible to assign Network Management to physical channels	SWS_ComM_00288, SWS_ComM_00599, SWS_ComM_00602, SWS_ComM_00667
SRS_ModeMgm_09133	It shall be possible to assign physical channels to the Communication Manager	SWS_ComM_00995
SRS_ModeMgm_09149	The Communication Manager shall provide an API for querying the requested communication mode	SWS_ComM_00079, SWS_ComM_00374, SWS_ComM_00744, SWS_ComM_00745, SWS_ComM_01022, SWS_ComM_01023, SWS_ComM_01024
SRS_ModeMgm_09155	The Communication Manager shall provide a	SWS_ComM_00138, SWS_ComM_00140, SWS_ComM_00141, SWS_ComM_00142,

	counter for inhibited communication requests	SWS_ComM_00625, SWS_ComM_00803, SWS_ComM_00962
SRS_ModeMgm_09156	It shall be provided an API to retrieve the number of inhibited "Full Communication" mode requests	SWS_ComM_00108, SWS_ComM_00143, SWS_ComM_00224, SWS_ComM_00802
SRS_ModeMgm_09157	It shall be possible to revoke a communication mode limitation, independently for each physical channel	SWS_ComM_00124, SWS_ComM_00156, SWS_ComM_00163, SWS_ComM_00744, SWS_ComM_00745
SRS_ModeMgm_09168	The Communication Manager shall support users that are connected to no physical channel	SWS_ComM_00664, SWS_ComM_00744, SWS_ComM_00745
SRS_ModeMgm_09172	It shall be possible to evaluate the current communication mode	SWS_ComM_00176, SWS_ComM_00744, SWS_ComM_00745
SRS_ModeMgm_09207	ComM shall allow for additional bus specific state managers	SWS_ComM_00957
SRS_ModeMgm_09243	The Communication Manager shall be able to handle the Partial Networks on Flexray, CAN and Ethernet	SWS_ComM_00825, SWS_ComM_00827, SWS_ComM_00910, SWS_ComM_00911, SWS_ComM_00926, SWS_ComM_00953, SWS_ComM_00979, SWS_ComM_00980, SWS_ComM_00982, SWS_ComM_00987, SWS_ComM_01077, SWS_ComM_01080
SRS_ModeMgm_09246	The communication manager shall arbitrate and coordinate requests from users on physical channel and users on PNCs	SWS_ComM_00151, SWS_ComM_00500, SWS_ComM_00827, SWS_ComM_00877, SWS_ComM_00932, SWS_ComM_00948, SWS_ComM_00972, SWS_ComM_00991, SWS_ComM_01025, SWS_ComM_01075, SWS_ComM_01087, SWS_ComM_01088
SRS_ModeMgm_09247	For each configured PNC an independent state machine shall be instantiated	SWS_ComM_00907, SWS_ComM_00909, SWS_ComM_00920, SWS_ComM_00924, SWS_ComM_00978, SWS_ComM_01087, SWS_ComM_01088
SRS_ModeMgm_09248	it shall be possible to distinguish between internal and external PNC activation requests	SWS_ComM_00694, SWS_ComM_00940, SWS_ComM_01014, SWS_ComM_01015, SWS_ComM_01060, SWS_ComM_01061, SWS_ComM_01062, SWS_ComM_01065, SWS_ComM_01068, SWS_ComM_01072, SWS_ComM_01085, SWS_ComM_01087, SWS_ComM_01088, SWS_ComM_01089, SWS_ComM_91028, SWS_ComM_91029
SRS_ModeMgm_09250	PNC activation requests shall be exchanged with the Network Management via a PNC bit vector	SWS_ComM_01060, SWS_ComM_01061, SWS_ComM_01062, SWS_ComM_01079, SWS_ComM_01080, SWS_ComM_01081, SWS_ComM_01085, SWS_ComM_01092, SWS_ComM_01093, SWS_ComM_91028, SWS_ComM_91029
SRS_ModeMgm_09251	PNC communication	SWS_ComM_00908, SWS_ComM_00976

	state shall be forwarded to the BswM	
SRS_ModeMgm_09256	PNC Gateway Functionality shall consider systems with more than one gateways connected to the same network	SWS_ComM_01073, SWS_ComM_01074, SWS_ComM_01076, SWS_ComM_01077, SWS_ComM_01078, SWS_ComM_01079, SWS_ComM_01080, SWS_ComM_01081, SWS_ComM_01084
SRS_ModeMgm_09257	ComM shall forward PNC-Clusters also to busses that are currently not awake	SWS_ComM_01066
SRS_ModeMgm_09258	Optional Dynamic Extension of PNC Gateway	SWS_ComM_01034, SWS_ComM_01037, SWS_ComM_01041, SWS_ComM_01044, SWS_ComM_01047, SWS_ComM_01091
SRS_ModeMgm_09259	ComM API shall provide interfaces to access PNC Mapping (optional)	SWS_ComM_01035, SWS_ComM_01036, SWS_ComM_01038, SWS_ComM_01039, SWS_ComM_01040, SWS_ComM_01042, SWS_ComM_01043, SWS_ComM_91013, SWS_ComM_91015, SWS_ComM_91017, SWS_ComM_91102, SWS_ComM_91105, SWS_ComM_91107
SRS_ModeMgm_09260	ComM API shall provide an interface to start PNC Learning mechanism for PNC Mapping (optional)	SWS_ComM_01026, SWS_ComM_01045, SWS_ComM_01046, SWS_ComM_01048, SWS_ComM_01049, SWS_ComM_01058, SWS_ComM_91019, SWS_ComM_91101, SWS_ComM_91106
SRS_ModeMgm_09261	ComM shall forward the information for Partial Networking Learning (optional)	SWS_ComM_01028, SWS_ComM_01090, SWS_ComM_01093, SWS_ComM_91026
SRS_ModeMgm_09262	ComM shall set all its assigned PNCs when partial networking learning is requested (optional)	SWS_ComM_01092
SRS_ModeMgm_09263	ComM API shall provide an interface to set PNC-membership on Host-ECU (optional)	SWS_ComM_91021, SWS_ComM_91100
SRS_ModeMgm_09265	ComM shall send the information for Partial Networking Learning (optional)	SWS_ComM_01029, SWS_ComM_91024
SRS_ModeMgm_09266	ComM shall support communication channels that act as communication slaves with wake-up capability	SWS_ComM_01017, SWS_ComM_01018, SWS_ComM_CONSTR_00003
SRS_ModeMgm_09267	ComM shall support communication channels which act as communication slaves without wake-up	SWS_ComM_00915, SWS_ComM_01018

	capability	
SRS_ModeMgm_09268	ComM shall support the possibility to forward the information if the communication request is active or passive to it's lower layer layer	SWS_ComM_00069, SWS_ComM_01056, SWS_ComM_01057, SWS_ComM_01067, SWS_ComM_01070, SWS_ComM_01071
SRS_ModeMgm_09269	The Communication Manager shall support synchronized PNC shutdown	SWS_ComM_01051, SWS_ComM_01082, SWS_ComM_01083, SWS_ComM_91003, SWS_ComM_91030
SRS_ModeMgm_09270	The ECU State Manager shall provide a service for the selection of the shutdown target	SWS_ComM_00991, SWS_ComM_01066, SWS_ComM_01072, SWS_ComM_01073, SWS_ComM_01074, SWS_ComM_01075, SWS_ComM_01076, SWS_ComM_01077, SWS_ComM_01078, SWS_ComM_01079, SWS_ComM_01080, SWS_ComM_01081, SWS_ComM_01084
SRS_ModeMgm_09278	The Communication Manager shall support synchronous and asynchronous request upon a indicated wakeup	SWS_ComM_00990, SWS_ComM_01063, SWS_ComM_01064
SRS_ModeMgm_09279	The Communication Manager shall support a coordinated release of PNCs	SWS_ComM_00947, SWS_ComM_00952
SRS_ModeMgm_09280	The Communication Manager shall support to act as a top-level PNC coordinator per PNC	SWS_ComM_01083

7 Functional specification

The Communication Manager (ComM) module simplifies the resource management for the users, whereat users may be runnable entities, SW-Cs, the BswM (e.g. SW-C request via BswM) or DCM (communication needed to diagnostic purpose).

[SWS_ComM_00867] [The ComM shall provide three different Communication Modes. The highest Communication Mode shall be `COMM_FULL_COMMUNICATION`. The lowest Communication Mode shall be `COMM_NO_COMMUNICATION`.](SRS_ModeMgm_09083)

[SWS_ComM_00151] [For a user it shall only be possible to request the Communication Modes `COMM_NO_COMMUNICATION` and `COMM_FULL_COMMUNICATION` (see `ComM_RequestComMode()`, [SWS_ComM_00110](#)).](SRS_ModeMgm_09246)

Rationale for [SWS_ComM_00151](#):

1. The Communication Mode `COMM_SILENT_COMMUNICATION` and sub-modes/sub-states are only necessary for **synchronization** with AUTOSAR NM.
2. The Communication Mode `COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST` is only necessary to **request the lower layer** to trigger a wake-up on the network (e.g. Ethernet hardware compliant to OA TC10 [see 33]). This mode could not be requested by a user.

[SWS_ComM_00868] [The Communication Mode `COMM_SILENT_COMMUNICATION` shall only be used for network synchronization.](SRS_ModeMgm_09083)

Note: The possibility to request `COMM_SILENT_COMMUNICATION` mode is removed since release 2.0.

Comment:

- The ComM module allows querying the Communication Mode requested by a particular user (see `ComM_GetRequestedComMode()`, [SWS_ComM_00079](#)).
- The ComM module allows querying the actual Communication Mode of a channel if the user is assigned to channel(see `ComM_GetCurrentComMode()`, [SWS_ComM_00083](#))
- The ComM module allows querying for the current PNC mode if the user is assigned to a PNC (see `ComM_GetCurrentPNCComMode()`, [SWS_ComM_91002](#))

[SWS_ComM_00845] [In `COMM_FULL_COMMUNICATION` mode, the ComM module shall allow transmission and reception on the affected physical channel.](SRS_ModeMgm_09083)

[SWS_ComM_00846] [In `COMM_NO_COMMUNICATION` mode, the ComM module shall prevent transmission and reception on the affected physical channel.](SRS_ModeMgm_09083)

[SWS_ComM_00686] [If at least one of multiple independent user requests demands a higher Communication Mode (see [SWS_ComM_00867](#) and [SWS_ComM_00868](#)), the ComM module shall set this higher Communication Mode as the target Communication Mode.](SRS_ModeMgm_09078)

Rationale for [SWS_ComM_00686](#): ComM coordinates multiple independent user requests according to the "highest wins" strategy: `COMM_FULL_COMMUNICATION` Communication Mode overrules `COMM_NO_COMMUNICATION`.

[SWS_ComM_00500] [The ComM module shall not queue user requests. The latest user request of the same user shall overwrite an old user request even if the request is not finished.](SRS_ModeMgm_09246)

[SWS_ComM_00866] [If configuration parameter `ComMNmVariant=FULL|LIGHT|NONE` ([ECUC_ComM_00568](#)), an `DCM_ActiveDiagnostic` indication shall be treated as a `COMM_FULL_COMMUNICATION` request for the specified communication channel (see `ComM_DCM_ActiveDiagnostic(channel)`, [SWS_ComM_00873](#)).](SRS_ModeMgm_09083)

Rationale for [SWS_ComM_00866](#): If more channels needed for diagnostic purpose, DCM needs to indicate `DCM_ActiveDiagnostic` for each channel.

[SWS_ComM_00092] [There shall be one Communication Mode target state (evaluated according to [SWS_ComM_00686](#)) per communication channel. This target mode can differ temporarily from the actual mode controlled by the corresponding Bus State Manager module.](SRS_ModeMgm_09078)

Comment: Mode switching by the corresponding Bus State Manager module takes time and a mode inhibition can be active.

[SWS_ComM_00084] [The ComM module shall propagate a call of `ComM_GetCurrentComMode()` (see [SWS_ComM_00083](#)) to the Bus State Manager module(s) for the channel(s) the user are configured to (see also [SWS_ComM_00176](#) and [SWS_ComM_00798](#)).](SRS_ModeMgm_09078)

Rationale for [SWS_ComM_00084](#): State requests have to be propagated to the corresponding Bus State Manager module since the ComM module does not control the actual bus state.

Comment: This feature is not used by a "normal SW-C" because they don't have knowledge about channels. This feature is necessary for privileged SW-Cs, which (have to) know about the system topology, e.g. system diagnostic functions.

[SWS_ComM_00884] [The ComM module shall store status if communication for a channel is allowed or not allowed in separate `CommunicationAllowed` boolean

flags for all supported channels. The default value after ComM initialization shall be communication is not allowed, i.e. `CommunicationAllowed` is set to `FALSE`.]()

[SWS_ComM_00885] [Status changes for communication allowed or not allowed in [SWS_ComM_00884](#) shall be provided to ComM in `ComM_CommunicationAllowed(<channel>, TRUE|FALSE)` ([SWS_ComM_00871](#)) indications.]()

7.1 Partial Network Cluster Management

The ComM offers users the option to wake and keep awake so-called “partial network cluster” (PNC). A PNC is a (logical) group of ECUs which have to be active at the same time to realize some distributed functionality. If PNC-enabled gateways are used, a PNC can span the whole network (different busses on different topology levels of the network hierarchy). Without the PN functionality, NM messages can only wake and keep awake whole busses.

7.1.1 Overview

ComM implements a state machine for each partial network cluster (PNC) to represent the communication mode of a PNC.

Each PNC has its own state. The state definitions are related to the states of ComM for a simple mapping.

ComM users are used to request and release the PNCs.

The status of all PNCs on the nodes of a system channel is exchanged within the so-called PNC bit vector via a network management message (NM message).

Each PNC uses a dedicated bit position within a bit vector (PNC bit vector) transferred by a NM message on CAN, FlexRay and Ethernet. If a PNC is requested by a local ComM user on the node, the node sets the corresponding PNC bit in the PNC bit vector to 1. If the PNC is not requested anymore; the node sets the corresponding PNC bit in the PNC bit vector to 0. (Please note: If the optional feature “Synchronized PNC shutdown” is used (see chapter 7.1.4.4) and a PN shutdown messages has to be transmitted, the PNC bits are set to 1 for the PNCs which are released and the remaining PNC bits are set to 0). The <Bus>Nm extract the PNC bit vector from a received NM-PDU and forward the PNC bit vector to the NM interface. The NM collect and aggregate PNC requests. Based on the aggregated PNC requests, the NM interface handle the timer for each PNC and provide the aggregated PNC state to ComM via dedicated APIs.

Each PNC uses the same bit position in the PNC bit vector on every system channel within the NM message. ComM uses 3 types of PNC bit vector named EIRA, ERA and IRA to exchange PNC status information with NM interface and <Bus>Nms. The definition of “EIRA” and “ERA” are located in the AUTOSAR SWS CAN NM, AUTOSAR SWS FlexRay NM and AUTOSAR SWS UDP NM. For definition of “IRA” please refer to chapter 2.

ComM requests and releases the system communication bus channels needed for a PNC on a node.

Enabling or disabling the partial network cluster management in the node shall be post-build configurable.

Partial networking is supported on the bus types CAN, FlexRay and Ethernet. Activation and deactivation of the I-PDU groups of the PNCs on a Can, FlexRay and Ethernet node is required to avoid false timeouts. Starting and Stopping of I-PDU

groups in COM are handled in BswM. Deactivation of single FlexRay ECUs is not possible.

7.1.2 Partial Network Cluster Management Functionality

[SWS_ComM_00910] [PNC functionality shall only exist if the parameter `ComMPncSupport` is set to `TRUE`. (see [ECUC ComM_00839](#)).](SRS_ModeMgm_09243)

[SWS_ComM_00911] [Enabling or disabling of the PNC functionality shall be post-build configurable using the parameter `ComMPncEnabled` (see [ECUC ComM_00878](#)).](SRS_ModeMgm_09243)

Comment: The ComM module notifies the BswM about every state change of the PNC state machine by calling `BswM_ComM_CurrentPncMode()`. (refer to [SWS ComM_00908](#))

[SWS_ComM_00982] [For exchanging PNC status information between ComM and Nm, bit vectors shall be used. The PNC bit vector is named “PN info” and contain a maximum of 504 bits.](SRS_ModeMgm_09243)

Comment: The PNC bit vector is provided as a reference to an array of type `uint8` to the ComM within the dedicated APIs. Each bit in the PNC bit vector represents the status of a particular PNC. The bit is called “PNC bit”.

[SWS_ComM_00825] [The `byteIndex` and `bitIndex`, in which a PNC bit corresponding to one `ComMPncId` resides, shall be determined as follows:

- $\text{byteIndex} = (\text{ComMPncId} \div 8) - \langle \text{PNC Vector Offset} \rangle$
- $\text{bitIndex} = (\text{ComMPncId} \bmod 8)$](SRS_ModeMgm_09243)

Hint: The value of the PNC bit vector length of the corresponding channel can be obtained from the configuration of the Network Management module.

Comment: SWS_ComM_00825 defines only the calculation of the `byteIndex` and `bitIndex`, not how it shall be implemented.

[SWS_ComM_00984] **{OBSOLETE replaced by [SWS_ComM_01060]}** [ComM receives the bit vectors (signals) which can be `ComMPncComSignalKind` EIRA or ERA using `Com_ReceiveSignal()`.]()

[SWS_ComM_00986] **{OBSOLETE replaced by [SWS_ComM_01061]}** [The ComM shall provide the API `ComM_COMCbk_<sn>()` to indicate a change of signal(s) within the module communication.]()

[SWS_ComM_00916] **{OBSOLETE replaced by [SWS_ComM_01062]}** [The ComM module shall be able to distribute the status of a PNC (result of the PNC state machine) via one or more communications busses using one or more COM signals ,as a bit vector, containing a bit which represents the status of the PNC with

ComMPncComSignalDirection “TX” assigned to this PNC. (For more details, refer to Figure 2: PNC State Machine))()

ComM receives the aggregated state of internal and external PNC requests as PNC bit vector via the callback function `ComM_Nm_UpdateEIRA(<PNC bit vector of internal and external PNC requests>)`.

[SWS_ComM_01060] {DRAFT} [If `ComM_Nm_UpdateEIRA(<PNC bit vector of EIRA>)` is called, then ComM shall transfer the content of the given PNC bit vector to the EIRA of ComM with respect to the PNC bit vector length configured in `Nmlf`.](`SRS_ModeMgm_09248`, `SRS_ModeMgm_09250`)

ComM receives the aggregated state of external PNC requests as PNC bit vector per channel via the callback function `ComM_Nm_UpdateERA(<Channel>, <PNC bit vector of external PNC requests>)`.

[SWS_ComM_01061] {DRAFT} [If the configuration parameter `ComMPncGatewayEnabled` (see `ECUC_ComM_00887`) is set to `TRUE`, `ComM_Nm_UpdateERA(<channel>, <PNC bit vector of ERA>)` is called and the parameter `ComMPncGatewayType` is set for the given channel, then ComM shall transfer the content of the given PNC bit vector to the ERA of ComM with respect to the given channel and the PNC bit vector length configured in `Nmlf`.](`SRS_ModeMgm_09248`, `SRS_ModeMgm_09250`)

Note:

- ComM transfers the EIRA PNC bit vector provided by Nm in one internal EIRA (see **[SWS_ComM_01060]**) and each ERA PNC bit vector in one ERA per `ComMChannel` (see **[SWS_ComM_01061]**)
- Transferring the content of a PNC bit vector result in the internal EIRA / ERA of ComM by setting the PNC bit in the internal EIRA / ERA to ‘1’ if the corresponding PNC bit in the PNC bit vector is set to ‘1’ or setting the PNC bit in the internal EIRA / ERA to ‘0’ if the corresponding PNC bit in the PNC bit vector is set to ‘0’

[SWS_ComM_01062] {DRAFT} [The ComM module shall be able to distribute the status of a particular PNC (result of the PNC state machine) across the assigned ComM channels. Therefore ComM shall forward the aggregated state of internal PNC request per communication channel (e.g. bus or network) as PNC bit vector by calling the API `Nm_UpdateIRA(<channel>, <PNC bit vector of aggregated internal PNC requests>)`. The IRA PNC bit vector designates the status of the internal PNC requests.](`SRS_ModeMgm_09248`, `SRS_ModeMgm_09250`)

Note:

- The meaning of the PNC bits is defined in **[SWS_ComM_00825]**
- Internal PNC requests are based on ComM user PNC requests and/or PNC requests, due to PNC gateway handling

7.1.3 ComM PNC state machine

[SWS_ComM_00953] [If the PNC functionality is enabled using the configuration parameter `ComMPncEnabled` set to TRUE (see [ECUC ComM 00878](#)), all actions related to PNC changes shall be executed before the channel related actions (channel related actions, see Chapter 7.3).](SRS_ModeMgm_09243)

[SWS_ComM_00909] [For every Partial Network Cluster, only one PNC state machine shall be implemented (i.e. one PNC state machine per PNC, independent of the amount of `ComMChannels`).](SRS_ModeMgm_09247)

[SWS_ComM_00920] [The ComM module shall support up to 504 PNC state machines.](SRS_ModeMgm_09247)

[SWS_ComM_00924] [The PNC state machine shall consist of the two main states `COMM_PNC_FULL_COMMUNICATION` and `COMM_PNC_NO_COMMUNICATION`.](SRS_ModeMgm_09247)

[SWS_ComM_00907] [The PNC main state `COMM_PNC_FULL_COMMUNICATION` shall consist of the sub states `COMM_PNC_PREPARE_SLEEP`, `COMM_PNC_READY_SLEEP` and `COMM_PNC_REQUESTED`.](SRS_ModeMgm_09247)

[SWS_ComM_00908] [Every state change (listed within the `ComM_PncModeType`), excluding entering of the main state `COMM_PNC_NO_COMMUNICATION` coming from `PowerOff`, shall be notified by the API call `BswM_ComM_CurrentPncMode()` with the entered PNC state.](SRS_ModeMgm_09251)

[SWS_ComM_00978] [State transitions of the PNC state machines in ComM, triggered by a call to `ComM_RequestComMode()` shall be executed in the `ComM_MainFunction_<Channel.ShortName>` only.](SRS_ModeMgm_09247)

Comment: Every PNC activation triggers sending of the PNC bit vector n-times, thus it would increase the busload without debouncing.

[SWS_ComM_00944] **{OBSOLETE replaced by [SWS_ComM_01060]}** [If at least one bit corresponding to the PNC within the Rx bitvectors with signal type "EIRA" equals '1', then the bit corresponding to this PNC within EIRA in ComM shall be set to '1'.]()

[SWS_ComM_00945] **{OBSOLETE replaced by [SWS_ComM_01061]}** [If the configuration parameter `ComMPncGatewayEnabled` (see [ECUC ComM 00840](#)) is true and the parameter `ComMPncGatewayType` is set for a `ComMChannel` and at least one bit corresponding to the PNC within the Rx bitvectors with signal type "ERA" equals '1', then the bit corresponding to this PNC within ERA in ComM shall be set to '1'.]()

[SWS_ComM_00971] {OBSOLETE} [The trigger `ComM_COMCbk` represents a notification by the AUTOSAR Communication module about a received signal containing PNC status information called ERA of EIRA.]()

[SWS_ComM_00972] [The trigger "ComMUser" represents a notification about a communication request of a ComMUser by calling the API `ComM_RequestComMode()`.](SRS_ModeMgm_09246)

[SWS_ComM_00987] [Within the `ComM_MainFunction_<Channel.ShortName>` of a channel that is mapped to one or more PNCs, the requested state shall be handled in the following order:

1. ComM user requests of ComM users mapped to one or more PNCs of that channel
2. ComM user requests of ComM users mapped to that channel
3. ERA (if the configuration switch `ComMPncGatewayEnabled` is set to TRUE)
4. EIRA

](SRS_ModeMgm_09243)

Comment: Requests are handled in main functions of those channels they affect.

[SWS_ComM_00919] {OBSOLETE} [It shall be possible to assign more than one COM signal containing bits representing the PNC to one PNC using the configuration container `ComMPncComSignal` (see [ECUC ComM_00881](#)).]()

[SWS_ComM_00827] [Regarding "Communication allowed" and mode inhibitions, requests originating from a PNC state machine shall be treated like user requests for the according channels.](SRS_ModeMgm_09243, SRS_ModeMgm_09246)

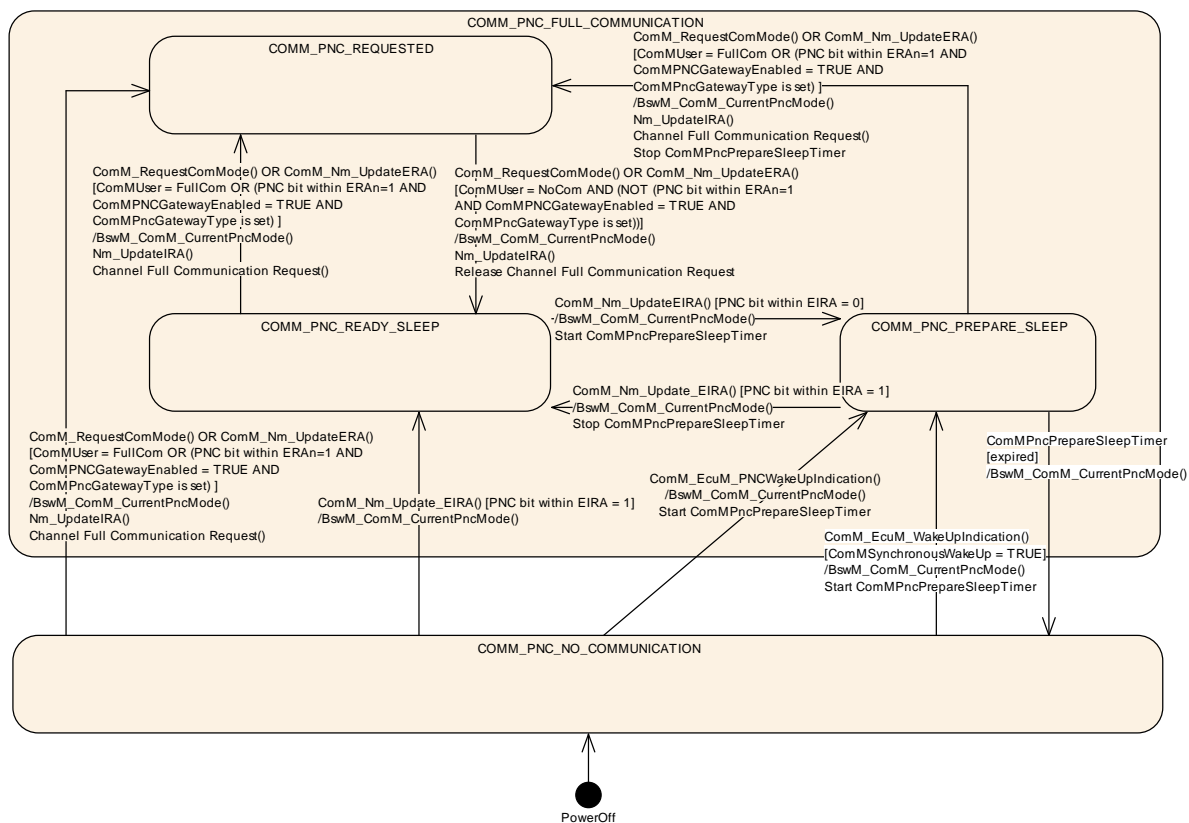


Figure 2: PNC State Machine

7.1.3.1 Behavior in PNC main state COMM_PNC_NO_COMMUNICATION

[SWS_ComM_00926] [The PNC main state COMM_PNC_NO_COMMUNICATION shall be the default PNC state from power off.](SRS_ModeMgm_09243)

The main state COMM_PNC_NO_COMMUNICATION is the target state as long as the PNC is neither requested ECU internally nor requested externally.

[SWS_ComM_00931] {OBOSLETE replaced by [SWS_ComM_01063]} [If the API `ComM_EcuM_WakeUpIndication()` is called in PNC state COMM_PNC_NO_COMMUNICATION, and the configuration switch `ComMSynchronousWakeUp` is set to TRUE (see [ECUC ComM_00695](#)), the PNC main state COMM_PNC_NO_COMMUNICATION shall be left and the PNC sub state COMM_PNC_PREPARE_SLEEP shall be entered.]()

[SWS_ComM_01063] {DRAFT} [If the API `ComM_EcuM_WakeUpIndication()` is called in PNC state COMM_PNC_NO_COMMUNICATION, the configuration switch `ComMSynchronousWakeUp` is set to TRUE (see [ECUC ComM_00695](#)) and for all PNCs that reference at least one `ComMChannel` via `ComMChannelPerPnc` (see [ECUC ComM_00880](#) :), the PNC main state COMM_PNC_NO_COMMUNICATION shall be left and the PNC sub state COMM_PNC_PREPARE_SLEEP shall be entered.](SRS_ModeMgm_09278)

[SWS_ComM_00990] [If the API `ComM_EcuM_WakeUpIndication()` is called in PNC state `COMM_PNC_NO_COMMUNICATION`, and the configuration switch `ComMSynchronousWakeUp` is set to `FALSE`, the PNC main state `COMM_PNC_NO_COMMUNICATION` shall be the current state.](SRS_ModeMgm_09278)

Comment: In case of asynchronous wake up, the PNC state shall stay in `COMM_PNC_NO_COMMUNICATION` until the PNC request is received (PNC bit in EIRA is set to '1').

[SWS_ComM_00964] {OBOSLETE replaced by [SWS_ComM_01064]} [If the API `ComM_EcuM_PNCWakeUpIndication(<PNC>)` (see [SWS ComM 91001](#)) is called in PNC state `PNC_NO_COMMUNICATION`, the PNC main state `PNC_NO_COMMUNICATION` shall be left and the PNC sub state `PNC_PREPARE_SLEEP` shall be entered.]()

[SWS_ComM_01064] {DRAFT} [If the API `ComM_EcuM_PNCWakeUpIndication(<PNC>)` (see [SWS ComM 91001](#)) is called in PNC state `PNC_NO_COMMUNICATION` and the indicated PNC reference at least one `ComMChannel` via `ComMChannelPerPnc` (see **ECUC_ComM_00880** :), the PNC main state `PNC_NO_COMMUNICATION` shall be left and the PNC sub state `PNC_PREPARE_SLEEP` shall be entered.](SRS_ModeMgm_09278)

[SWS_ComM_00932] [When at least one `ComMUser` assigned to this PNC requests "Full Communication" in PNC main state `COMM_PNC_NO_COMMUNICATION`, this state shall be left and the sub state `COMM_PNC_REQUESTED` of the main state `COMM_PNC_FULL_COMMUNICATION` shall be entered.](SRS_ModeMgm_09246)

[SWS_ComM_00933] {OBOSLETE replaced by [SWS_ComM_01065]} [When in main state `COMM_PNC_NO_COMMUNICATION` at least one PNC bit representing this PNC in EIRA changes to '1', the main state `COMM_PNC_NO_COMMUNICATION` shall be left and the `COMM_PNC_READY_SLEEP` shall be entered.]()

[SWS_ComM_01065] {DRAFT} [When in main state `COMM_PNC_NO_COMMUNICATION` at least one PNC bit representing this PNC in EIRA changes to '1' and this PNC reference at least one `ComMChannel` via `ComMChannelPerPnc` (see **ECUC_ComM_00880** :), the main state `COMM_PNC_NO_COMMUNICATION` shall be left and the `COMM_PNC_READY_SLEEP` shall be entered.](SRS_ModeMgm_09248)

7.1.3.1.1 PNC gateway related requirements

[SWS_ComM_00934] {OBOSLETE replaced by [SWS_ComM_01066]} [When in main state `COMM_PNC_NO_COMMUNICATION` at least one bit representing this PNC in an `ERAn` changes to '1', the main state `COMM_PNC_NO_COMMUNICATION` shall be left and the sub state `COMM_PNC_REQUESTED` shall be entered if the parameter

ComMPncGatewayEnabled ([ECUC_ComM_00840](#)) set to TRUE and ComMPncGatewayType is set for the according ComMChannels of these ERAs.](

[SWS_ComM_01066] {DRAFT} [When in main state COMM_PNC_NO_COMMUNICATION at least one PNC bit representing this PNC in ERAn changes to '1', then the main state COMM_PNC_NO_COMMUNICATION shall be left and the sub state COMM_PNC_REQUESTED shall be entered under the following conditions:

- the parameter ComMPncGatewayEnabled (see **ECUC_ComM_00887** :) is set to TRUE
- this PNC references at least one channel via ComMChannelPerPnc (see **ECUC_ComM_00880** :) and the referenced channels have the ComMPncGatewayType set

](SRS_ModeMgm_09257, SRS_ModeMgm_09270)

7.1.3.2 On entry of PNC main state COMM_PNC_NO_COMMUNICATION from PowerOff

[SWS_ComM_00927] {OBSOLETE} [After switching on the power supply, main state COMM_PNC_NO_COMMUNICATION shall be entered from PowerOff.](

Note: After switching on the power supply, main state COMM_PNC_NO_COMMUNICATION is entered from PowerOff (see **[SWS_ComM_00926]**)

7.1.3.3 Behavior in PNC main state COMM_PNC_FULL_COMMUNICATION

[SWS_ComM_00929] [As long as a specific PNC is in state COMM_PNC_FULL_COMMUNICATION all ComMChannels which are referenced by this PNC via ComMChannelPerPnc (see **ECUC_ComM_00880** :) shall be in COMM_FULL_COMMUNICATION.](SRS_ModeMgm_00049)

7.1.3.4 On entry of PNC sub state COMM_PNC_REQUESTED

[SWS_ComM_01053] {OBSOLETE replaced by [SWS_ComM_01067]} [When entering the PNC sub state COMM_PNC_REQUESTED from COMM_PNC_NO_COM or COMM_PNC_PREPARE_SLEEP, and ComMPncWakeupSleepRequestEnabled of this PNC is set to TRUE, BswM_ComM_CurrentPNCMode shall be called with COMM_PNC_REQUESTED_WITH_WAKEUP_REQUEST, instead of calling BswM_ComM_CurrentPNCMode with COMM_PNC_REQUESTED.](

[SWS_ComM_01067] {DRAFT} [When entering the PNC sub state COMM_PNC_REQUESTED from COMM_PNC_NO_COM or COMM_PNC_PREPARE_SLEEP, this PNC reference at least one ComMChannel via ComMChannelPerPnc (see **ECUC_ComM_00880** :) and ComMPncWakeupSleepRequestEnabled of this PNC is set to TRUE, BswM_ComM_CurrentPNCMode shall be called with COMM_PNC_REQUESTED_WITH_WAKEUP_REQUEST, instead of calling

BswM_ComM_CurrentPNCMode with COMM_PNC_REQUESTED.](
SRS_ModeMgm_09268)

Note: Notification towards the BswM with COMM_PNC_REQUESTED_WITH_WAKEUP_REQUEST is used for Ethernet switch port switching to trigger a wake-up on the network where the used Ethernet hardware is compatible to the OA TC10 (see[33])

[SWS_ComM_00930] {OBSOLETE replaced by [SWS_ComM_01068]} [When entering the PNC sub state COMM_PNC_REQUESTED and if ComMPncGatewayEnabled = FALSE or on all ComMChannels the PNC belongs to the ComMPncGatewayType is not set, the API Com_SendSignal() shall be called with the value '1' for the bit representing this PNC for the Com signal assigned to this PNC with ComMPncComSignalDirection "TX".]()

[SWS_ComM_01068] {DRAFT} [When entering the PNC sub state COMM_PNC_REQUESTED, then the ComM module shall set the PNC bit with value '1' of the PNC bit representing this PNC within the IRA and forward the aggregated internal PNC requests to each channel which is referenced this PNC by calling Nm_UpdateIRA(<channel>, <IRA>) under either of the following conditions:

- ComMPncGatewayEnabled is set to FALSE
- ComMPncGatewayType is not set on any of the ComMChannels referenced by this PNC

](SRS_ModeMgm_09248)

[SWS_ComM_00993] {OBSOLETE replaced by [SWS_ComM_01069]} [Every time the sub state COMM_PNC_REQUESTED is entered from other states, ComM shall request COMM_FULL_COMMUNICATION for all configured ComM channels for this PNC, where ComMWakeupSleepRequestEnabled is set to FALSE or not available, even if the channel is already requested.]()

[SWS_ComM_01069] {DRAFT} [Every time the sub state COMM_PNC_REQUESTED is entered from other states, ComM shall request COMM_FULL_COMMUNICATION for all configured ComM channels which are referenced by this PNC via parameter ComMChannelPerPnc (see **ECUC_ComM_00880** :) and where ComMWakeupSleepRequestEnabled is set to FALSE or not available, even if the channel is already requested.](SRS_ModeMgm_00049)

[SWS_ComM_01054] {OBSOLETE replaced by [SWS_ComM_01070]} [Every time the sub state COMM_PNC_REQUESTED is entered from COMM_PNC_NO_COM or COMM_PNC_PREPARE_SLEEP, ComM shall request COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST for all configured ComM channels for this PNC, where ComMWakeupSleepRequestEnabled is set to TRUE, even if the channel is already requested.]()

[SWS_ComM_01070] {DRAFT} [Every time the sub state COMM_PNC_REQUESTED is entered from COMM_PNC_NO_COM or COMM_PNC_PREPARE_SLEEP, ComM shall request

COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST for all configured ComM channels which are referenced by this PNC via parameter ComMChannelPerPnc (see **ECUC_ComM_00880** :) and where ComMWakeupSleepRequestEnabled is set to TRUE, even if the channel is already requested.](SRS_ModeMgm_09268)

[SWS_ComM_01055] {OBSOLETE replaced by [SWS_ComM_01071]} [Every time the sub state COMM_PNC_REQUESTED is entered from COMM_PNC_READY_SLEEP, ComM shall request COMM_FULL_COMMUNICATION for all configured ComM channels for this PNC, where ComMWakeupSleepRequestEnabled is set to TRUE, even if the channel is already requested.](SRS_ModeMgm_09268)

[SWS_ComM_01071] {DRAFT} [Every time the sub state COMM_PNC_REQUESTED is entered from COMM_PNC_READY_SLEEP, ComM shall request COMM_FULL_COMMUNICATION REQUEST for all configured ComM channels which are referenced by this PNC via parameter ComMChannelPerPnc (see **ECUC_ComM_00880** :) and where ComMWakeupSleepRequestEnabled is set to TRUE, even if the channel is already requested.](SRS_ModeMgm_09268, SRS_ModeMgm_00049)

Comment on [SWS_ComM_01071]: Entering from COMM_PNC_READY_SLEEP should not result in a wake-up on the network, since the PNC is already requested remotely by another ECU

7.1.3.4.1 PNC gateway related requirements

[SWS_ComM_00992] {OBSOLETE replaced by [SWS_ComM_01072]} [When entering the PNC sub state COMM_PNC_REQUESTED and if ComMPncGatewayEnabled = TRUE, the API Com_SendSignal() shall be called with the value '1' for the bits representing this PNC for the Com signals assigned to this PNC with ComMPncComSignalDirection "TX" for all ComM channels related to this PNC and having ComMPncGatewayType == COMM_GATEWAY_TYPE_ACTIVE.]()

[SWS_ComM_01072] {DRAFT} [When entering the PNC sub state COMM_PNC_REQUESTED and ComMPncGatewayEnabled is set to TRUE, then ComM shall set the PNC bit with value '1' of the PNC bit representing this PNC within the IRA on all referenced ComMChannels where ComMPncGatewayType is set to COMM_GATEWAY_TYPE_ACTIVE and forward the aggregated internal PNC request accordingly to those ComMChannels by calling Nm_UpdateIRA(<channel>, <IRA>)](SRS_ModeMgm_09270, SRS_ModeMgm_09248)

7.1.3.5 Behavior in PNC sub state COMM_PNC_REQUESTED

[SWS_ComM_00938] [When all ComMUsers assigned to this PNC request "No Communication", the sub state COMM_PNC_REQUESTED shall be left and the sub state COMM_PNC_READY_SLEEP shall be entered, if ComMPncGatewayEnabled

is set to FALSE or `ComMPncGatewayType` is not set on all channels which are referenced by this PNC.](())

Note: As long as at least one `ComMUser` assigned to this PNC requests "Full Communication", `COMM_PNC_REQUESTED` will be the current PNC state. Please refer to the following requirements.

[SWS_ComM_01073] {DRAFT} [In sub state `COMM_PNC_REQUESTED` when `ComMPncGatewayEnabled` is set to FALSE and at least one `ComMUser` assigned to a specific PNC requests "Full Communication", then `ComM` shall request `COMM_FULL_COMMUNICATION` of those `ComMChannels` which are referenced via `ComMChannelPerTxOnlyPnc` by this PNC.](SRS_ModeMgm_09270, SRS_ModeMgm_09256)

[SWS_ComM_01074] {DRAFT} [In sub state `COMM_PNC_REQUESTED` when `ComMPncGatewayEnabled` is set to FALSE and all `ComMUsers` assigned to a specific PNC requests "No Communication", then `ComM` shall request `COMM_NO_COMMUNICATION` of those `ComMChannels` which are referenced via `ComMChannelPerTxOnlyPnc` by this PNC.](SRS_ModeMgm_09270, SRS_ModeMgm_09256)

7.1.3.5.1 PNC gateway related requirements

[SWS_ComM_00991] [The sub state `COMM_PNC_REQUESTED` shall be left and the sub state `COMM_PNC_READY_SLEEP` shall be entered under the following conditions:

- all `ComMUsers` assigned to this PNC request "No Communication"
- the parameter `ComMPncGatewayEnabled` is set to TRUE
- at least one `ComMChannel` is referenced via `ComMChannelPerPnc` (see **ECUC_ComM_00880** :) by this PNC
- all `ComMChannels` referenced by this PNC have `ComMPncGatewayType` parameter set
- the PNC bit representing this PNC equals to '0' in ERAn

](SRS_ModeMgm_09246, SRS_ModeMgm_09270)

[SWS_ComM_01075] {DRAFT} [The sub state `COMM_PNC_REQUESTED` shall be left and the sub state `COMM_PNC_READY_SLEEP` shall be entered under the following conditions:

- all `ComMUsers` assigned to this PNC request "No Communication"
- the parameter `ComMPncGatewayEnabled` is set to TRUE
- all `ComMChannels` referenced by this PNC have `ComMPncGatewayType` parameter NOT set

](SRS_ModeMgm_09246, SRS_ModeMgm_09270)

[SWS_ComM_00164] {OBSOLETE replaced by [SWS_ComM_01076] and [SWS_ComM_01077]} [If `ComMPncGatewayEnabled` == TRUE and either on entering `COMM_PNC_REQUESTED` or within `COMM_PNC_REQUESTED`, the API `Com_SendSignal()` shall be called with the value "1" for the bit representing this PNC for the `Com` signal assigned to this PNC with `ComMPncComSignalDirection`

== "TX" on all ComM channels related to this PNC where ComMPncGatewayType == COMM_GATEWAY_TYPE_PASSIVE
if at least one ComM user assigned to this PNC request "Full Com" or at least one ComMPncComSignal received by Com_ReceiveSignal() from a channel where the signal attributes ComMPncComSignalDirection == "RX" and ComMPncComSignalKind == "ERA" and the channel attribute ComMPncGatewayType == "COMM_GATEWAY_TYPE_ACTIVE" is set to "1".](SRS_ModeMgm_09243)

[SWS_ComM_01076] {DRAFT} [In sub state COMM_PNC_REQUESTED when ComMPncGatewayEnabled is set to TRUE and at least one ComMUser assigned to a specific PNC requests "Full Communication", then ComM shall set the PNC bit representing this specific PNC to value '1' within the IRA of those ComMChannels

- which have ComMPncGatewayType parameter set to COMM_GATEWAY_TYPE_PASSIVE and
- referenced either via ComMChannelPerPnc or via ComMChannelPerTxOnlyPnc by this PNC,

and forward the updated IRA with a call of Nm_UpdateIRA(<channel>, <IRA>).
](SRS_ModeMgm_09270, SRS_ModeMgm_09256)

[SWS_ComM_01077] {DRAFT} [In sub state COMM_PNC_REQUESTED when ComMPncGatewayEnabled is set to TRUE and the PNC bit representing a specific PNC equals to '1' in at least one ERA, whose corresponding ComMChannel has the ComMPncGatewayType parameter set to COMM_GATEWAY_TYPE_ACTIVE, then ComM shall set the PNC bit representing this specific PNC to value '1' within the IRA of those ComMChannels

- which have ComMPncGatewayType parameter set to COMM_GATEWAY_TYPE_PASSIVE and
- referenced via ComMChannelPerPnc or via ComMChannelPerTxOnlyPnc by this PNC,

and forward the updated IRA with a call of Nm_UpdateIRA(<channel>, <IRA>).
](SRS_ModeMgm_09270, SRS_ModeMgm_09256)

[SWS_ComM_01078] {DRAFT} [In sub state COMM_PNC_REQUESTED when ComMPncGatewayEnabled is set to TRUE and at least one ComMUser assigned to a specific PNC requests "Full Communication", then ComM shall request COMM_FULL_COMMUNICATION of those ComMChannels which are referenced via ComMChannelPerTxOnlyPnc by this PNC.](SRS_ModeMgm_09270, SRS_ModeMgm_09256)

[SWS_ComM_00959] {OBSOLETE replaced by [SWS_ComM_01079] and [SWS_ComM_01080]} [If ComMPncGatewayEnabled == TRUE and within COMM_PNC_REQUESTED, the API Com_SendSignal() shall be called with the value "0" for the bit representing this PNC for the Com signal assigned to this PNC with ComMPncComSignalDirection == "TX" on all ComM channels related to this PNC where ComMPncGatewayType == COMM_GATEWAY_TYPE_PASSIVE if all ComM users assigned to this PNC request "No Com" and all ComMPncComSignals received by Com_ReceiveSignal() from a channel where the signal attributes

`ComMPncComSignalDirection == "RX" and ComMPncComSignalKind == "ERA" and the channel attribute ComMPncGatewayType == "COMM_GATEWAY_TYPE_ACTIVE" are set to "0".`](SRS_ModeMgm_09243)

[SWS_ComM_01079] {DRAFT} [In sub state `COMM_PNC_REQUESTED` when `ComMPncGatewayEnabled` is set to `TRUE`, if

- all `ComMUsers` assigned to a specific PNC request "No Communication" and
- the PNC bit representing this specific PNC equals to '0' in ERA, whose corresponding `ComMChannel` has the `ComMPncGatewayType` parameter set to `COMM_GATEWAY_TYPE_ACTIVE`,

then `ComM` shall set the PNC bit representing this specific PNC to value '0' within the IRA of those `ComMChannels`

- which have `ComMPncGatewayType` parameter set to `COMM_GATEWAY_TYPE_PASSIVE` and
- which are referenced via `ComMChannelPerPnc` or via `ComMChannelPerTxOnlyPnc` by this PNC,

and forward the updated IRA with a call of `Nm_UpdateIRA(<channel>, <IRA>)`.] (SRS_ModeMgm_09270, SRS_ModeMgm_09256, SRS_ModeMgm_09250)

[SWS_ComM_01080] {DRAFT} [In sub state `COMM_PNC_REQUESTED` when `ComMPncGatewayEnabled` is set to `TRUE`, if

- all `ComMUsers` assigned to a specific PNC request "No Communication" and
- the `ComMChannels` which are referenced by this PNC have the `ComMPncGatewayType` parameter not set,

then `ComM` shall set the PNC bit representing this specific PNC to value '0' within the IRA of all `ComMChannels` which are referenced by this PNC and forward the updated IRA with a call of `Nm_UpdateIRA(<channel>, <IRA>)`.] (SRS_ModeMgm_09270, SRS_ModeMgm_09256, SRS_ModeMgm_09250)

[SWS_ComM_01081] {DRAFT} [In sub state `COMM_PNC_REQUESTED` when `ComMPncGatewayEnabled` is set to `TRUE` and all `ComMUsers` assigned to a specific PNC request "No Communication", then `ComM` shall request `COMM_NO_COMMUNICATION` of those `ComMChannels` which are referenced via `ComMChannelPerTxOnlyPnc` by this PNC.] (SRS_ModeMgm_09270, SRS_ModeMgm_09256, SRS_ModeMgm_09250)

[SWS_ComM_01050] {OBSOLETE replaced by [SWS_ComM_01082]} [If a request to forward a synchronized PNC shutdown has been indicated via a call of `ComM_Nm_ForwardSynchronizedPncShutdown(<channel>)`, the indicated channel is assigned to this PNC, this PNC is qualified to be released for the indicated channel (`ComMPncComSignal` received by `Com_ReceiveSignal()` with signal attributes `ComMPncComSignalDirection == "RX" and ComMPncComSignalKind == "ERA" changed to "0"`), the indicated channel has `ComMPncGatewayType` set to `COMM_GATEWAY_TYPE_PASSIVE` and `ComMSynchronizedPncShutdownEnabled` is set to `TRUE`, then the `ComM` module shall perform the following actions:

- If all `ComM` users assigned to this PNC request "No Com", all `ComMPncComSignals` received by `Com_ReceiveSignal()` from all

channels which are assigned to this PNC where the signal attributes `ComMPncComSignalDirection == "RX"` and `ComMPncComSignalKind == "ERA"` and the channel attribute `ComMPncGatewayType == "COMM_GATEWAY_TYPE_ACTIVE"` are set to "0", ComM shall call `Nm_RequestSynchronizedPncShutdown (<channel>, <PncId>)` for each of those <channel> with <PncId> of the current handled PNC

- The sub state `COMM_PNC_REQUESTED` shall be left and the sub state `COMM_PNC_READY_SLEEP` shall be entered

](SRS_ModeMgm_09269)

[SWS_ComM_01082] {DRAFT} [When a request to forward a synchronized PNC shutdown has been indicated via a call of `ComM_Nm_ForwardSynchronizedPncShutdown (<channel>, <PNC bit vector>)` and all following conditions apply:

- all ComM users assigned to this PNC request "No Com",
- all corresponding PNC bits are set to '0' in ERAn of all channels which are referenced by this PNC via `ComMChannelPerPnc` (see **ECUC_ComM_00880**) where the channel attribute `ComMPncGatewayType` is set to `COMM_GATEWAY_TYPE_ACTIVE`,
- the indicated channel of the `ComM_Nm_ForwardSynchronizedPncShutdown` call is assigned to this PNC and the PNC is indicated for a shutdown (PNC bit set to '1' in the given PNC bit vector),
- the indicated channel has `ComMPncGatewayType` set to `COMM_GATEWAY_TYPE_PASSIVE` and the channel is referenced via `ComMChannelPerPnc` (see **ECUC_ComM_00880** :),
- `ComMSynchronizedPncShutdownEnabled` is set to `TRUE`,

then the ComM module shall perform the following actions:

- ComM shall call `Nm_RequestSynchronizedPncShutdown (<channel>, <PncId>)` for each <channel> with <PncId> of the current handled PNC, where `ComMPncGatewayType` is set to `"COMM_GATEWAY_TYPE_ACTIVE"` and the channel is referenced via `ComMChannelPerPnc` (see **ECUC_ComM_00880** :)
- The sub state `COMM_PNC_REQUESTED` shall be left and the sub state `COMM_PNC_READY_SLEEP` shall be entered

](SRS_ModeMgm_09269)

Comment on [SWS_ComM_01082]:

- Every time an intermediate PNC coordinator (PNC coordinator which have at least one `ComMChannel` with `ComMPncGatewayType` set to `COMM_GATEWAY_TYPE_PASSIVE`) receive an top-level PNC coordinator Nm frame, the ComM shall immediately release the PNC, forward the PN information of the top-level PNC coordination Nm frame and request an top-level PNC coordination Nm frame on `ComMChannels` with `ComMPncGatewayType` set to `COMM_GATEWAY_TYPE_ACTIVE` which are assigned to the affected PNC

- ComM has to ensure that the procedure upon the reception of a top-level PNC coordination Nm frame has to be performed as fast as possible, to minimize the delay of the synchronized PNC shutdown
- The forwarding of a synchronized PNC shutdown is not performed if a local user has indicated to request the affected PNC, or a PNC request was received via a ComM channel with `ComMPncGatewayType` set to `COMM_GATEWAY_TYPE_ACTIVE`. The request for a PNC either local requested or remotely requested always overrule a request for a synchronized PNC shutdown.

[SWS_ComM_01051] {OBSOLETE replaced by SWS_ComM_01082} [If a request to forward a synchronized PNC shutdown has been indicated via a call of `ComM_Nm_ForwardSynchronizedPncShutdown(<channel>)` for this PNC, the PNC is qualified to be released and the precondition to forward the synchronized PNC request are not fullfield (at least one ComM user request Full Com for this PNC or at least one ComM channel assigned to this PNC has `ComMPncComSignal` set to "1" where the signal attributes `ComMPncComSignalDirection == "RX"` and `ComMPncComSignalKind == "ERA"` and the channel attribute `ComMPncGatewayType == "COMM_GATEWAY_TYPE_ACTIVE"`, see **[SWS_ComM_01050]**), then the ComM modul shall reject to perform the forwarding of a synchronized PNC shutdown and stay in sub state `COMM_PNC_REQUESTED`.](SRS_ModeMgm_09269)

[SWS_ComM_01052] {OBSOLETE replaced by [SWS_ComM_01083]} [If `ComMSynchronizedPncShutdownEnabled` is set to `TRUE`, all ComM channels which are assigned to this PNC have `ComMPncGatewayType` set to `COMM_GATEWAY_TYPE_ACTIVE`, the according PNC bit in all ERAn is equal to 0 and all ComMUsers assigned to this PNC request "No Communication", the API `Nm_RequestSynchronizedPncShutdown (<channel>, <PncId>)` shall be called , whereat `<channel>` represent the current handled `ComMChannel` and `<PncId>` the `ComMPncId` of this PNC.](SRS_ModeMgm_09269)

[SWS_ComM_01083] {DRAFT} [If `ComMSynchronizedPncShutdownEnabled` is set to `TRUE` and `ComMPncGatewayType` set to `COMM_GATEWAY_TYPE_ACTIVE` on all ComM channels assigned to this PNC, the API `Nm_RequestSynchronizedPncShutdown (<channel>, <PncId>)` shall be called, whereat `<channel>` represent the current handled `ComMChannel` and `<PncId>` the `ComMPncId` of this PNC under the following conditions:

- corresponding PNC bit in ERAn is equal to "0"
- all ComMUsers assigned to this PNC request "No Communication"
- The channel is referenced via `ComMChannelPerPnc` (see **ECUC_ComM_00880** :) by this PNC

](SRS_ModeMgm_09269, SRS_ModeMgm_09280)

Comment on [SWS_ComM_01083]: Everytime a PNC is released, synchronized PNC shutdown is configured and the ECU act as an top-level PNC coordinator (all coordinated PNC are mapped to `ComMChannels` where `ComMPncGatewayType` is set to `COMM_GATEWAY_TYPE_ACTIVE`) a PN shutdown message has to be triggered. Therefore ComM forward the PN information regarding the detection of a

released PNC to NmIf by calling `Nm_RequestSynchronizedPncShutdown` for each `ComMChannel` the PNC is assigned to. NmIf is forwarding the call to the affected `<Bus>Nm`. The PN shutdown message is transmitted within the `<Bus>Nm_Mainfunction`.

[SWS_ComM_00966] {OBSOLETE replaced by [SWS_ComM_01084]} [If `ComM0PncVectorAvoidance` is set to TRUE and if all signals are set to '0' (because of [SWS_ComM_00959](#)) for the referenced `ComMChannel(s)`, the ComM module shall release this `ComMChannel`.

As soon as at least one signal is set back to '1' again, the ComM module shall request this `ComMChannel` again.]()

[SWS_ComM_01084] {DRAFT} [In sub state `COMM_PNC_REQUESTED` if `ComM0PncVectorAvoidance` is set to TRUE and all PNC bits in the calculated IRA of a `ComMChannel` referenced via `ComMChannelPerPnc` (see **ECUC_ComM_00880** :) are set to '0', the ComM module shall release this `ComMChannel`. As soon as at least one bit in the IRA changes back to '1' again, the ComM module shall request this `ComMChannel` again.](`SRS_ModeMgm_09270`, `SRS_ModeMgm_09256`)

Comment on [SWS_ComM_01084]: As long as a PNC is requested remotely (i.e. at least one PNC bit within ERAn assigned to this PNC equals '1') and the configuration switch `ComMPncGatewayEnabled` is set to TRUE, `COMM_PNC_REQUESTED` will be the current PNC state.

7.1.3.6 On entry PNC sub state `COMM_PNC_READY_SLEEP`

[SWS_ComM_00960] {OBSOLETE replaced by [SWS_ComM_01085]} [When entering the PNC sub state `COMM_PNC_READY_SLEEP` from `COMM_PNC_REQUESTED`, the API `Com_SendSignal()` shall be called with the value '0' for the bit representing this PNC for all Com signals assigned to this PNC with `ComMPncComSignalDirection` "TX".]()

[SWS_ComM_01085] {DRAFT} [When entering the PNC sub state `COMM_PNC_READY_SLEEP` from `COMM_PNC_REQUESTED`, then the PNC bit representing this PNC within the IRA shall be set to value '0' and the aggregated internal PNC requests shall be forwarded to each channel which is referenced by this PNC by calling `Nm_UpdateIRA(<channel>, <IRA>)`](`SRS_ModeMgm_09248`, `SRS_ModeMgm_09250`)

[SWS_ComM_00961] {OBSOLETE replaced by [SWS_ComM_01086]} [When entering the PNC sub state `COMM_PNC_READY_SLEEP` from `COMM_PNC_REQUESTED`, ComM shall release the `COMM_FULL_COMMUNICATION` request for all configured ComM channels for this PNC.]()

[SWS_ComM_01086] {DRAFT} [When entering the PNC sub state `COMM_PNC_READY_SLEEP` from `COMM_PNC_REQUESTED`, ComM shall release the `COMM_FULL_COMMUNICATION` request for all configured ComM

channels referenced via `ComMChannelPerPnc` (see **ECUC_ComM_00880** :) by this PNC](`SRS_ModeMgm_00049`)

7.1.3.7 Behavior in PNC sub state **COMM_PNC_READY_SLEEP**

As long as the PNC is requested (i.e. the PNC bit representing this PNC within EIRA equals '1') and no `ComMUser` assigned to this PNC requests "Full Communication", **COMM_PNC_READY_SLEEP** will be the current state.

[SWS_ComM_00940] [If the PNC is released (i.e. the PNC bit representing this PNC within EIRA equals '0'), the sub state **COMM_PNC_READY_SLEEP** shall be left and the sub state **COMM_PNC_PREPARE_SLEEP** shall be entered.](`SRS_ModeMgm_09248`)

[SWS_ComM_01087] {DRAFT} [The sub state **COMM_PNC_READY_SLEEP** shall be left and the sub state **COMM_PNC_REQUESTED** shall be entered if at least one `ComMUser` assigned to this PNC requests "Full Communication".](`SRS_ModeMgm_09246` ,`SRS_ModeMgm_09247` ,`SRS_ModeMgm_09248`)

7.1.3.7.1 PNC gateway related requirement

[SWS_ComM_00165] {OBSOLETE replaced by [SWS_ComM_01088]} [In PNC sub state **COMM_PNC_READY_SLEEP** when at least one `ComMUser` assigned to this PNC requests "Full Communication" or `ComMPncGatewayEnabled` ([ECUC ComM_00887](#)) equals `TRUE`, one bit representing this PNC in an `ERAn` changes to '1' and `ComMPncGatewayType` is set for the according `ComMChannels` of these `ERAs`, this state shall be left and the sub state **COMM_PNC_REQUESTED** shall be entered.](`SRS_ModeMgm_09246` ,`SRS_ModeMgm_09247` ,`SRS_ModeMgm_09248`)

[SWS_ComM_01088] {DRAFT} [When in sub state **COMM_PNC_READY_SLEEP** at least one PNC bit representing this PNC in `ERAn` changes to '1', the sub state **COMM_PNC_READY_SLEEP** shall be left and the sub state **COMM_PNC_REQUESTED** shall be entered under the following conditions:

- the parameter `ComMPncGatewayEnabled` (see **ECUC_ComM_00887** :) is set to `TRUE`,
- this PNC references at least one channel via `ComMChannelPerPnc` (see **ECUC_ComM_00880** :) and the referenced channels have the `ComMPncGatewayType` set

](`SRS_ModeMgm_09248`)

7.1.3.8 On entry of PNC sub state **COMM_PNC_PREPARE_SLEEP**

[SWS_ComM_00952] [If the sub state **COMM_PNC_PREPARE_SLEEP** is entered, the timer `ComMPncPrepareSleepTimer` (see [ECUC ComM_00841](#)) shall be started with the configured initial value.](`SRS_ModeMgm_09279`)

7.1.3.9 Behavior in PNC sub state COMM_PNC_PREPARE_SLEEP

As long as the timer `ComMPncPrepareSleepTimer` (see [ECUC ComM_00841](#)) is running and no changes in `ComMUser`, `EIRA` or `ERAn` occur, `COMM_PNC_PREPARE_SLEEP` will be the current state.

[SWS_ComM_00947] [When the timer `ComMPncPrepareSleepTimer` (see [ECUC ComM_00841](#)) expires, the PNC sub state `COMM_PNC_PREPARE_SLEEP` shall be left and the PNC main state `COMM_PNC_NO_COMMUNICATION` shall be entered.](SRS_ModeMgm_09279)

[SWS_ComM_00948] [When in `COMM_PNC_PREPARE_SLEEP` at least one `ComMUser` assigned to this PNC requests “Full Communication”, the `COMM_PNC_PREPARE_SLEEP` state shall be left. The timer `ComMPncPrepareSleepTimer` shall be stopped and the sub state `COMM_PNC_REQUESTED` state shall be entered.](SRS_ModeMgm_09246)

[SWS_ComM_00950] [When in `COMM_PNC_PREPARE_SLEEP` the PNC bit representing this PNC within `EIRA` changes to ‘1’ and this PNC references at least one channel via `ComMChannelPerPnc` (see [ECUC ComM_00880](#) :), the sub state `COMM_PNC_PREPARE_SLEEP` shall be left. The timer `ComMPncPrepareSleepTimer` shall be stopped and the sub state `COMM_PNC_READY_SLEEP` shall be entered.]()

7.1.3.9.1 PNC gateway related requirements

[SWS_ComM_00951] {OBSOLETE replaced by [SWS_ComM_01089]} [When in sub state `COMM_PNC_PREPARE_SLEEP` at least one PNC bit within `ERAn` changes to ‘1’, the parameter `ComMPncGatewayEnabled` equals `TRUE` and `ComMPncGatewayType` is set for the according `ComMChannels` of these `ERAs`, the sub state `COMM_PNC_PREPARE_SLEEP` shall be left. The timer `ComMPncPrepareSleepTimer` shall be stopped and the sub state `COMM_PNC_REQUESTED` shall be entered.]()

[SWS_ComM_01089] {DRAFT} [When in sub state `COMM_PNC_PREPARE_SLEEP` at least one PNC bit representing this PNC in `ERAn` changes to ‘1’, then sub state `COMM_PNC_PREPARE_SLEEP` shall be left, `COMM_PNC_REQUESTED` shall be entered and timer `ComMPncPrepareSleepTimer` shall be stopped under the following conditions:

- the parameter `ComMPncGatewayEnabled` (see [ECUC ComM_00887](#) :) is set to `TRUE`,
- this PNC references at least one channel via `ComMChannelPerPnc` (see [ECUC ComM_00880](#) :) and the referenced channels have the `ComMPncGatewayType` set

](SRS_ModeMgm_09248)

7.1.4 PNC Gateway

The PNC Gateway feature is used to span (logical) partial network clusters across bus / communication channel boundaries, “gatewaying” PNC requests from one bus/network to the others. (Therefore, for a PNC gateway to exist, it needs to be connected to multiple physical channels.)

To do so, the PNC gateway configuration contains information for each PNC which physical channels are required to reach all members of that PNC (PNC-to-channel-mapping, see Figure 3).

The PNC gateway collects PNC requests from all of its multiple active channels (which are called active since it actively keeps them awake, if required) and aggregates them. The PNC gateway sends the aggregated PNC state in the network to all its active channels, which causes all nodes to have the same view on the global PNC request state as the gateway.

If the PNC gateway is not the topmost PNC gateway in the network hierarchy, the PNC gateway will also send the aggregated PNC request state of all subordinate nodes, plus its own internal request state, to its superior PNC coordinator, which is connected via the so-called “passive” channel (which is called passive because it’s the opposite of active).

The superior PNC coordinators will aggregate the subordinate coordinators’ PNC request states, so the top level coordinator will know about all active PNC requests in the network, and send that info to the subordinate nodes.

Subordinate PNC coordinators forward the PNC request information received on their passive channel to their active channels to distribute the top level coordinators holistic view of the PNC request state to all leaf nodes in the logical hierarchy, so every node in the system is on the same page regarding the PNC request state.

A PNC coordinator must never aggregate and send back the information received via its passive channel in order not to create an endless mirroring loop of “phantom PNC requests”.

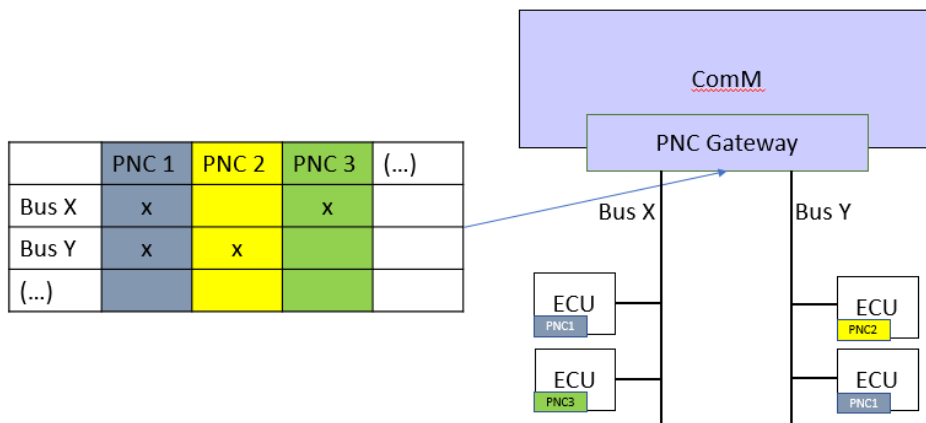


Figure 3: PNC-to-channel-mapping

The PNC to channel mapping is provided statically by configuration. Additionally, the optional feature Dynamic PNC-to-channel-mapping (see chapter 7.1.5) could be used to extend the PNC-to-channel mapping during run-time.

Note that when PNC Gateway is active and even if a PNC is only assigned to one channel, coordination might occur when request comes in from another channel where PNC is not assigned to. This is intended as there might be only PNC-requestor on the other channel which is not interested in being kept awake by this PNC.

7.1.4.1 Support for not coordinated PNCs assigned to multiple channels

Comment: When a Partial Network is assigned to more than one ComMChannel than this PNC is coordinated either on all affected ComMChannels or not at all (see AUTOSAR_TPS_SystemTemplate [constr_5094]).

Note: If PNCs are assigned to different ComMChannels and those ComMChannels are not coordinated by a PNC gateway, then the network topology and communication design has to ensure, that the affected ComMChannels are requested and released to the same point in time. If PNCs are used, an application should not care about ComMChannel states, and additionally, ComM will not take care about ComMChannel states for this use case, since the PNC coordination for those ComMChannels is not performed. Or in other words, if a PNC is requested (passively) then also all referenced ComMChannels shall be requested (passively), because an application expects that all ComMChannels assigned to this PNCs reside in COMM_FULL_COMMUNICATION.

Figure 4 depict an example for a PNC gateway (Node2) with not coordinated ComMChannels

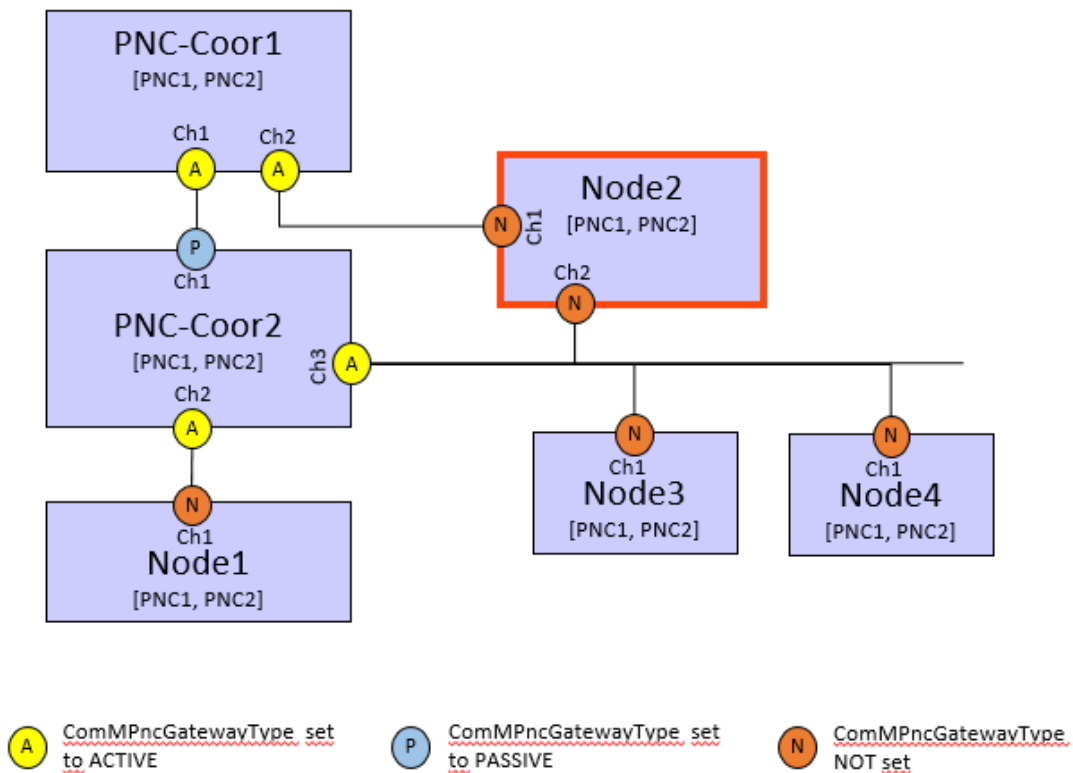


Figure 4: Example for a PNC gateway with not coordinated ComMChannels (see Node2)

7.1.4.2 Active PNC Gateway

Note: Even if the configuration parameter `ComMPncGatewayEnabled` (see **ECUC_ComM_00887** :) is TRUE and the parameter `ComMPncGatewayType` is set to `COMM_GATEWAY_TYPE_ACTIVE` for a `ComMChannel` (see **ECUC_ComM_00842** :), the active PNC gateway still behaves as shown in Figure 2: PNC State Machine.

Comment: An active PNC gateway on a system channel shall be the last node on a system channel that releases a PNC.

Comment: If the PNC bit for a PNC is equal to zero in all ERA_n, no other node than the PNC gateway is requesting the PNC.

7.1.4.3 Passive PNC Gateway

Comment: The passively coordinated channels exist only if they are connected to more than one PNC gateway. If the PNC gateway functionality of ComM is enabled (`ComMPncGatewayEnabled` is set to TRUE) ComM channels mapped to this PNC gateway can be set to type active or passive (`COMM_GATEWAY_TYPE_ACTIVE` or `COMM_GATEWAY_TYPE_PASSIVE`). If a ComM channel is mapped to two different PNC gateways, only one gateway coordinates this channel actively, while the other passively. That means, a PNC gateway is always mapped to at least one ComM channel type active and may be mapped to one or some ComM channels type passive.

Comment: A PNC gateway requests the PNC if a local ComM user requests the PNC or at least one PNC bit within ERA originate from the actively coordinated system channels of a passive PNC gateway is not equal to 0.

Comment to [SWS_ComM_00959]: A PNC gateway calculates the PNCs bit value in the ERA Tx PNC bit vectors to be sent for a passively coordinated channel, in the same manner as the PNC bit value in ERA for an actively coordinated channel, but sets the PNC's bit to '0' according to the rules of **[SWS_ComM_00959]**.

7.1.4.4 Synchronized PNC shutdown

A PN topology always reflects a hierarchical topology, where the so-called top-level PNC coordinator is located on the highest level. On the subordinated levels multiple so-called intermediate PNC coordinators and PNC leaf nodes could reside.

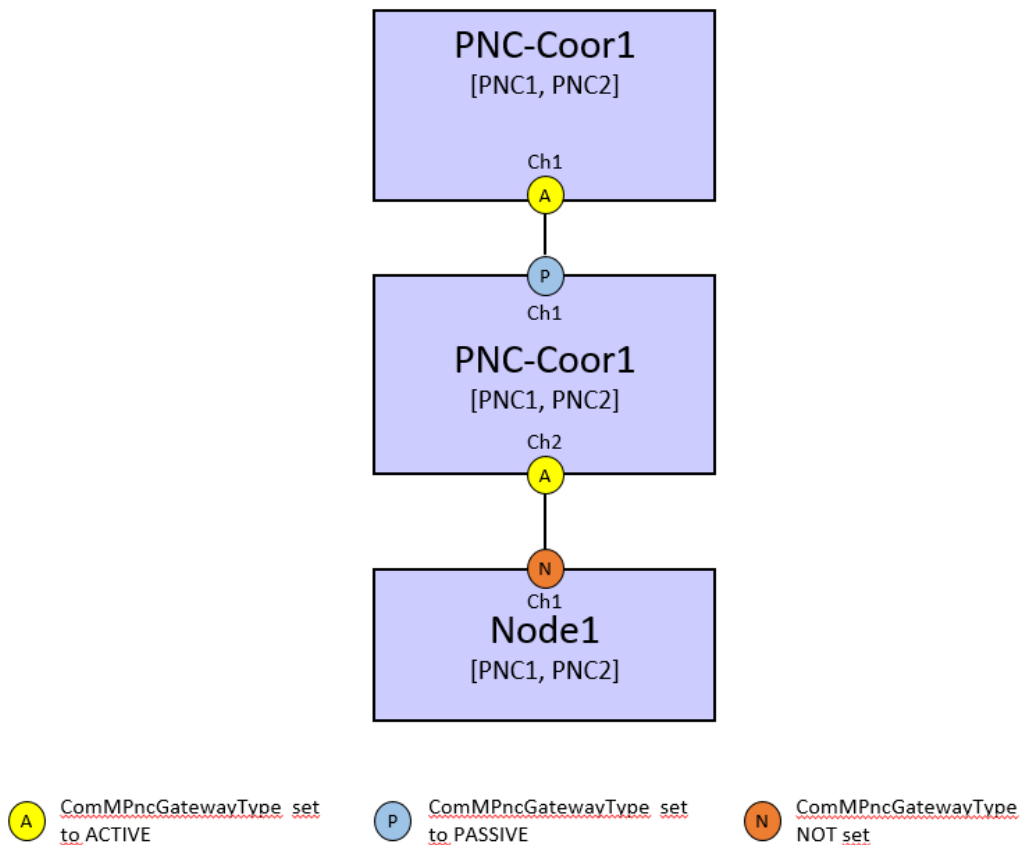


Figure 5: Example for a partial network (PN) topology that reflect the hierarchy

Figure 5 shows PNC-Coor1 as top-level PNC coordinator, PNC-Coor2 as intermediate PNC coordinator and Node1 as PNC leaf node which resides on the lowest level of the PN topology. For example, if Node1 requests PNC1, then the PNC request is propagated across the PN to the top-level PNC coordinator. The top-level PNC coordinator “takes over” the PNC request and mirrors back the PNC request to the whole PN. If for example Node1 releases PNC1 and no other ECU in the network has PNC1 requested, then Node1 will still receive Nm frames from the top-level PNC coordinator where the PNC1 is requested. The release of the PNC leaf node is not forwarded immediately across the PN topology from the PNC leaf node to the top-level PNC coordinator. The release of a PNC is delayed by the PN reset time on each PN topology level. If the top-level PNC coordinator detect that a PN reset timer for a particular PNC expires, then no other ECU in the PN request this PNC. The top-level PNC coordinator resets the PN reset timer of the released PNC once more and transmits a so-called PN shutdown message to ensure a nearly synchronized shutdown of the PNC, across all PN levels from the top-level PNC coordinator down to the PNC leaf node. An intermediate PNC coordinator reacts immediately upon reception on a PN shutdown message. Therefor the intermediate PNC coordinator releases the indicated PNC, resets the PN reset timer once more and forwards the PN shutdown message on all ComMChannels which are actively coordinated and assigned to the affected PNC. Thus, all PNC state machines of the released PNC across all PN level from the top-level PNC coordinator down to the PNC leaf nodes reside in COMM_PNC_READY_SLEEP and reset the corresponding PN reset timer nearly in the same point in time. This will lead to a synchronized PNC shutdown to avoid timeouts on application level.

Please refer also to the sequence diagrams Figure 17 and Figure 18 which depict the handling of a synchronized PNC shutdown in the role of a top-level PNC coordinator and an intermediate PNC coordinator.

Note:

- For ComMChannels which are configured for a uni-directional PNC handling (see 7.1.6.2), no synchronized PNC shutdown is performed.
- For PNCs which reference a ComMChannel via the parameter ComMChannelPerTxOnlyPnc (see 7.1.6.3), no synchronized PNC shutdown is performed.

7.1.4.5 Support for multiple top-level PNC coordinators

According to chapter 7.1.4.4 a PN topology always have at least one top-level PNC coordinator. The top-level PNC coordinator for a particular PNC is designated if all ComMChannels have ComMPncGatewayType set to GATEWAYE_TYPE_ACTIVE where this particular PNC is assigned to (see [SWS_Comm_01083]). Thus, for different PNCs it is possible to have different top-level PNC coordinators. But for the same PNC only one top-level coordinator is supported. The modelling of such a PN topology has to ensure a strict separation of PNCs. Figure 6 shows a supported PN topology for multiple top-level PNC coordinators.

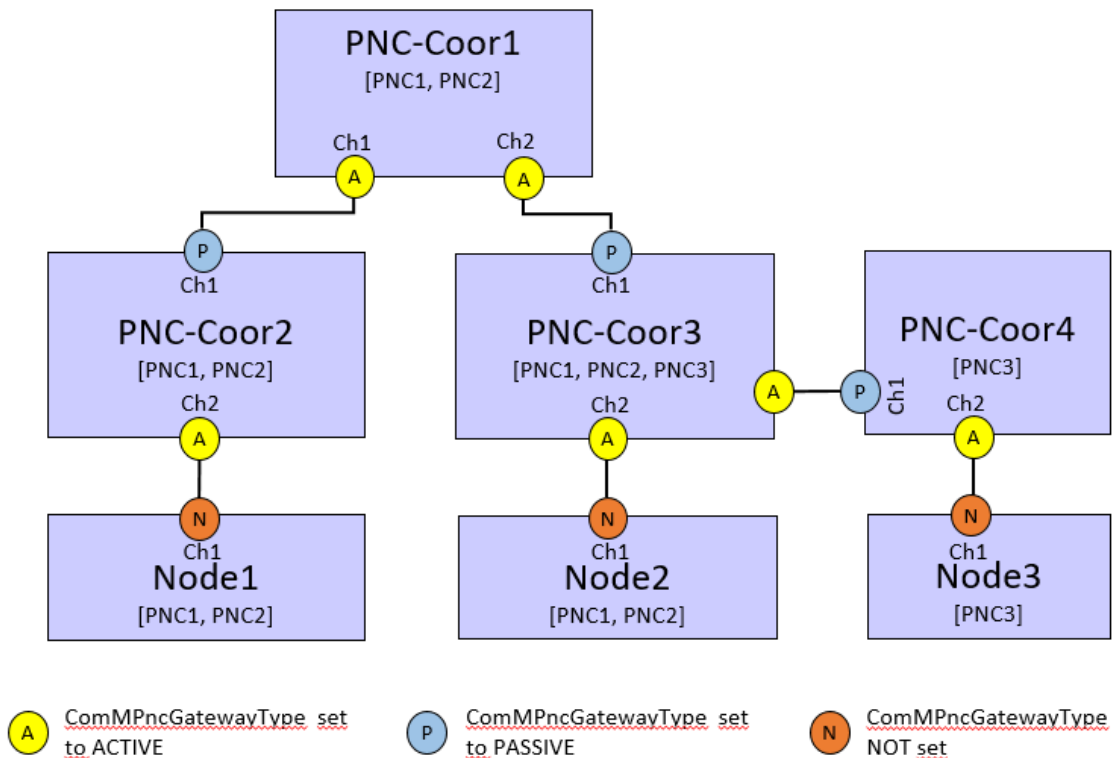


Figure 6: Example for a valid PN with multiple top-level PNC coordinators

In Figure 6 PNC-Coor1 act as top-level PNC coordinator for PNC1 and PNC2. PNC-Coor3 act as top-level PNC coordinator for PNC3. Thus, if synchronized PNC shutdown is enabled, PNC-Coor1 is responsible to initiate a synchronized PNC

shutdown for PNC1 and PNC2. PNC-Coor3 is responsible to initiate a synchronized PNC shutdown for PNC3.

Note: The network topology and communication design has to ensure a valid and supported PN topology

7.1.5 Dynamic PNC-to-channel-mapping (optional)

This feature adds the possibility to update the PNC-to-channel-mapping of the PNC Gateway during runtime. This update works via a request-response-based learning process of all participating Nodes. When Partial Network learning is requested within the Nm PDUs, all participating Nodes will respond their current PNC membership on the corresponding channel and the PNC Gateway then updates the current PNC-to-channel-mapping accordingly.

[SWS_ComM_CONSTR_00004] {DRAFT} [If at least one channel is referenced by a PNC by using ComMChannelPerTxOnlyPnc, then ComMDynamicPncToChannelMappingSupport shall be set to FALSE. Otherwise the configuration is invalid. A configuration tool shall reject such a configuration as invalid (error).] ()

[SWS_ComM_01026] {DRAFT} [If the function ComM_Nm_PncLearningBitIndication has been called on a channel where ComMDynamicPncToChannelMappingEnabled is set to TRUE or when ComM calls Nm_PnLearningRequest on a channel ComM shall set the PNC Learning Phase to active for the according channel.](SRS_ModeMgm_09260)

[SWS_ComM_01029] {DRAFT} [If ComMDynamicPncToChannelMappingEnabled is set to TRUE and function ComM_Nm_RepeatMessageLeftIndication has been called ComM shall set the PNC Learning Phase to inactive for the according channel.](SRS_ModeMgm_09265)

[SWS_ComM_01028] {DRAFT} [If ComMPncGatewayEnabled is set to TRUE and the function ComM_Nm_PncLearningBitIndication has been called for a channel either of the following actions shall be performed:

- when ComM_Nm_PncLearningBitIndication is called for a channel where ComMPncGatewayType is set to COMM_GATEWAY_TYPE_ACTIVE, ComM shall forward the Learning Request by calling Nm_PnLearningRequest on all further coordinated ComM channels (active or passive) with ComMDynamicPncToChannelMappingEnabled is set to TRUE
- when ComM_Nm_PncLearningBitIndication is called for a channel where ComMPncGatewayType is set to COMM_GATEWAY_TYPE_PASSIVE, ComM shall forward the Learning Request by calling Nm_PnLearningRequest on ComM channels with ComMPncGatewayType set to COMM_GATEWAY_TYPE_ACTIVE and ComMDynamicPncToChannelMappingEnabled is set to TRUE

](SRS_ModeMgm_09261)

Rational: Partial network learning bit needs to be forwarded to all nodes in the network but it needs not to be mirrored back even in the case when network topology contains circles.

Note: Forwarding only on all active coordinated channels where the request has not been received is sufficient as due to there is only one master gateway (see AUTOSAR_TPS_SystemTemplate [constr_5094] the information is spread over the whole system. The initial request is started by the application of one node in the network with a call of ComM_PnLearningRequest (refer to [SWS_ComM_01046]).

[SWS_ComM_01030] {OBSOLETE replaced by [SWS_ComM_01090]} [If ComMPncGatewayEnabled and ComMPncDynamicMappingSupport are set to TRUE and when the PNC Learning Phase is active, then ComM shall forward received ERA Rx information on channels where ComMPncDynamicMappingEnabled is set to TRUE. Therefore, ComM shall set the affected PN bit(s) to the assigned Com signals with ComMPncComSignalDirection "TX" on all other channels where ComMPncDynamicMappingEnabled is set to TRUE.] (SRS_ModeMgm_09261)

[SWS_ComM_01090] {DRAFT} [If ComMPncGatewayEnabled and ComMPncDynamicMappingSupport are set to TRUE and when the PNC Learning Phase is active, then ComM shall forward received ERA Rx information on channels where ComMPncDynamicMappingEnabled is set to TRUE. Therefore, ComM shall set the affected PNC bit(s) in all affected ERAN on all other channels where ComMPncDynamicMappingEnabled is set to TRUE.] (SRS_ModeMgm_09261)

7.1.5.1 Update PNC-to-channel-mapping

The PNC Gateway needs to be capable to update its PNC-to-channel Mapping on runtime.

[SWS_ComM_01031] {OBSOLETE replaced by [SWS_ComM_01091]} [If ComMPncGatewayEnabled is set to TRUE and when the PNC Learning Phase is active and an ERA Rx Signal is received on a channel where ComMDynamicPncToChannelMappingEnabled is set to TRUE ComM shall set PNC-to-channel Mapping to 1 for every ComMPnc on the according channel where this ERA Rx Signal has been received with 1 for the according PNC.](SRS_ModeMgm_09258)

[SWS_ComM_01091] {DRAFT} [If ComMPncGatewayEnabled is set to TRUE and when the PNC Learning Phase is active and an PNC bit in the ERA is set to "1" on a channel where ComMDynamicPncToChannelMappingEnabled is set to TRUE ComM shall set PNC-to-channel Mapping to 1 for every ComMPnc on the according channel where this PNC bit in the ERA has been set to "1" for the according PNC.](SRS_ModeMgm_09258)

7.1.5.2 PNC Membership Forwarding

Every participating Node has to transmit its current PNC membership during PNC Learning phase. The PNC Gateway needs additionally also forward PNC memberships received from other channels.

[SWS_ComM_01032] {OBSOLETE replaced by [SWS_ComM_01092]} [If `ComMPncGatewayEnabled` is set to FALSE and when the PNC Learning Phase is active, the ComM shall send for all ComM channels where `ComMDynamicPncToChannelMappingEnabled` is set to TRUE the assigned Com signals with `ComMPncComSignalDirection` "TX" with the value of the current PNC membership. `.(SRS_ModeMgm_09262)`

[SWS_ComM_01092] {DRAFT} [If `ComMPncGatewayEnabled` is set to FALSE and when the PNC Learning Phase is active, the ComM shall set the corresponding PNC bits in the IRA with the value of the current PNC membership and call `Nm_UpdateIRA(<channel>, <IRA>)` for all ComM channels where `ComMDynamicPncToChannelMappingEnabled` is set to TRUE. `.(SRS_ModeMgm_09262, SRS_ModeMgm_09250)`

[SWS_ComM_01033] {OBSOLETE replaced by [SWS_ComM_01093]} [If `ComMPncGatewayEnabled` is set to TRUE and when the PNC Learning Phase is active, the ComM shall send for all ComM channels where `ComMDynamicPncToChannelMappingEnabled` is set to TRUE the assigned Com signals with `ComMPncComSignalDirection` "TX" with the value of the current PNC membership merged with the PNC information that needs to be forwarded according to **[SWS_ComM_01030]**. `.(SRS_ModeMgm_09261)`

[SWS_ComM_01093] {DRAFT} [If `ComMPncGatewayEnabled` is set to TRUE and when the PNC Learning Phase is active, the ComM shall call `Nm_UpdateIRA(<channel>, <IRA>)` for all ComM channels where `ComMDynamicPncToChannelMappingEnabled` is set to TRUE with the IRA set with the value of the current PNC membership merged with the PNC information that needs to be forwarded according to **[SWS_ComM_01090]**. `.(SRS_ModeMgm_09261, SRS_ModeMgm_09250)`

7.1.6 Partial Networking Configuration Hints

The partial network configuration has to consider the configuration of the corresponding PN filter mask in NM of the corresponding NM-channels. If using a `SystemDescriptionExtract` to configure the BSW stack and the modelled partial network is available within the `SystemDescriptionExtract`, then the PN filter mask is derived automatically per each NM-channel. It is up to the integration process and the integration restriction to change the PN filter mask manually after the derivation. The integration process and particular restrictions is not dedined by AUTOSAR to support flexibility.

The following chapters describe the supported use cases to be considered for a proper PNC handling of PNC gateways and none PNC gateways

7.1.6.1 Bi-directional PNC handling

This means, that PNC requests are always transferred in both directions. The handling of PNC request is symmetrically for transmission and reception:

- PNC gateways forward incoming (external) PNC request and mirror them back
- None PNC gateways react on incoming PNC request and transmit PNC requests according to PNC assignment

Thus, ComM transmit and handle received PNC requests for a PNC on those ComMChannels, where a particular PNC refer to the ComMChannel by using the parameter ComMChannelPerPnc (see **ECUC_ComM_00880** :). The correctness of received PNCs within the PNC bit vector according to the ComMChannel assignment has to be ensured by a proper configuration of the PN filter mask per NM-channel in the Nmlf.

Note: ComM doesn't check the correctness of the received PNC according to the ComMChannel assignment:

- For EIRA updates, ComM has no possibility to check on which ComMChannel the PNC request was received, since the ComMChannel information is not forwarded by the Nmlf.
- For ERA updates, a check could be done, but it was decided in AUTOSAR to release ComM from this responsibility.

In both cases (PNC gateway use case and none PNC gateway) the PN filter mask of a NM-channel have to pass all PNCs which are reference the corresponding ComMChannels via ComMChannelPerPnc

7.1.6.2 Uni-directional PNC handling

This means, that PNC requests are transferred in one direction. The handling of PNC request is asymmetrically for transmission and reception:

- PNC gateways forward incoming (external) PNC request but do not mirror it back on the ComMChannel the PNC request was received
- None PNC gateways transmit PNC requests for PNCs on ComMChannels, where this PNC is not assigned to

For PNC gateways the PN filter mask of a NM-channel has to pass all PNCs which are acceptable to be received on a ComMChannel and the PNCs do NOT refer the ComMChannels via ComMChannelPerPnc (no PNC-channel relation exist). Additionally, the PNC ERA handling has to be enabled for the according NM-channel. For received PNC requests on ComMChannel where **no** PNC-channel relation exist, only the forwarding of PNC requests and no mirroring back on the receiving ComMChannel will be performed. For received PNC requests on a ComMChannel where a PNC-channel relation exist, the bi-directional PNC handling will be performed. The uni-directional PNC handling for PNC gateways could be used, e.g. when a network needs information from a certain PNC but there is no need to provide any information back.

For none PNC gateways the PN filter mask of a NM-channel has to reject all PNCs which are considered to be only transmitted on a ComMChannel. Received PNC request of those ComMChannel should not be handled and therefore should not reach ComM.

The uni-directional PNC handling for none PNC gateways could be used, e.g. when an ECU needs to wake-up or keep-alive some functionality without being part of it.

7.1.6.3 Transmission only PNC handling

This means, that internal PNC requests (i.e. triggered by a user) or PNC requests due to PNC coordination (i.e. triggered externally by a received PNC request (ERA Rx)) are transferred for transmission. Thereby only the internal request array (IRA) is updated without requesting the according ComMChannel. This could be achieved via a proper configuration:

- If a PNC refer to a ComMChannel by using the parameter ComMChannelPerTxOnlyPnc (see **ECUC_ComM_00900** :), then the corresponding PNC state machine is NOT requesting the referenced ComMChannel state machine, if the PNC is requested. Such PNC requests are transmitted on the according ComMChannel within a NM-PDU, if the ComMChannel is requested by e.g. another PNC which refer to this ComMChannel via ComMChannelPerPnc or a dedicated ComMUser

Note: If a PNC refer to a ComMChannel via ComMChannelPerTxOnlyPnc (see **ECUC_ComM_00900** :), the ComMChannel is not referenced by another PNC via ComMChannelPerPnc and a PNC request is received on that channel or a wake up is detected, then the PNC statemachine will stay in PNC_NO_COMMUNICATION. (Please refer to **[SWS_ComM_01063, [SWS_ComM_01064], [SWS_ComM_01065], [SWS_ComM_01066]**)

The transmission-only-PNC handling could be used e.g. for none PNC gateways to request only PNCs without additionally requesting the NM.

The transmission only PNC handling could be used e.g. for PNC gateways to receive uni-directional PNC request (PNC1) on one channel (channel A) and forward the PNC request without requesting the NM on another channel (channel B). On channel B PNC1 is configured for bi-directional PNC handling, therefore a received PNC request for PNC1 is forwarded to channel A by considering to request the affected ComMChannels and the according NM.

Note: The reference ComMChannelPerTxOnlyPnc cannot be derived from a SystemDescriptionExtract. The reference from a PNC to a ComM channel via ComMChannelPerTxOnlyPnc could only be added manually within the integration phase.

7.2 ComM channel state machine

[SWS_ComM_00979] [If the optional PNC functionality is enabled (see [ECUC ComM 00883](#)), all PNC actions shall be performed before the channel related actions are executed.](SRS_ModeMgm_09243)

[SWS_ComM_00980] [If the parameter ComMPncNmRequest is set to TRUE (see [ECUC ComM 00886](#)), if the "FULL Communication" is requested due to a change in the PNC state machine to COMM_PNC_REQUESTED (see [SWS_COMM_00993](#)) API Nm_NetworkRequest() shall be called, even if the current state is already "Full communication".](SRS_ModeMgm_09243)

Rationale: It is the trigger to enable the NM to transmit the NM message immediately n-times (n=configurable) to ensure a wake up and a synchronization of the PNC transceiver.

[SWS_ComM_00051] [ComM shall implement one channel state machine as shown in Figure 7 with requirements as listed in Table 1 for every communication channel independently.](SRS_ModeMgm_09080)

Rationale for SWS ComM 00051: Needed communication capability of channels may be different, thus the controlling must be independent.

Use Case for SWS ComM 00051: On an ECU with CAN and LIN channel, only the LIN requires full communication to request e.g. sensor values while the CAN remains inactive.

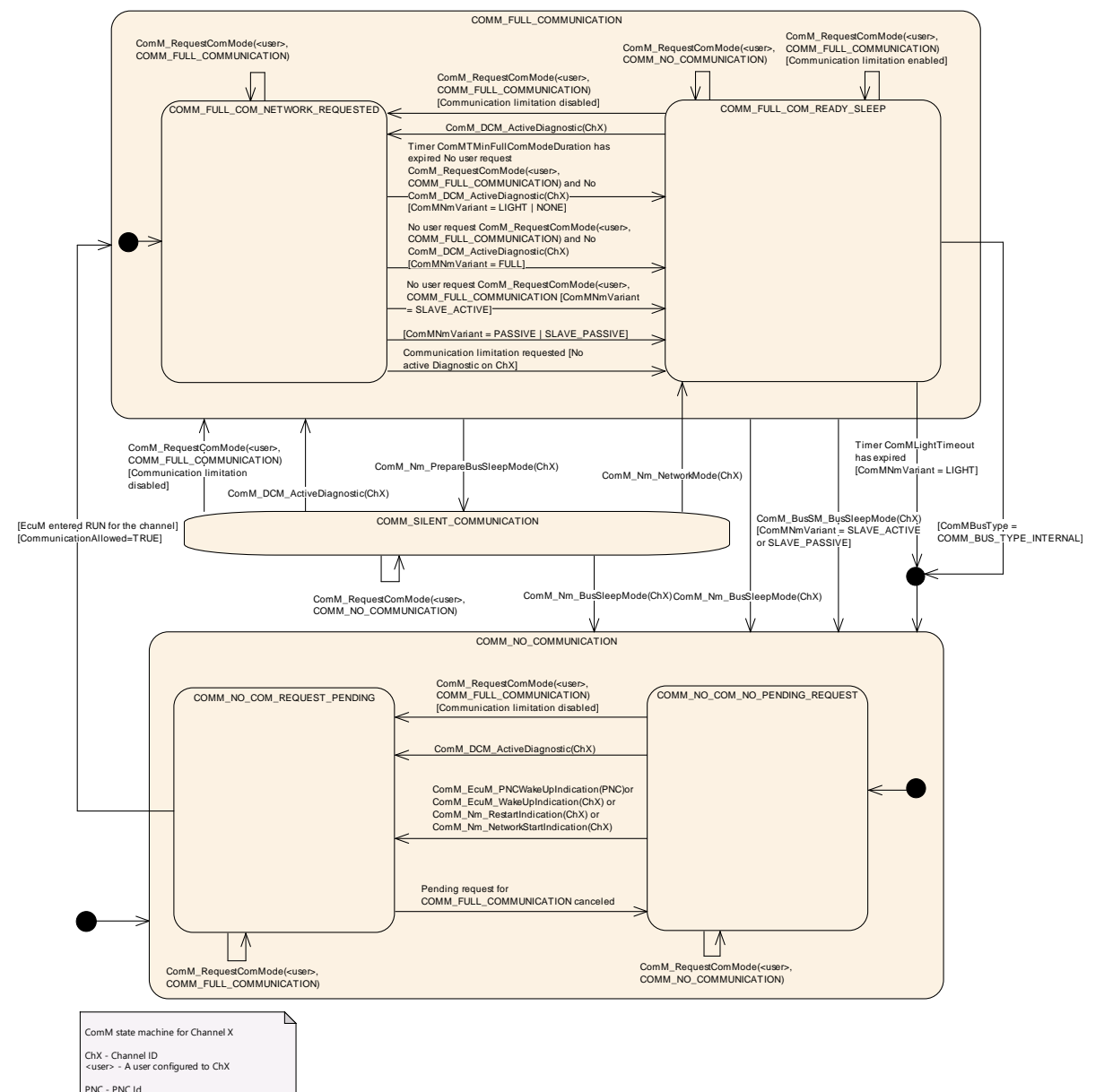


Figure 7: ComM channel state machine

State	Section / Requirement
COMM_NO_COMMUNICATION	7.2.2 Entering state: SWS ComM 00898 , SWS ComM 00313 , SWS ComM 00073 , SWS ComM 00288 In sub-state <code>COMM_NO_COM_NO_PENDING_REQUEST</code> : SWS ComM 00875 , SWS ComM 00876 , SWS ComM 00893 , SWS ComM 00894 , SWS ComM 00694 , SWS ComM 01014 , SWS ComM 01015 In sub-state <code>COMM_NO_COM_REQUEST_PENDING</code> : SWS ComM 00895 , SWS ComM 00897
COMM_SILENT_COMMUNICATION	7.2.3 Entering state: SWS ComM 00071 In state: SWS ComM 00877 , SWS ComM 00878 SWS ComM 00295 , SWS ComM 00296
COMM_FULL_COMMUNICATION	7.2.4 Entering state: SWS ComM 00069 In state: SWS ComM 00637 , SWS ComM 00826 7.2.4.1 sub-state <code>COMM_FULL_COM_NETWORK_REQUESTED</code> : In sub-state: SWS ComM 00869 , SWS ComM 00870 , SWS ComM 00665 , SWS ComM 00888 , SWS ComM 00889 , SWS ComM 00890 7.2.4.2 sub-state <code>COMM_FULL_COM_READY_SLEEP</code> Entering sub-state: SWS ComM 00133 In sub-state: SWS ComM 00610 , SWS ComM 00671 , SWS ComM 00882 , SWS ComM 00883
Transition	Requirement
COMM_NO_COMMUNICATION → COMM_FULL_COMMUNICATION	SWS ComM 00893 , SWS ComM 00894 , SWS ComM 00694 , SWS ComM 00875 SWS ComM 00876 , SWS ComM 01014 , SWS ComM 01015
COMM_FULL_COM_NETWORK_REQUESTED → COMM_FULL_COM_READY_SLEEP	SWS ComM 00665
COMM_FULL_COM_READY_SLEEP → COMM_FULL_COM_NETWORK_REQUESTED	SWS ComM 00882 , SWS ComM 00883
COMM_FULL_COMMUNICATION → COMM_SILENT_COMMUNICATION	SWS ComM 00826
COMM_FULL_COM_READY_SLEEP → COMM_NO_COMMUNICATION	SWS ComM 00610 , SWS ComM 00671
COMM_FULL_COMMUNICATION → COMM_NO_COMMUNICATION	SWS ComM 00637
COMM_SILENT_COMMUNICATION → COMM_FULL_COMMUNICATION	SWS ComM 00877 , SWS ComM 00878
COMM_SILENT_COMMUNICATION → COMM_FULL_COM_READY_SLEEP	SWS ComM 00296
COMM_SILENT_COMMUNICATION → COMM_NO_COMMUNICATION	SWS ComM 00295

Table 1: Link to detailed explanation of the channel state machine resp. transition

[SWS_ComM_00879] [The ComM channel state machine shall consist of the three main states corresponding to the Communication Modes: `COMM_NO_COMMUNICATION`, `COMM_SILENT_COMMUNICATION` and `COMM_FULL_COMMUNICATION`.](SRS_ModeMgm_09083)

[SWS_ComM_00880] [The `COMM_FULL_COMMUNICATION` state shall have two sub-states `COMM_FULL_COM_NETWORK_REQUESTED` and `COMM_FULL_COM_READY_SLEEP`.](SRS_ModeMgm_09083)

[SWS_ComM_00881] [The `COMM_NO_COMMUNICATION` state shall have two sub-states `COMM_NO_COM_REQUEST_PENDING` and `COMM_NO_COM_NO_PENDING_REQUEST`](SRS_ModeMgm_09083)

Rationale for [SWS ComM 00879](#) and [SWS ComM 00880](#):
`COMM_FULL_COM_READY_SLEEP` and `COMM_SILENT_COMMUNICATION` are necessary to synchronize a communication shutdown on the bus. If only one ECU switches the communication off, the others store errors because this ECU stops sending application signals.

Comment: The main states present an abstracted status of communication capabilities per channel, which are in focus of the users' interests. The sub-states represent intermediate states, which perform activities to support a synchronized transition with external partners and managing protocols (e.g. NM)

[SWS_ComM_00485] [The default state for each ComM channel state machine shall be `COMM_NO_COMMUNICATION`.](SRS_ModeMgm_09083)

[SWS_ComM_00896] [Each ComM channel state machine shall only evaluate its corresponding communication status flag `CommunicationAllowed` according to [SWS ComM 00884](#) in sub-state `COMM_NO_COM_REQUEST_PENDING`.]()

Rationale for [SWS ComM 00896](#):

A `ComM_CommunicationAllowed(<channel>, FALSE)` ([SWS ComM 00871](#)) indication has no visible effect if the channel is not in sub-state `COMM_NO_COM_REQUEST_PENDING`, i.e. ComM channel state machine will not immediately change to state `COMM_NO_COMMUNICATION` if in another state as e.g. `COMM_FULL_COMMUNICATION`

[SWS_ComM_00472] [Main state changes (see [SWS ComM 00879](#)) shall be indicated to the users with the corresponding notifications (see section 8.6.1.5 and 8.6.1.6). Exception: Default state after initialization, see [SWS ComM 00313](#).](SRS_ModeMgm_09085)

Comment: If more than one user is related to the corresponding channel state machine, the ComM module has to perform a Fan-out to all users.

[SWS_ComM_00191] [The internal functionality of the ComM channel state machine(s) shall be invisible for the users. The user neither needs nor shall get any

information about the internal mechanisms and rules (e.g. "highest wins" strategy) of the ComM channel state machine.](())

An overview of the requested communication capabilities in the Corresponding Mode is shown in Table 2.

Communication Mode	Message Transmission	Message Reception	NM (COMM_NM_VARIANT=FULL)	Wake-up/Restart capability
COMM_FULL_COMMUNICATION	On	On	Bus communication requested	N/A
COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST	On	On	Bus communication requested	Request the lower layer to trigger a wake-up on the network
COMM_SILENT_COMMUNICATION	Off	On	Bus communication released	<ul style="list-style-type: none"> • User/diagnostic request • Network indication
COMM_NO_COMMUNICATION	Off	Off	Bus communication released	<ul style="list-style-type: none"> • User/diagnostic request • Passive wake-up

Table 2: Granted communication capabilities in the corresponding modes

[SWS_ComM_01056] [Requests for communication mode COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST shall be handled as request for COMM_FULL_COMMUNICATION within the ComM channel state machine. Deviations of ComM channel machine state transitions and behavior within the states are specified explicitly.](SRS_ModeMgm_09268)

Note for section 7.1.1 - 7.1.3: Each ComM channel state machine is responsible to handle one channel/network with a connected Bus State Manager (“corresponding” = the channel/network the ComM channel state machine is responsible for).

Note for section 7.1.1 - 7.1.3: The ComM module contains one or several ComM channel state machine(s). ComM channel state machine communicates directly with its connected Bus State Manager, other interfaces are handled by the ComM module.

7.2.1 ComM managed and managing channels

A ComM channel could reference other ComM channels. The reference is configurable by setting ComMManageReference (see [ECUC ComM 00893](#)). The source ComM channel of a ComMManageReference is called "managing channel" and the target ComM channel is called "managed channel". A managing channel could reference 0..n managed channels. A managed channel could be referenced by exclusively 1 managing channel.

This is used to support use cases, were a managing channel handle the interaction with the NM module and the managed channel has no NM.

Note: The following limitation have to be considered for a managing channel:

- ComMNmVariant of a managing channel is set to FULL (see [ECUC ComM 00568](#))

Note: The following limitations have to be considered for a managed channel:

- ComMNmVariant of a managed channel is set to LIGHT, since the managing channel is responsible for the interaction with the NmChannel (see [ECUC ComM 00568](#))

- ComMPncGatewayType of a managed channel is neither set to COMM_GATEWAY_TYPE_ACTIVE nor COMM_GATEWAY_TYPE_PASSIVE (see [ECUC ComM 00842](#))

7.2.2 Behavior in state COMM_NO_COMMUNICATION

[SWS_ComM_00898] [On entering state COMM_NO_COMMUNICATION the ComM channel state machine shall go to sub-state COMM_NO_COM_NO_PENDING_REQUEST.](())

[SWS_ComM_00313] [On entering state COMM_NO_COMMUNICATION by default after initialization, ComM module shall not indicate the mode change to users via RTE or BswM.](())

Rationale for [SWS ComM 00313](#): The RTE is not yet initialized at this point in time.

[SWS_ComM_00073] [On entering state COMM_NO_COMMUNICATION the ComM channel state machine shall switch off the transmission and reception capability. This shall be performed by the ComM channel state machine requesting the corresponding Communication Mode from the Bus State Manager module (XXSM_RequestComMode(network:=<channel state machine's network>, mode:= COMM_NO_COMMUNICATION, see [SWS ComM 00829](#))).](())

Rationale for [SWS ComM 00073](#): The COMM_NO_COMMUNICATION mode forbids sending and receiving of bus communication PDUs for the corresponding channels.

[SWS_ComM_00288] [On entering state COMM_NO_COMMUNICATION and configuration parameter ComMNmVariant=FULL (see [ECUC ComM 00568](#)) the ComM module shall request release of the network from the Network Management module, Nm_NetworkRelease().](SRS_ModeMgm_09132)

Note: Nm_NetworkRelease is needed if ComM has requested the NM (Nm_NetworkRequest or Nm_PassiveStartup) for that channel before and has not yet released it.

Rationale for [SWS ComM 00073](#), [SWS COMM 00288](#), [SWS ComM 00875](#) and [SWS ComM 00876](#): FlexRay shutdown cannot be interrupted to avoid partial networks.

Comment: In state COMM_NO_COMMUNICATION ComM channel state machine may not request bus communication for the configured channel from the Bus State Manager module.

Use Case for above Comment: The ECU is performing control functions locally without participation in bus communication.

Comment: The communication mode is local for one channel, thus the ECU may still communicate via other channels.

7.2.2.1 COMM_NO_COM_NO_PENDING_REQUEST sub-state

[SWS_ComM_00875] [In sub-state `COMM_NO_COM_NO_PENDING_REQUEST` and user requests `COMM_FULL_COMMUNICATION` and communication limitation is disabled (see Section 7.4.1), the ComM channel state machine shall immediately switch to sub-state `COMM_NO_COM_REQUEST_PENDING`.]()

[SWS_ComM_00876] [In sub-state `COMM_NO_COM_NO_PENDING_REQUEST`, configuration parameter `ComMNmVariant=FULL|LIGHT|NONE` (see [ECUC ComM 00568](#)) and DCM indicate `ComM_DCM_ActiveDiagnostic` (see [SWS ComM 00873](#)), the ComM channel state machine shall immediately switch to sub-state `COMM_NO_COM_REQUEST_PENDING`.]()

Rationale for [SWS ComM 00876](#): A potential communication limitation (see Section 7.4.1) shall temporarily be inactive during an active diagnostic session (see [SWS ComM 00182](#))

Note for [SWS ComM 00876](#): For diagnostic activation it is assumed that diagnostic tester keeps the bus awake, therefore no special handling needed for managed channels.

[SWS_ComM_00893] [If `ComM_EcuM_WakeUpIndication` is called in sub-state `COMM_NO_COM_NO_PENDING_REQUEST` and configuration parameter `ComMSynchronousWakeUp` is set to `FALSE` (see [ECUC ComM 00695](#)), the ComM module shall switch the requested ComM channel state machine (resp. channels) to sub-state `COMM_NO_COM_REQUEST_PENDING`. If the indicated ComM channel is a managed channel, then the ComM channel state machine of the referencing managing channel (see [ECUC ComM 00893](#)) shall also be switched to sub-state `COMM_NO_COM_REQUEST_PENDING`.]([SRS_ModeMgm_09087](#))

[SWS_ComM_00894] [In sub-state `COMM_NO_COM_NO_PENDING_REQUEST` and the NM module indicates a restart, `ComM_Nm_RestartIndication()` [SWS ComM 00792](#), the ComM channel state machine shall immediately switch to sub-state `COMM_NO_COM_REQUEST_PENDING`.]([SRS_ModeMgm_09087](#))

Rationale for [SWS ComM 00893](#) and [SWS ComM 00894](#): It must be guaranteed that communication starts as soon as possible after a bus wake up.

Comment: The ComM channel state machine switches immediately to sub-state `COMM_FULL_COM_NETWORK_REQUESTED` after entering the `COMM_FULL_COMMUNICATION` state. If no user requests `COMM_FULL_COMMUNICATION` mode, the AUTOSAR NM resp. the ComM module timer for `ComMTMinFullComModeDuration` ([ECUC ComM 00557](#)) prevent toggling between `COMM_NO_COMMUNICATION` and `COMM_FULL_COMMUNICATION` to overcome the init-/start-up time of the system, before possible user requests occur.

[SWS_ComM_00694] [If ComM_EcuM_WakeUpIndication is called in sub-state COMM_NO_COM_NO_PENDING_REQUEST and configuration parameter ComMSynchronousWakeUp is set to TRUE (see [ECUC ComM_00695](#)), the ComM module shall switch all ComM channel state machines (resp. channels) to sub-state COMM_NO_COM_REQUEST_PENDING.] (SRS_ModeMgm_09248)

[SWS_ComM_01014] [If ComM_EcuM_PNCWakeUpIndication(<PNC>) (see [SWS ComM_91001](#)) is called in sub-state COMM_NO_COM_NO_PENDING_REQUEST and configuration parameters ComMSynchronousWakeUp is set to FALSE (see [ECUC ComM_00695](#)) and ComMPncSupport is set to TRUE (see [ECUC ComM_00839](#)), the ComM module shall switch these ComM channel state machines (resp. channels) which are referenced by the PNC to sub-state COMM_NO_COM_REQUEST_PENDING.] (SRS_ModeMgm_09248)

Note for [SWS ComM_01014](#): This includes ComM channel state machines of managing channels, which are referenced by the indicated managed channels, as ComMPncS reference always both types (see [31] constr_3484)

[SWS_ComM_01015] [If ComM_EcuM_PNCWakeUpIndication(<PNC>) (see [SWS ComM_91001](#)) is called in sub-state COMM_NO_COM_NO_PENDING_REQUEST and configuration parameters ComMSynchronousWakeUp is set to TRUE (see [ECUC ComM_00695](#)) and ComMPncSupport is set to TRUE (see [ECUC ComM_00839](#)), the ComM module shall switch all ComM channel state machines (resp. channels) to sub-state COMM_NO_COM_REQUEST_PENDING.] (SRS_ModeMgm_09248)

7.2.2.2 COMM_NO_COM_REQUEST_PENDING sub-state

[SWS_ComM_00895] [In sub-state COMM_NO_COM_REQUEST_PENDING the ComM channel state machine shall evaluate its corresponding CommunicationAllowed flag, stored and set according to [SWS ComM_00884](#) and [SWS ComM_00885](#). If evaluated to CommunicationAllowed is set to TRUE, the ComM channel state machine shall immediately switch to state COMM_FULL_COMMUNICATION.]()

[SWS_ComM_00897] [In sub-state COMM_NO_COM_REQUEST_PENDING and no longer any valid pending request for COMM_FULL_COMMUNICATION, the ComM channel state machine shall switch back to default sub-state COMM_NO_COM_NO_PENDING_REQUEST.] (SRS_ModeMgm_09083)

Rationale for [SWS ComM_00897](#): This enable the possibility to switch back to default sub-state if communication for some reason was never allowed. E.g. transition to COMM_NO_COM_REQUEST_PENDING triggered by user request for ComM_RequestComMode(<user>, COMM_FULL_COMMUNICATION) (see [SWS ComM_00871](#)) or DCM indicated ComM_DCM_ActiveDiagnostic(<channel>) (see [SWS ComM_00873](#)), but now canceled with ComM_RequestComMode(<user>, COMM_NO_COMMUNICATION) (see

[SWS_ComM_00871](#)) or DCM `ComM_DCM_InactiveDiagnostic(<channel>)` (see [SWS_ComM_00874](#)).

7.2.3 Behaviour in state `COMM_SILENT_COMMUNICATION`

[SWS_ComM_00071] [On entering state `COMM_SILENT_COMMUNICATION` the ComM channel state machine shall switch off the transmission capability (and keep reception capability on). This shall be performed by the ComM channel state machine requesting the corresponding Communication Mode from the Bus State Manager module (`XXSM_RequestComMode(network:=<channel state machine's network>, mode:= COMM_SILENT_COMMUNICATION)` [SWS_ComM_00829](#)).]()

Rationale for [SWS_ComM_00071](#): The `COMM_SILENT_COMMUNICATION` mode permits receiving of bus communication PDUs and forbids sending of bus communication PDUs.

Comment: It may happen that nothing is received (e.g. during bus off) despite receiving capability is switched on.

Use Case: Shut down coordination with means of the NM module (prepare bus sleep state).

[SWS_ComM_00877] [In state `COMM_SILENT_COMMUNICATION` and user requests `COMM_FULL_COMMUNICATION` and communication limitation is disabled (see Section 7.4.1), the ComM channel state machine shall switch to state `COMM_FULL_COMMUNICATION`.]([SRS_ModeMgm_09246](#))

[SWS_ComM_00878] [In state `COMM_SILENT_COMMUNICATION`, configuration parameter `ComMNmVariant=FULL|LIGHT|NONE` ([ECUC_ComM_00568](#)) and DCM indicate `ComM_DCM_ActiveDiagnostic(SWS_ComM_00873)`, the ComM channel state machine shall switch to state `COMM_FULL_COMMUNICATION`.]()

Rationale for [SWS_ComM_00878](#): A potential communication limitation (see Section 7.4.1) shall temporarily be inactive during an active diagnostic session, see [SWS_ComM_00182](#)

[SWS_ComM_00295] [In state `COMM_SILENT_COMMUNICATION` and the Network Manager module indicates `ComM_Nm_BusSleepMode()` ([SWS_ComM_00392](#)), the ComM channel state machine shall switch to state `COMM_NO_COMMUNICATION`.]()

[SWS_ComM_00296] [In state `COMM_SILENT_COMMUNICATION` and the Network Manager module indicates `ComM_Nm_NetworkMode()` ([SWS_ComM_00390](#)), the ComM channel state machine shall switch to state `COMM_FULL_COMMUNICATION` and sub-state `COMM_FULL_COM_READY_SLEEP`.]()

7.2.4 Behaviour in state COMM_FULL_COMMUNICATION

[SWS_ComM_00899] [On entering state COMM_FULL_COMMUNICATION the ComM channel state machine shall go to sub-state COMM_FULL_COM_NETWORK_REQUESTED, if not a specific sub-state is specified in the transition.]()

Rationale for [SWS ComM 00899](#): When switching from COMM_SILENT_COMMUNICATION, the ComM channel state machine can switch directly to sub-state COMM_FULL_COM_READY_SLEEP, if specified in the transition, see [SWS ComM 00296](#).

[SWS_ComM_00069] [On entering state COMM_FULL_COMMUNICATION the ComM channel state machine shall switch on the transmission and reception capability. This shall be performed by the ComM channel state machine requesting the corresponding Communication Mode from the Bus State Manager module:

- If Communication Mode COMM_FULL_COMMUNICATION was requested , then `<Bus>SM_RequestComMode(network:=<channel state machine's network>, mode:= COMM_FULL_COMMUNICATION)` shall be called
- If Communication Mode COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST was requested and ComMWakeupSleepRequestEnabled of the ComM channel is set to TRUE, then `<Bus>SM_RequestComMode (network:=<channel state machine's network>, mode:= COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST)` shall be called
- If Communication Mode COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST was requested and ComMWakeupSleepRequestEnabled of the ComM channel is set to FALSE or not available, then `<Bus>SM_RequestComMode (network:=<channel state machine's network>, mode:= COMM_FULL_COMMUNICATION)` shall be called

] (SRS_ModeMgm_09268)

Rationale for [SWS ComM 00069](#): The COMM_FULL_COMMUNICATION or COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST mode permits sending and receiving of bus communication PDUs for the corresponding channels.

[SWS_ComM_01057] [Every time a ComM channel is requested with COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST and ComMWakeupSleepRequestEnabled of the ComM channel is set to TRUE, ComM shall request the corresponding network of the ComM channel by calling `<Bus>SM_RequestComMode (COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST)`, even if the ComM channel is already in state COMM_FULL_COMMUNICATION. If ComMWakeupSleepRequestEnabled of the ComM channel is set to FALSE or not available, the ComM shall ignore the request.] (SRS_ModeMgm_09268)

Note: The re-trigger of the <Bus>SM state machine is used to trigger a wake-up on the network, if the used hardware is supporting such a functionality (e.g. Ethernet hardware compliant to OA TC10 (see [33]))

[SWS_ComM_00637] [In state `COMM_FULL_COMMUNICATION` and the Network Manager module indicates `ComM_Nm_BusSleepMode()` ([SWS ComM_00392](#)), the ComM channel state machine shall switch to state `COMM_NO_COMMUNICATION`.]()

Rationale for [SWS ComM_00637](#): A user may request to keep the bus awake "too late" (NM is not able to send a vote to keep the bus awake because the cluster already agreed to shutdown).

[SWS_ComM_01018] [In state `COMM_FULL_COMMUNICATION` and configuration parameter `ComMNmVariant=SLAVE_ACTIVE | SLAVE_PASSIVE` and the Bus State Manager module indicates `ComM_BusSm_BusSleepMode()` (see [SWS ComM_91000](#)), the ComM channel state machine shall switch to state `COMM_NO_COMMUNICATION`.]([SRS_ModeMgm_09266](#), [SRS_ModeMgm_09267](#))

[SWS_ComM_00826] [In `COMM_FULL_COMMUNICATION` and configuration parameter `ComMNmVariant=FULL|PASSIVE` ([ECUC ComM_00568](#)) and the Network Manager module indicates `ComM_Nm_PrepareBusSleepMode()` ([SWS ComM_00391](#)), the ComM state machine shall switch to state `COMM_SILENT_COMMUNICATION`.]()

Rationale for [SWS ComM_00826](#): `ComM_Nm_PrepareBusSleepMode()` cannot be received before an active request is released via `Nm_NetworkRelease()`, and a `PASSIVE` channel cannot be woken up by an active wake-up, therefore it is safe to assume that the transition is always valid.

7.2.4.1 `COMM_FULL_COM_NETWORK_REQUESTED` sub-state

[SWS_ComM_00886] [On entering sub-state `COMM_FULL_COM_NETWORK_REQUESTED` and configuration parameter `ComMNmVariant=LIGHT|NONE` ([ECUC ComM_00568](#)), the timer for `ComMTMinFullComModeDuration` ([ECUC ComM_00557](#)) shall be started.]()

[SWS_ComM_00665] [On entering sub-state `COMM_FULL_COM_NETWORK_REQUESTED` from `COMM_NO_COM_REQUEST_PENDING` and `EcuM` module has indicated a wake-up by `ComM_EcuM_WakeUpIndication(<channel>)` (see [SWS ComM_00275](#)) or by `ComM_EcuM_PNCWakeUpIndication(<PNC>)` (see [SWS ComM_91001](#)), the ComM module shall request `Nm_PassiveStartup(<channel>)` from the Network Management. If the indicated ComM channel is a managed channel, the ComM module shall request `Nm_PassiveStartup(<referencing managing channel>)` (see [ECUC ComM_00893](#)) from the Network Management.]()

[SWS_ComM_01016] [If the indicated ComM channel is a managed channel, the ComM module shall request Nm_PassiveStartup(<referencing managing channel>) (see [ECUC ComM_00893](#)) from the Network Management.]()

[SWS_ComM_00902] [On entering sub-state COMM_FULL_COM_NETWORK_REQUESTED and Nm module has indicated a restart, ComM_Nm_RestartIndication(<channel>) ([SWS ComM_00792](#)), the ComM module shall request Nm_PassiveStartup(<channel>) from the Network Management.]()

[SWS_ComM_00903] [On entering sub-state COMM_FULL_COM_NETWORK_REQUESTED and Nm module has indicated a Network start, ComM_Nm_NetworkStartIndication(<channel>) ([SWS ComM_00383](#)), the ComM module shall request Nm_PassiveStartup(<channel>) from the Network Management.]()

Comment for [SWS ComM_00903](#):

This is not a “normal” transition to COMM_FULL_COMMUNICATION, ComM handle ComM_Nm_NetworkStartIndication() as “race condition” error (see section 7.7.1)

[SWS_ComM_00869] [On entering sub-state COMM_FULL_COM_NETWORK_REQUESTED from another state or substate, if configuration parameter ComMNmVariant=FULL ([ECUC ComM_00568](#)) and if a user has requested ComM_RequestComMode(<user>, COMM_FULL_COMMUNICATION) ([SWS ComM_00110](#)) the ComM module shall request Nm_NetworkRequest(<channel>) from the Network Management for the corresponding NM channel.](SRS_ModeMgm_00049)

Note: Additionally Nm_NetworkRequest may be invoked due to [SWS ComM_00980](#).

[SWS_ComM_00870] [On entering sub-state COMM_FULL_COM_NETWORK_REQUESTED, if configuration parameter ComMNmVariant=FULL ([ECUC ComM_00568](#)) and the DCM has indicated ComM_DCM_ActiveDiagnostic(<channel>) ([SWS ComM_00873](#)), the ComM module shall request Nm_NetworkRequest(<channel>) from the Network Management for the corresponding NM channel.](SRS_ModeMgm_00049)

[SWS_ComM_00889] [In sub-state COMM_FULL_COM_NETWORK_REQUESTED and configuration parameter ComMNmVariant=LIGHT|NONE ([ECUC ComM_00568](#)) and timer for ComMTMinFullComModeDuration([ECUC ComM_00557](#)) has expired and no user request ComM_RequestComMode(<user>, COMM_FULL_COMMUNICATION) and the DCM does not indicate ComM_DCM_ActiveDiagnostic(<channel>) ([SWS ComM_00873](#)), the ComM channel state machine shall switch to sub-state COMM_FULL_COM_READY_SLEEP.]()

Rationale for [SWS ComM_00889](#):

As long as timer for `ComMTMinFullComModeDuration` has not expired the sub-state shall be kept, to prevent toggling.

[SWS_ComM_00888] [In sub-state `COMM_FULL_COM_NETWORK_REQUESTED` and configuration parameter `ComMNmVariant=FULL` (see [ECUC ComM_00568](#)) and no user request `ComM_RequestComMode(<user>, COMM_FULL_COMMUNICATION)` and the DCM does not indicate `ComM_DCM_ActiveDiagnostic(<channel>)` (see [SWS ComM_00873](#)), the ComM channel state machine shall switch to sub-state `COMM_FULL_COM_READY_SLEEP`.]()

Rationale for [SWS ComM_00888](#):

No timer needed if AUTOSAR NM is used. This avoids redundant functionality because AUTOSAR NM also ensures this functionality

[SWS_ComM_01017] [In sub-state `COMM_FULL_COM_NETWORK_REQUESTED` and configuration parameter `ComMNmVariant=SLAVE_ACTIVE` ([ECUC ComM_00568](#)) and no user request `ComM_RequestComMode(<user>, COMM_FULL_COMMUNICATION)`, the ComM channel state machine shall switch to sub-state `COMM_FULL_COM_READY_SLEEP`.]([SRS_ModeMgm_09266](#))

[SWS_ComM_00915] [In sub-state `COMM_FULL_COM_NETWORK_REQUESTED` and configuration parameter `ComMNmVariant=PASSIVE | SLAVE_PASSIVE` ([ECUC ComM_00568](#)), the ComM channel state machine shall switch to sub-state `COMM_FULL_COM_READY_SLEEP`.]([SRS_ModeMgm_09267](#))

[SWS_ComM_00890] [In sub-state `COMM_FULL_COM_NETWORK_REQUESTED` and the DCM does not indicate `ComM_DCM_ActiveDiagnostic(<channel>)` (see [SWS ComM_00873](#)) and communication limitation is requested (see section 7.4.1), ComM channel state machine shall immediately switch to sub-state `COMM_FULL_COM_READY_SLEEP` and cancel the timer for `ComMTMinFullComModeDuration`.]()

7.2.4.2 `COMM_FULL_COM_READY_SLEEP` sub-state

[SWS_ComM_00133] [On entering sub-state `COMM_FULL_COM_READY_SLEEP` and configuration parameter `ComMNmVariant=FULL` (see [ECUC ComM_00568](#)), the ComM module shall request `Nm_NetworkRelease()` from the Network Management for the corresponding NM channels.]()

[SWS_ComM_00891] [On entering sub-state `COMM_FULL_COM_READY_SLEEP` and configuration parameter `ComMNmVariant=LIGHT` (see [ECUC ComM_00568](#)), the timer for `ComMNmLightTimeout` (see [ECUC ComM_00606](#)) shall be started.]()

[SWS_ComM_00610] [In sub-state `COMM_FULL_COM_READY_SLEEP` and configuration parameter `ComMNmVariant=LIGHT` (see [ECUC ComM_00568](#)) and

the timer for `ComMNmLightTimeout` (see [ECUC ComM 00606](#)) has expired, the ComM channel state machine shall switch to state `COMM_NO_COMMUNICATION`.]()

[SWS_ComM_00671] [In sub-state `COMM_FULL_COM_READY_SLEEP` and configuration parameter `ComMBusType=COMM_BUS_TYPE_INTERNAL` ([ECUC ComM 00567](#)), the ComM channel state machine shall immediately switch to state `COMM_NO_COMMUNICATION`.]()

[SWS_ComM_00882] [In sub-state `COMM_FULL_COM_READY_SLEEP` and a user request `COMM_FULL_COMMUNICATION` and communication limitation is disabled (see Section 7.4.1), the ComM channel state machine shall immediately switch to sub-state `COMM_FULL_COM_NETWORK_REQUESTED`.]()

[SWS_ComM_00883] [In sub-state `COMM_FULL_COM_READY_SLEEP`, configuration parameter `ComMNmVariant=FULL|LIGHT|NONE` ([ECUC ComM 00568](#)) and DCM indicate `ComM_DCM_ActiveDiagnostic` ([SWS ComM 00873](#)), the ComM channel state machine shall switch to sub-state `COMM_FULL_COM_NETWORK_REQUESTED`.]()

Rationale for [SWS ComM 00883](#): A potential communication limitation (see Section 7.4.1) shall temporarily be inactive during an active diagnostic session, see [SWS ComM 00182](#)

[SWS_ComM_00892] [In sub-state `COMM_FULL_COM_READY_SLEEP` and configuration parameter `ComMNmVariant=LIGHT` ([ECUC ComM 00568](#)) and a switch to sub-state `COMM_FULL_COM_NETWORK_REQUESTED`, due to request for `COMM_FULL_COMMUNICATION` according to requirements in [SWS ComM 00882](#) or [SWS ComM 00883](#), the timer for `ComMNmLightTimeout` ([ECUC ComM 00606](#)) shall be canceled.]()

7.3 ComM User to PNC Relations

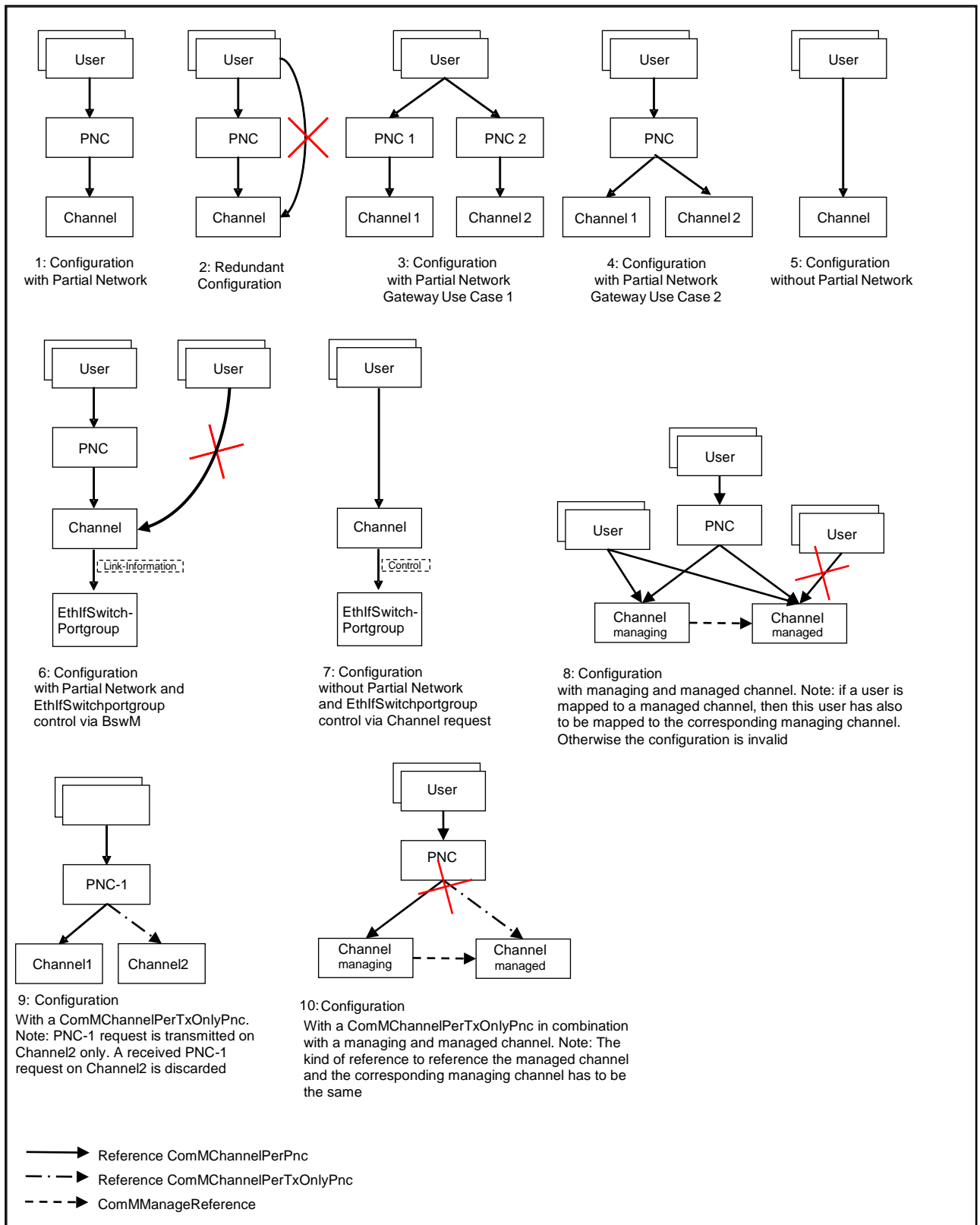


Figure 8: User to Partial network and channel Mapping Use Cases

[SWS_ComM_00994] [No restrictions from the configuration of the BusNm Filter for partial networking shall apply to ComM user assignment to PNCs.]()

Comment: The BusNM Filter configuration shall be independent from the ComM PNC configuration.

Rational: This enables waking up a PNC without being a member of the PNC, e.g. if a node just triggers a wake up of a PNC but the node is not kept awake by the PNC and other nodes keep the PNC awake

[SWS_ComM_00995] [It shall be possible to map a configurable amount of ComMUsers to one or more ComM channels using the parameter ComMUserPerChannel.](SRS_ModeMgm_09133, SRS_ModeMgm_09090)

Comment:

- 1.) The existing mapping of ComM users to system channels shall still be possible for backward compatibility. (i.e. the configuration containers will stay untouched)
- 2.) In a multi channel system each user can be assigned to one or more channels. If the user requests a mode, all channels assigned to this user, shall switch to the corresponding mode. All other channels shall not be affected.

[SWS_ComM_00912] [It shall be possible to map a configurable amount of ComMUsers to one or more PNCs using the parameter ComMUserPerPnc (see [ECUC ComM 00876](#)).]()

[SWS_ComM_00913] {OBSOLETE replaced by [SWS_ComM_01094]} [It shall be possible to map a configurable amount of PNC(s) to a configurable amount of ComM channels using the parameter ComMChannelPerPnc (see [ECUC ComM 00880](#)). The mapping shall be possible for all ComMChannels in combination with the following ComMNmVariants:

- ComMVariant=FULL
- ComMVariant=LIGHT, if the ComMChannel is in the role of a managed ComMChannel and the corresponding managing ComMChannel is also mapped to this PNC (see also 7.2.1)

]()

[SWS_ComM_01094] {DRAFT} [It shall be possible to map a configurable amount of PNC(s) to a configurable amount of ComM channels by using the parameter ComMChannelPerPnc (see [ECUC ComM 00880](#) :) or ComMChannelPerTxOnlyPnc (see [ECUC ComM 00900](#) :). The mapping shall be possible for all ComMChannels in combination with the following ComMNmVariants:

- ComMVariant=FULL
- ComMVariant=LIGHT, if the ComMChannel is in the role of a managed ComMChannel and the corresponding managing ComMChannel is also mapped to this PNC (see also 7.2.1)

]()

[SWS_ComM_00996] [It shall not be possible to map a ComMUsers to a PNC and in addition to a ComM channel which is already referenced by the PNC (see figure 8 Use Case 2)]()

Rational: Avoid redundant configuration since the channel is implicitly already referenced by the PNC.

[SWS_ComM_CONSTR_00001] [ComM channel's that are referenced by a PNC are not allowed to be referenced by any ComMUsers, if the PNC references at least one EthIfSwitchPortGroup (see figure 8 “use Case 6”). A configuration tool shall reject such a configuration as invalid (error). This constraint is only valid for a host ecu that control an Ethernet switch. In all other UseCases ComMChannels can be referenced by a PNC's and ComMUsers.]()

Rational: If using PNC and SwitchPortGroups were derived (EcuInstance.ethSwitchPortGroupDerivation is set to TRUE), then the SwitchPortGroups are switched by the EthIf_SwitchPortGroupRequestMode API and not by a channel request.

[SWS_ComM_CONSTR_00002] [If a ComM user reference a managed channel, then this ComM user shall also reference the corresponding managing channel. Otherwise the configuration is invalid. A configuration tool shall reject a configuration as invalid (error), if a user references a managed channel without referencing the corresponding managing channel.]()

[SWS_ComM_CONSTR_00003] {DRAFT} [ComM channels with ComMNmVariant = SLAVE_PASSIVE are not allowed to be referenced by any ComMUser or PNC. A configuration tool shall reject such a configuration as invalid (error).] (SRS_ModeMgm_09266)

Rational: ComM channels with ComMNmVariant = SLAVE_PASSIVE shall always follow the communication request of their communication master and are not allowed to request the corresponding master to wake-up the communication channel.

[SWS_ComM_CONSTR_00005] {DRAFT} [A PNC shall reference the same ComM channel exclusively either by using ComMChannelPerPnc or ComMChannelPerTxOnlyPnc. Otherwise the configuration is invalid. A configuration tool shall reject such a configuration as invalid (error).] ()

[SWS_ComM_CONSTR_00006] {DRAFT} [The kind of reference (either ComMChannelPerPnc or ComMChannelPerTxOnlyPnc) from a PNC to a managed channel and the corresponding managing channel shall be the same. Otherwise the configuration is invalid. A configuration tool shall reject such a configuration as invalid (error).] ()

7.4 Extended functionality

[SWS_ComM_00470] [The extended functionality described in this chapter shall be individually configurable during runtime per feature (e.g. enable wake up inhibition but disable limitation to no communication).]()

Rationale for [SWS ComM 00470](#): During runtime a change in the inhibition / limitation strategy is required in order to cope with changing conditions.

Use Case: Change the wakeup inhibition via diagnostics.

Comment: Configurable with parameter `ComMEcuGroupClassification` (see [ECUC ComM 00563](#)).

7.4.1 Communication inhibition

Note:

1. The purpose of mode inhibition is to limit the communication capabilities. For details see Section 7.4.1.1 and Section 7.4.1.2.
2. The following parameters are relevant to communication inhibition and have relationship to APIs described below:
 - a. `ComMNoCom`: "request bit" of mode inhibition (limit to NoCom), can be controlled by `ComM_LimitChannelToNoComMode()` and `ComM_LimitECUToNoComMode()`, only if `ComMEcuGroupClassification` enable this functionality (see [ECUC ComM 00563](#), [SWS ComM 00163](#), [SWS ComM 00124](#)).
 - b. `ComMNoWakeup`: "request bit" of mode inhibition (wakeup inhibition), can be controlled by `ComM_PreventWakeUp()`, only if `ComMEcuGroupClassification` enable this functionality (see [ECUC ComM 00563](#), [SWS ComM 00156](#)).
 - c. `ComMEcuGroupClassification`: "mask bits" of mode inhibition behavior, can be controlled by `ComM_SetECUGroupClassification()`, regardless of `ComMNoCom` and `ComMNoWakeup` values

[SWS_ComM_00301] [The ComM module shall offer interfaces to request and release the corresponding mode inhibitions.]()

Comment: The ComM module doesn't care about who requests the mode inhibition but it is not a "normal" SW-C. It is a privileged SW-C or an OEM specific BSW.

[SWS_ComM_00488] [It shall be possible to enable and disable the mode inhibition for each channel (channel state machine) independently. This functionality shall not be used by the ComM module itself.]()

[SWS_ComM_00839] [The ComM module shall store the status of the user requests.]()

Comment: `SWS_ComM_00839` describes the desired behaviour during an active mode limitation.

[SWS_ComM_00840] [The ComM module shall store the updated status of the user requests if a user releases a request during an active mode inhibition.]()

Rationale for [SWS ComM 00840](#): User requests shall be granted if the inhibition gets disabled.

Comment: Amount of active user requests from different users. [SWS ComM_00840](#) describes the desired behaviour during an active mode limitation.

[SWS_ComM_00182] [The communication inhibition shall get temporarily inactive during an active diagnostic session.]()

Rationale for [SWS ComM_00182](#): ECUs must not fall asleep during an active diagnostic session.

Comment: The DCM indicates the start of an active diagnostic session with `ComM_DCM_ActiveDiagnostic(<channel>)` ([SWS ComM_00873](#)) and the end of a diagnostic session with `ComM_DCM_InactiveDiagnostic(<channel>)` ([SWS ComM_00874](#)).

7.4.1.1 Bus wake up inhibition

Information: Bus wake up inhibition in context of the ComM module means that the ComM module should take precautions against awaking other ECUs by starting the communication.

Rationale: Awaking other ECUs by communication should be avoided because it is assumed that the ECU wakes up the bus because of an error (e.g. broken sensor).

Use Case: An error was detected on signal path of an active wake up line and this non reliable wake-up-source should not be able to awake the whole system anymore. An SW-C that controls error-reactions could set the wake up inhibition-status of related communication channels that usually get communication-requests from SW-Cs as the consequence of this event. This corrupts the forwarding of communication system-wide, based on unreliable wake up events. Or in case of application-specific system control, there is an SW-C that should switch off forwarding system wide wakeup's by communication under conditions like e.g. transport mode.

[SWS_ComM_00302] [Bus wake up Inhibition shall be performed by ignoring user requests.](SRS_ModeMgm_09089)

Comment: Ignoring user requests means accepting the requests but not executing them due to mode inhibition. The “highest win” strategy would apply immediately as soon as mode inhibition is switched off (see [SWS ComM_00839](#) and [SWS ComM_00840](#)).

[SWS_ComM_00218] [A communication request (`COMM_FULL_COMMUNICATION`) by a user shall be inhibited if the ComM Inhibition status is equal to `ComMNoWakeUp` is set to TRUE (see [ECUC ComM_00569](#)) for the corresponding channel and the current state of the channel is `COMM_NO_COMMUNICATION` or `COMM_SILENT_COMMUNICATION`]()

Rationale for [SWS ComM 00218](#): The inhibition should not get active, if the inhibition-status is set but the communication channel is already active.

[SWS_ComM_00219] [The inhibition shall not get active if the current communication state is `COMM_FULL_COMMUNICATION`.]()

Rationale for [SWS ComM 00219](#): The bus is already awake if the current communication state is `COMM_FULL_COMMUNICATION`.

[SWS_ComM_00066] [The ComM module shall never inhibit the “passive wake-up” capability.](SRS_ModeMgm_09071)

Rationale for [SWS ComM 00066](#): It must be always possible to react on bus wake ups indicated by the EcuM module.

Comment: Reception is switched off in `COMM_NO_COMMUNICATION` mode but the wake up capability is switched on.

[SWS_ComM_00157] [ComMNoWakeup status must be stored non volatile.](SRS_ModeMgm_09089)

Rationale for [SWS ComM 00157](#): Information must be available during start-up, before the communication is active (“Full Communication” mode entered). Changing or query is only possible after start-up with active communication (usually the “master”, who decides if the inhibition is active or not, is not on the same ECU).

[SWS_ComM_00625] [The status of the user requests shall also be updated if a user releases a request.](SRS_ModeMgm_09155)

7.4.1.2 Limit to `COMM_NO_COMMUNICATION` mode

[SWS_ComM_00303] [If the current state is `COMM_FULL_COM_NETWORK_REQUESTED` and when mode limitation to `COMM_NO_COMMUNICATION` has been requested for the corresponding channel, ComM module shall switch to `COMM_FULL_COM_READY_SLEEP` state to initiate a shutdown despite any user requests for `COMM_FULL_COMMUNICATION`.](SRS_ModeMgm_09071)

Rationale for [SWS ComM 00303](#): Forcing into `COMM_NO_COMMUNICATION` mode is needed to shut down software components, which keeps the bus awake.

Comment for [SWS ComM 00303](#): Limit to `COMM_NO_COMMUNICATION` will only be performed if a channel was request actively. In that case all current user requests for full communication or even new requests will be ignored (see also [SWS_ComM_00215](#)). The limit to no communication will not be performed, if a ComM channel is remotely kept awake due to a passive wakeup.

[SWS_ComM_00842] [When `ComM_LimitChannelToNoComMode()` is called, ComM module shall update the inhibition status (limitation to

COMM_NO_COMMUNICATION) for the corresponding channel.](SRS_ModeMgm_09071)

Note: An update of the inhibition status due to a request for limit to COMM_NO_COMMUNICATION has to be performed always, independent of the current state.

[SWS_ComM_00355] [If ComMResetAfterForcingNoComm is set to TRUE (see **ECUC_ComM_00558** :) and when ComM enters COMM_NO_COMMUNICATION after state transition from COMM_FULL_COM_NETWORK_REQUESTED to COMM_FULL_COM_READY_SLEEP has been forced due to mode limitation to COMM_NO_COMMUNICATION request (see **[SWS_ComM_00303]**), then ComM shall call BswM_ComM_InitiateReset().](SRS_ModeMgm_09071)

Note: A call of BswM_ComM_InitiateReset() is the trigger for an ECU reset which has to be executed as soon as possible, depending on further needed actions (e.g. storing all NvM blocks).

Rationale: It is assumed that a faulty user will not release his "Full Communication" request without a re-initialization. Keeping the "Full Communication" request active leads to a toggling between network shutdown and network startup.

Use Case: It is assumed that a faulty ECU keeps the bus awake. As a consequence a "network master" decides to force all ECUs to go to sleep.

[SWS_ComM_00215] [When mode limitation to COMM_NO_COMMUNICATION has been requested, ComM module shall ignore all user requests with COMM_FULL_COMMUNICATION for the corresponding channel.](SRS_ModeMgm_09071)

[SWS_ComM_00582] [The ComM module shall clear the user requests after all the channels that belong to the corresponding user enter COMM_NO_COMMUNICATION mode.](SRS_ModeMgm_09078)

Rationale for [SWS ComM 00582](#): Stored (faulty) user requests, which are assumed to keep the bus awake, must be cleared.

Description: The ComM module shall reload the default value of the ComM inhibition status from ComMNoCom (see [ECUC ComM 00571](#)) during initialization.

Comment: The current ComMNoCom status for each channel shall not be stored persistently. SWS_ComM_00582 describes the desired behaviour after an executed mode limitation.

7.5 Bus communication management

[SWS_ComM_00402] [The ComM module shall use the corresponding interfaces of the Bus State Manager modules to control the communication capabilities.](SRS_ModeMgm_09071)

[SWS_ComM_00664] [The ComM module shall omit calls to control the communication capabilities if configuration parameter ComMBusType=COMM_BUS_TYPE_INTERNAL ([ECUC ComM 00567](#)).](SRS_ModeMgm_09168)

Rationale for [SWS ComM 00664](#): Internal communication has no corresponding bus interface.

7.6 Network management dependencies

[SWS_ComM_00599] [The ComM module shall support the shutdown synchronization variants (configured with ComMNmVariant, see [ECUC ComM 00568](#)) LIGHT, SLAVE_ACTIVE, SLAVE_PASSIVE, PASSIVE and FULL described in Table 3.](SRS_ModeMgm_09132)

Comment: Only variant FULL and PASSIVE guarantees a synchronized shutdown between all nodes of a network. Note that since the Nmlf cannot start the synchronized shutdown of coordinated networks before all networks are ready to go to sleep, requests from ComM to Nmlf to release network communication on such a coordinated bus will be considered, but not always acted on directly. The Nmlf will still answer with E_OK, but network will not be released until all coordinated networks are ready to go to sleep.

NM variant	Keep bus awake capability	Shutdown synchronization
NONE		No shutdown synchronization by ComM. Shutdown by switching off the power of the ECU.
SLAVE_ACTIVE	No (but the corresponding master could trigger a wake-up based on a slave request for a wake-up. E.g. the LIN State Manager of a LIN master restarts wake-up repetition)	Synchronized by its master (e.g. LIN master)
SLAVE_PASSIVE	No (the slave will always follow the communication request of the corresponding master. The slave has no possibility to request a wake-up on the corresponding communication channel.	Synchronized by its master (e.g. ComM channel with ComMBusType set to COMM_BUS_TYPE_ETH and used Ethernet hardware is compliant to OA TC10 (see [33]))
LIGHT		Shutdown synchronization by ComM with means of a timeout (configured with ComMNmLightTimeout, ECUC ComM 00606)
PASSIVE	ECU is not allowed to keep the bus awake	Shutdown synchronization by ComM with means of AUTOSAR NM.
FULL	ECU is allowed to keep the bus awake.	Shutdown synchronization by ComM with means of AUTOSAR NM.

Table 3: Network management variants supported by the Communication Manager Module

Comment: A synchronized shutdown is not possible with the LIGHT variant thus the ECU may continuously restart ("toggle") because of a message from a node shutting down later.

[SWS_ComM_00602] [The ComM module shall omit calls of NM services if configuration parameter `ComMNmVariant = LIGHT | SLAVE_ACTIVE | SLAVE_PASSIVE | NONE` (see [ECUC ComM_00568](#)).](SRS_ModeMgm_09132)

Rationale for [SWS ComM_00602](#): NM services are not available if no NM is available.

[SWS_ComM_00667] [The ComM module shall omit to call `Nm_NetworkRequest()` from NM if configuration parameter `ComMNmVariant = LIGHT | SLAVE_ACTIVE | SLAVE_PASSIVE | NONE` (see [ECUC ComM_00568](#)).](SRS_ModeMgm_09132)

Rationale for [SWS ComM_00667](#): Service `Nm_NetworkRequest()` is not available.

7.7 Bus error management

7.7.1 Network Start Indication

[SWS_ComM_00583] [The ComM module shall switch channel X to `COMM_FULL_COMMUNICATION` if NM indicates `ComM_Nm_NetworkStartIndication(<channel X>)` and `CommunicationAllowed` flag is set to TRUE.](SRS_ModeMgm_09132)

Use Case for [SWS ComM_00583](#): A node sends an NM message in "Prepare Bus Sleep" state but other nodes are already in "Bus Sleep" state because of "race conditions".

7.8 Test support requirements

7.8.1 Inhibited Full Communication Request Counter

[SWS_ComM_00138] [The ComM module shall provide one Inhibit counter for all rejected `COMM_FULL_COMMUNICATION` mode requests. It shall count user requests, which cannot be fulfilled because the system has inhibited communication modes.](SRS_ModeMgm_09155)

Rationale for [SWS ComM_00138](#): The counter is used for detecting latent software problems related to unmotivated communication bus wake ups.

[SWS_ComM_00140] [The Inhibit counter ([SWS ComM_00138](#)) for all rejected `COMM_FULL_COMMUNICATION` mode requests shall be stored in non-volatile memory.](SRS_ModeMgm_09155)

[SWS_ComM_00141] [The range of the Inhibit counter ([SWS ComM 00138](#)) for all rejected `COMM_FULL_COMMUNICATION` mode requests shall be 0 to 65535.](SRS_ModeMgm_09155)

[SWS_ComM_00142] [The Inhibit counter ([SWS ComM 00138](#)) for all rejected `COMM_FULL_COMMUNICATION` mode requests shall stop to increment if the maximum counter value is reached.](SRS_ModeMgm_09155)

[SWS_ComM_00143] [It shall be possible to read out and reset the Inhibit counter ([SWS ComM 00138](#)) for all rejected `COMM_FULL_COMMUNICATION` mode requests value by a ComM module API call.](SRS_ModeMgm_09156)

Use Case for [SWS ComM 00143](#): It shall be possible to read out and reset the current status of the counter by a diagnostic service.

7.9 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.

7.9.1 Development errors

[SWS_ComM_00234][

<i>Type of error</i>	<i>Related error code</i>	<i>Error value</i>
API service used without module initialization	COMM_E_UNINIT	0x1
API service used with wrong parameters	COMM_E_WRONG_PARAMETERS	0x2
API Service used with a null pointer	COMM_E_PARAM_POINTER	0x3
Initialization failed	COMM_E_INIT_FAILED	0x4

](SRS_BSW_00323, SRS_BSW_00327, SRS_BSW_00337, SRS_BSW_00385, SRS_BSW_00386)

[SWS_ComM_00612] [If ComM is not initialized, all ComM module and all API service other than `ComM_Init()` (see [SWS ComM 00146](#)), `ComM_GetVersionInfo()` (see [SWS COMM 00370](#)) and `ComM_GetStatus()` (see [SWS COMM 00242](#)); shall:

- not execute their normal operation,
- and return `E_NOT_OK`, if it has a standard return type.](SRS_BSW_00406)

[SWS_ComM_00858] [If development error detection is enabled by ComMDevErrorDetect (see [ECUC ComM_00555](#)): the function shall check that the service ComM_Init was previously called. If the check fails, the function shall raise the development error COMM_E_UNINIT](SRS_BSW_00406)

7.9.2 Runtime Errors

There are no runtime errors.

7.9.3 Transient Faults

There are no transient faults.

7.9.4 Production Errors

There are no production errors.

7.9.5 Extended Production Errors

There are no extended production errors.

7.10 Communication Manager Module Services

This section defines the AUTOSAR Interfaces of the Communication Manager Module Service (ComM).

7.10.1 Architecture

The overall architecture of the Communication Manager Module service is depicted in Figure 9:

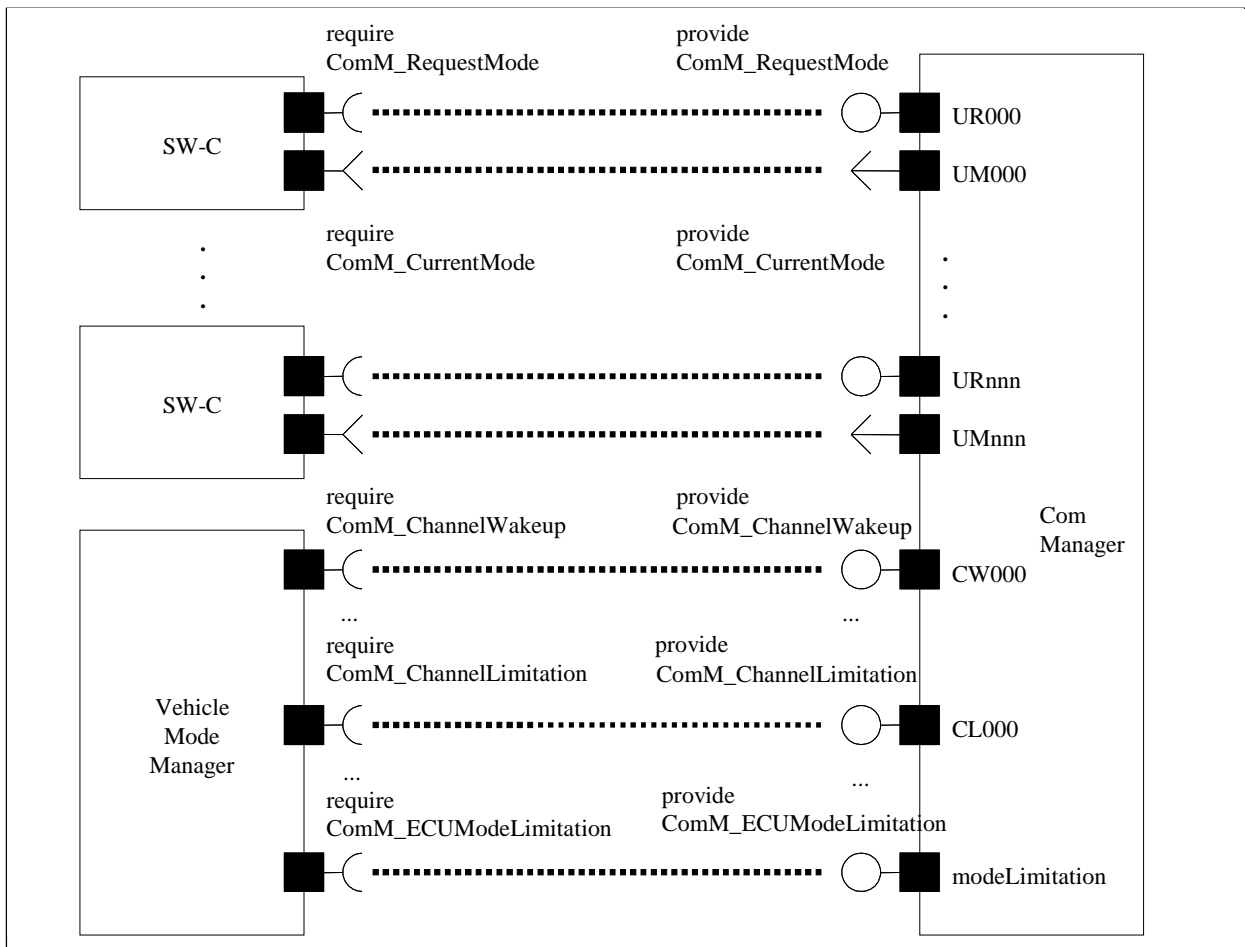


Figure 9: ARPackage of the Communication Manager Module

7.10.2 Use Cases

7.10.2.1 SW-Cs does not care about the ComM module at all

A SW-C that does not care about the Communication Manager Module will not require any of the interfaces defined in the ARPackage of the Communication Manager Module.

7.10.2.2 SW-Cs only cares about the state of its communication system

In this use case, a SW-C wants to know what communication capabilities it has (expressed by a communication mode 'none', 'silent' or 'full' - see `ComM_ModeType`). The SW-C finds out about that by defining a port requiring the Interface `ComM_GetCurrentComMode`. Depending on the available communication capabilities, the SW-C can specify that certain runnables of the SW-C should be executed or not. The Communication Manager Module must be configured correctly (with e.g. the physical channels that this SW-C uses for its logical communication) such that it has a port that provides this information about the current communication mode to the SW-C.

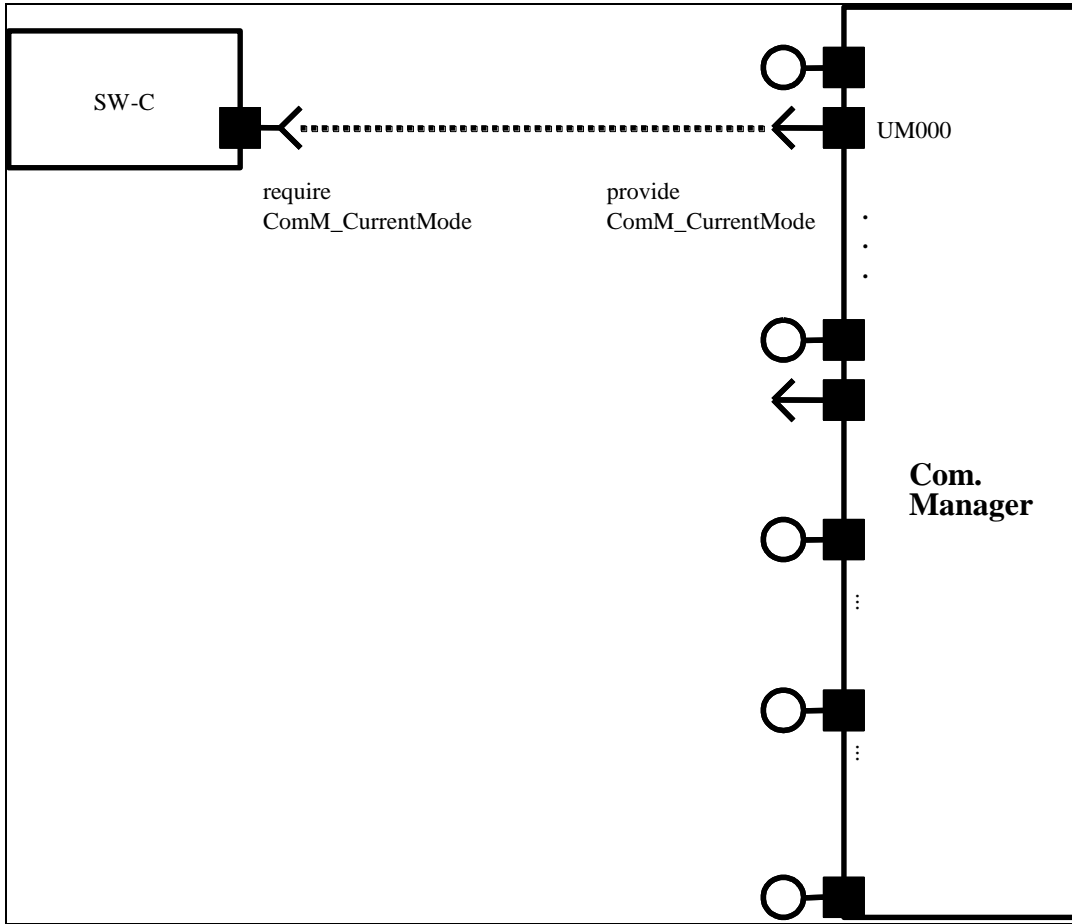


Figure 10: SW-C requests state changes to the Communication Manager Module

7.10.2.3 SW-Cs explicitly wants to take influence on its communication state

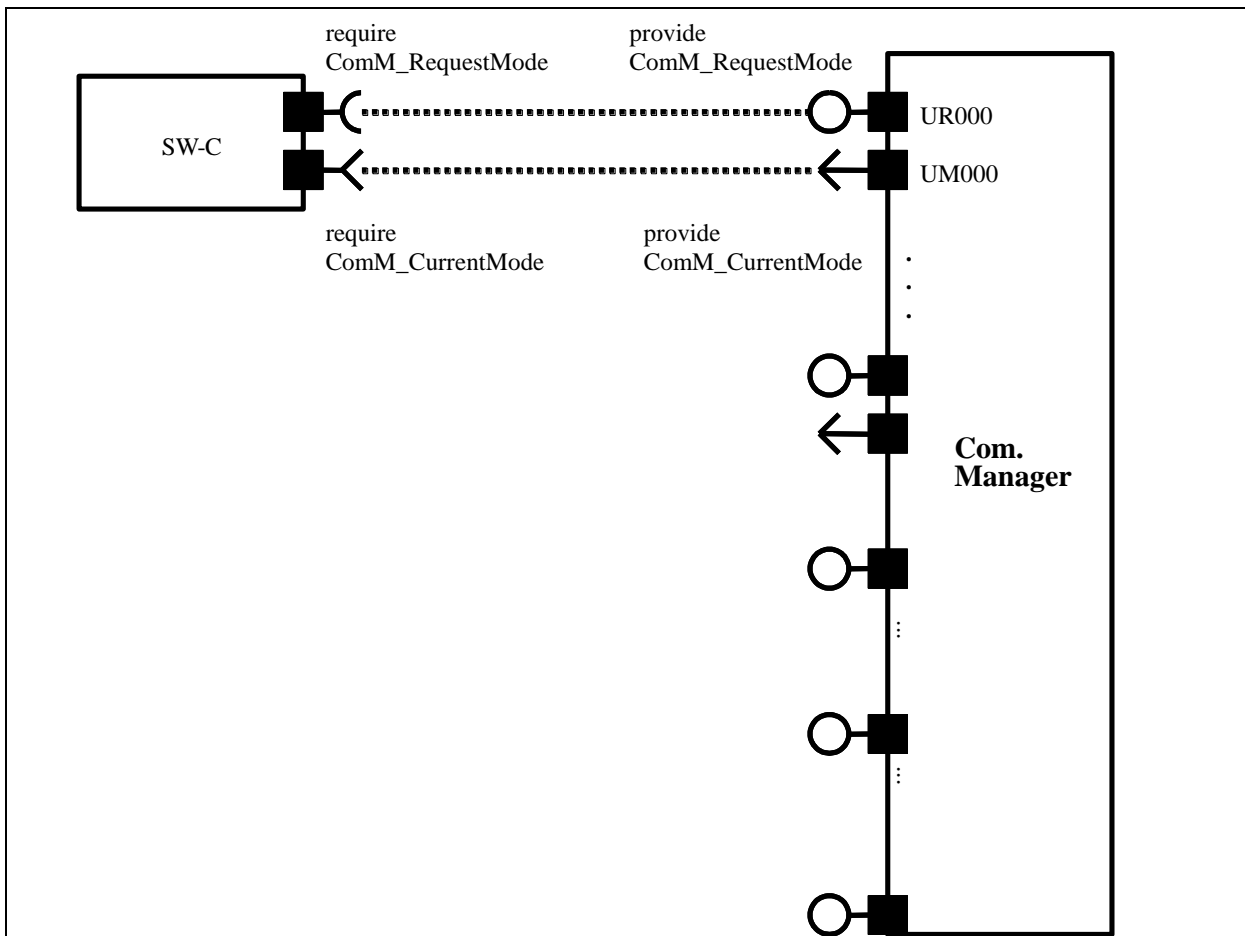


Figure 11: SW-C requires state changes within the Communication Manager Module and reads out current communication state

In this use case, the SW-C wants to explicitly take influence on the communication-state of the physical channels it needs. The SW-C indicates this by a specific port. Through this port, the SW-C can then request the Communication Manager Module mode “No Communication” or “Full Communication”. The Communication Manager Module will use these calls to request the corresponding communication mode from the corresponding Bus State Manager module.

[SWS_ComM_00848] The Communication Manager Module shall provide an AUTOSAR port to allow the request of an communication mode by calling ‘ComM_RequestComMode’ (see [SWS ComM 00110](#)).|(SRS_ModeMgm_09078)

For a SW-C using the “direct API” of the RTE, the SW-C could for example do the following:

```
MySW-C_Runnable_Init(self)
{
    // SW-C wants to send and receive data
    e = Rte_Call_comRequest_RequestComMode(COMM_FULL_COMMUNICATION);
```

```

if (e == RTE_E_OK)
{
    // successfully requested the Com Manager Module to move to
    // full communication mode
}
else
{
    // an error occurred when
    // interacting with the Com Manager module
    if (e == E_MODE_LIMITATION)
    {
        // a current ComMMode limitation forbids going into
        // that mode;
        // let's ask what the maximal allowed ComMMode is
        Rte_Call_comRequest_GetMaxComMode(&max);
        if (max==COMM_NO_COMMUNICATION)
        {
            ...
        };
    }
    else
    {
        // a more serious error occurred ...
    };
};
...
};

MySW-C_Runnable_Loop(self)
{
    if (status == ready_to_sleep)
    {
        //no need to send; ready for shutdown communication
        Rte_Call_comRequest_RequestComMode(COMM_NO_COMMUNICATION);
        ...
    };
};

```

Comment: Note that these APIs do not require that the SW-C has knowledge of the channels that it needs.

7.10.2.4 SW-C wants to interact directly with physical channels activate ECU Mode Limitation

The SW-C shall request mode from BswM. BswM will handle the direct communication with ComM.

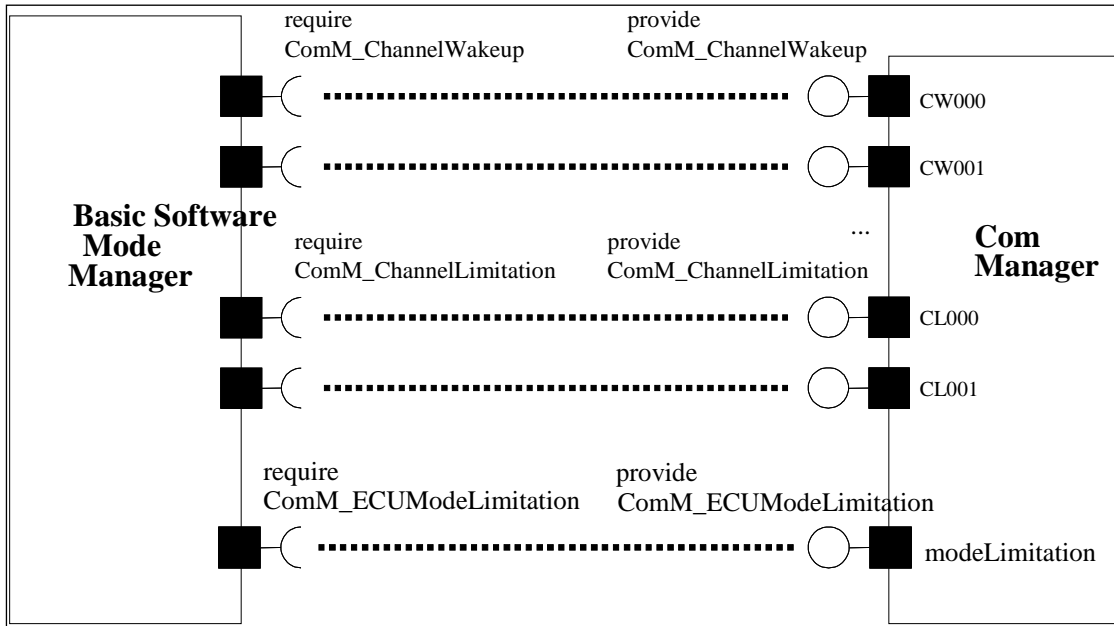


Figure 12: Interaction between BswM and the ComM module

7.10.3 Specification of Ports and Port Interfaces

This section specifies the Port Interfaces that are needed to operate the Communication Manager Module functionality over the RTE.

7.10.3.1 Types used by the interfaces

See 8.7.4 [Implementation Data Types](#)

7.10.3.2 Ports and Port Interface for User Requests

7.10.3.2.1 General Approach

A SW-C that wants to explicitly direct the local Communication Manager Module of the ECU towards a certain state requires the client-server interface `ComM_UserRequest`. Through this interface the SW-C can set the desired state of all communication channels that are relevant for that component, to “No Communication” or “Full Communication”. In order to keep the SW-Cs code independent from the values of the handles that are used to identify the user towards the Communication Manager Module, these handles are not passed from the SW-C to the Communication Manager Module. Rather they are modeled as “port defined argument values” of the Provide Ports on the Communication Manager Module’s side. As a consequence, these handles do not show up as arguments in the operations of the client-server interface `ComM_UserRequest`. As a further consequence of this approach, the Communication Manager Module has a separate port for each user.

7.10.3.2.2 Data Types

No data types are needed for this interface.

7.10.3.2.3 Port interface ComM_UserRequest

See 8.7.2.4 [ComM_UserRequest](#)

7.10.3.3 Ports and Port Interfaces for the current mode of the Communication Manager Module

7.10.3.3.1 General approach

[SWS_ComM_00847] [The Communication Manager Module shall have an AUTOSAR port providing the ModeSwitchInterface interface 'ComM_CurrentMode'.](SRS_ModeMgm_09085)

[SWS_ComM_00733] [The Communication Manager Module shall have a separate port providing the ModeSwitchInterface interface 'ComM_CurrentMode' for each configured user, to which a SW-C is connected.](SRS_ModeMgm_09085)

A SW-C that wants to get informed about its current Communication Manager Module Mode requires the ModeSwitchInterface interface ComM_CurrentMode.

7.10.3.3.2 Port interface ComM_CurrentMode

See 8.7.3.1 [ComM_CurrentMode](#).

7.10.3.4 Ports and Port Interfaces for the ComM users currently requesting COMM_FULL_COMMUNICATION

7.10.3.4.1 General approach

[SWS_ComM_00734] [The Communication Manager Module shall have an optional (see [ECUC_ComM_00787](#)) separate port providing the sender-receiver interface 'ComM_CurrentChannelRequest' for each configured ComM channel.](SRS_ModeMgm_09084)

Rationale for SWS_ComM_00734: A SW-C that wants to get informed about, which users are currently requesting COMM_FULL_COMMUNICATION requires the sender-receiver interface ComM_CurrentChannelRequest'.

[SWS_ComM_00736] [Whenever the set of ComM users currently requesting COMM_FULL_COMMUNICATION for a channel changes, the Communication Manager Module shall update the data element fullComRequestors. A change shall update the data element only, when the Communication Manager Module accepts the communication request of the ComM user.](SRS_ModeMgm_09078)

Note: Requests which are accepted but not processed because of active ModeLimitations will lead to an update of the data element.

7.10.3.4.2 Data Types

See 8.7.4.4 [ComM_UserHandleArrayType](#).

7.10.3.4.3 Port Interface ComM_CurrentChannelRequest

See 8.7.1.1 [ComM_CurrentChannelRequest](#).

7.10.3.5 Ports and Port Interface for ECU Mode Limitation

7.10.3.5.1 General approach

[SWS_ComM_00740] [The Communication Manager Module can be configured to have an AUTOSAR port providing the client-server interface ComM_ECUModeLimitation.](SRS_ModeMgm_09071)

A SW-C, which plays the role of a “Mode Manager”, can use this interface to change the behaviour of the entire ECU.

7.10.3.5.2 Port interface ComM_ECUModeLimitation

See 8.7.2.3 ComM_ECUModeLimitation.

7.10.3.6 Ports and Port Interface for Channel Wake up

7.10.3.6.1 General approach

[SWS_ComM_00747] [The Communication Manager Module can be configured to have an AUTOSAR port providing the Client-Server Interface ComM_ChannelWakeup.](SRS_ModeMgm_09089)

A SW-C playing the role of a “Mode Manager” can use this interface to configure the Communication Manager Module to take precautions against awaking other ECU's by starting the communication. In order to keep the SW-Cs code independent from the values of the handles that are used to identify a specific handle towards the Communication Manager Module, these handles are **not** passed from the SW-C to the Communication Manager Module. Rather they are modeled as “port defined argument values” of the Provide Ports on the Communication Manager Module's side. As a consequence, these handles do not show up as arguments in the operations of the client-server interface ComM_ChannelWakeup. As a further consequence of this approach, the Communication Manager Module has separate ports for each channel.

7.10.3.6.2 Port interface ComM_ChannelWakeup

See 8.7.2.2 ComM_ChannelWakeup.

7.10.3.7 Ports and Port Interface for interface Channel Limitation

7.10.3.7.1 General approach

[SWS_ComM_00752] [The Communication Manager Module can be configured to have an AUTOSAR port providing the Client-Server Interface ComM_ChannelLimitation.](SRS_ModeMgm_09071)

A SW-C playing the role of a “Mode Manager” can use this interface to configure the Communication Manager Module to inhibit communication mode for a given channel. In order to keep the SW-Cs code independent from the values of the handles that are used to identify a specific handle towards the Communication Manager Module, these handles are **not** passed from the SW-C to the Communication Manager Module. Rather they are modelled as “port defined argument values” of the Provide Ports on the Communication Manager Module side. As a consequence, these handles do not show up as arguments in the operations of the client-server interface ComM_ChannelLimitation. As a further consequence of this approach, the Communication Manager Module has separate ports for each channel.

7.10.3.7.2 Port interface ComM_ChannelLimitation

See 8.7.2.1 ComM_ChannelLimitation.

7.10.3.8 Definition of the Service of the Communication Manager Module

This section provides guidance on the definition of the Communication Manager Module service. There are ports on both sides of the RTE. This description of the Communication Manager Module service defines the ports below the RTE. Each SW-C, which uses the Service, must contain “service ports” in its own SW-C description which will be connected to the ports of the COM Manager module, so that the RTE can be generated.

Comment: Note that these definitions can only be completed during ECU configuration (because it depends on certain configuration parameters of the Communication Manager Module, which determine the number of ports provided by the Communication Manager Module service). Also note that the implementation of an SW-C does *not* depend on these definitions.

[SWS_ComM_00744]

```
[
/* This is the definition of the Communication Manager Module as a service.
This is the 'outside-view' of the Communication Manager Module */
Service ComM
{
```

```

// port present if ComMModeLimitationEnabled (see ECUC ComM 00560)
ProvidePort ComM_ECUModeLimitation modeLimitation;

// port present for each channel
// if ComMModeLimitationEnabled (see ECUC ComM 00560);
// there are NC channels;
ProvidePort ComM_ChannelLimitation CL000;
...
ProvidePort ComM_ChannelLimitation CL<NC-1>;

// port present for each channel
// if COMM_WAKEUP_INHIBITION_ENABLED (see ECUC ComM 00559)
ProvidePort ComM_ChannelWakeup CW000;
...
ProvidePort ComM_ChannelWakeup CW<NC-1>;

// For each user the Communication Manager Module provides 2 ports.
// To facilitate configuration, the index of this user shall
// correspond to the index in the array COMM_USER_LIST used for the
// configuration of the Communication Manager Module.
// The number of users must correspond to the size of this array.
ProvidePort ComM_UserRequest UR000; // (see 7.10.3.2.2)
ProvidePort ComM_CurrentMode UM000;
ProvidePort ComM_UserRequest UR001; //(see 7.10.3.2.2)
ProvidePort ComM_CurrentMode UM001;
...
ProvidePort ComM_UserRequest UR<COMM_USER_LIST.size-1>;
ProvidePort ComM_CurrentMode UM<COMM_USER_LIST.size-1>;

// port present for each channel if configured
// (see ECUC_ComM_00787)
// there are NC channels;
ProvidePort ComM_CurrentChannelRequest CR000;
...
ProvidePort ComM_CurrentChannelRequest CR<NC-1>;

};|(SRS_ModeMgm_09078,          SRS_ModeMgm_09080,          SRS_ModeMgm_09084,
SRS_ModeMgm_09172, SRS_ModeMgm_09149, SRS_ModeMgm_09168, SRS_ModeMgm_09071,
SRS_ModeMgm_09157)

```

7.10.4 Runnables and Entry points

7.10.4.1 Internal behaviour

This is the inside description of the Communication Manager Module. This detailed description is only needed for the configuration of the local RTE.

[SWS_ComM_00745]

```

|
InternalBehavior of the Communication Manager Module
{
    // Runnable entities of the Communication Manager Module
    RunnableEntity LimitECUToNoComMode
        symbol "ComM_LimitECUToNoComMode" /* see SWS ComM 00124 */
        canbeInvokedConcurrently = FALSE

    RunnableEntity ReadInhibitCounter
        symbol "ComM_ReadInhibitCounter" /* see SWS ComM 00224 */
        canbeInvokedConcurrently = FALSE

```

```

RunnableEntity ResetInhibitCounter
    symbol "ComM_ResetInhibitCounter" /* see SWS ComM 00108 */
    canbeInvokedConcurrently = FALSE

RunnableEntity SetECUGroupClassification
    symbol "ComM_SetECUGroupClassification" /* see SWS ComM 00552 */
    canbeInvokedConcurrently = FALSE

RunnableEntity LimitChannelToNoComMode
    symbol "ComM_LimitChannelToNoComMode" /* see SWS ComM 00163 */
    canbeInvokedConcurrently = FALSE

RunnableEntity GetInhibitionStatus
    symbol "ComM_GetInhibitionStatus" /*see SWS ComM 00619 */
    canbeInvokedConcurrently = FALSE

RunnableEntity PreventWakeup
    symbol "ComM_PreventWakeup"
    canbeInvokedConcurrently = FALSE

RunnableEntity RequestComMode
    symbol "ComM_RequestComMode" /* see SWS ComM 00110 */
    canbeInvokedConcurrently = TRUE

RunnableEntity GetMaxComMode
    symbol "ComM_GetMaxComMode" /* see SWS ComM 00085 */
    canbeInvokedConcurrently = TRUE

RunnableEntity GetRequestedComMode
    symbol "ComM_GetRequestedComMode"
    canbeInvokedConcurrently = TRUE

RunnableEntity GetCurrentComMode
    symbol "ComM_GetCurrentComMode" /*see SWS ComM 00083 */
    canbeInvokedConcurrently = TRUE

// the following applies if ComMModeLimitationEnabled
// (see ECUC ComM 00560)
modeLimitation.LimitECUToNoComMode -> LimitECUToNoComMode
modeLimitation.ReadInhibitCounter -> ReadInhibitCounter
modeLimitation.ResetInhibitCounter -> ResetInhibitCounter
modeLimitation.SetECUGroupClassification -> SetECUGroupClassification

// per-channel behaviour only present
// if ComMModeLimitationEnabled (see ECUC ComM 00560)
// there are NC channels
// To facilitate configuration, the names of the channels correspond
// to the index of the channel in the "Channel" container used to
// configure the Communication Manager Module
CL000.LimitChannelToNoComMode -> LimitChannelToNoComMode
CL000.GetInhibitionStatus -> GetInhibitionStatus
PortArgument {port=CL000,
    value.type=NetworkHandleType,
    value.value=Channel[0].COMM_CHANNEL_ID}
...
CLnnn.LimitChannelToNoComMode -> LimitChannelToNoComMode
CLnnn.GetInhibitionStatus -> GetInhibitionStatus
PortArgument {port=CLnnn,
    value.type=NetworkHandleType,
    value.value=Channel[nnn].COMM_CHANNEL_ID}

```

```

// per-channel behaviour only present
// if COMM_WAKEUP_INHIBITION_ENABLED (see ECUC ComM\_00559)
CW000.preventWakeUp -> PreventWakeUp
PortArgument {port=CW000,
               value.type=NetworkHandleType,
               value.value=Channel[0].COMM_CHANNEL_ID}
...
CWnnn.preventWakeUp -> PreventWakeUp
PortArgument {port=CWnnn,
               value.type=NetworkHandleType,
               value.value=Channel[nnn].COMM_CHANNEL_ID}

// per-user behaviour
// Note that the port-argument value must be consistent with the
// value in the configuration COMM_USER_LIST
// Note that the exact data-type of the UserHandleType must of course
// be defined BEFORE RTE_configuration, but does NOT affect the
// API seen by the SW-Cs that use the service
UR000.RequestComMode -> RequestComMode
UR000.GetMaxComMode -> GetMaxComMode
UR000.GetRequestedComMode -> GetRequestedComMode
UR000.GetCurrentComMode -> GetCurrentComMode
PortArgument {port=UR000,
               value.type= ComM_UserhandleType,
               value.value=COMM_USER_LIST[0]}
...
URnnn.RequestComMode -> RequestComMode
URnnn.GetMaxComMode -> GetMaxComMode
URnnn.GetRequestedComMode -> GetRequestedComMode
URnnn.GetCurrentComMode -> GetCurrentComMode
PortArgument {port=URnnn,
               value.type= ComM_UserhandleType,
               value.value=COMM_USER_LIST[n]}
};|(      SRS_ModeMgm_09078,      SRS_ModeMgm_09080,      SRS_ModeMgm_09084,
SRS_ModeMgm_09172, SRS_ModeMgm_09149, SRS_ModeMgm_09168, SRS_ModeMgm_09071,
SRS_ModeMgm_09157)
    
```

Comment:

'modeLimitation.LimitECUToNoComMode -> LimitECUToNoComMode' is supposed to define an OperationInvokedEvent that links the OperationPrototype to the runnable entity that is supposed to be executed.

7.10.4.2 Header file to be included by the Communication Manager Module

The RTE deals with the Communication Manager Module as with any normal SW-C. The RTE will be able to generate a header-file based on the internal-behaviour description of the Communication Manager Module which contains for instance a definition of the API's (like `Rte_Ports_CurrentMode_P`) which are available to the Communication Manager Module. This implies that an implementation of the Communication Manager Module must include this generated header-file.

7.11 Multicore Distribution

In its role as central module dealing with different network types the ComM interaction spans across partitions in case the Com-Stack is distributed and so shall provide required multi-core features to ensure a clean architecture and keep the network dependent clusters free of multi-partition (multi-core) add-ons.

[SWS_ComM_01019] The ComM module shall apply appropriate mechanisms to allow calls of its APIs from other partitions than its main function, e.g. by providing a ComM satellite.](SRS_BSW_00459)

[SWS_ComM_01020] ComM shall interact with <Bus>SM (i.e. call <Bus>SM APIs) only in the partition, where the respective <Bus>SM module is assigned to.](SRS_BSW_00459)

[SWS_ComM_01059] ComM shall interact with Dcm (i.e. call Dcm APIs) only in the partition, where the Dcm module is assigned to.](SRS_BSW_00459)

[SWS_ComM_01021] {OBSOLETE} The ComM shall call signal related Com APIs (Com_SendSignal / Com_ReceiveSignal) only in the partition, where the respective ComSignal is handled by the Com module.](SRS_BSW_00459)

Hint: The partition assignment information of the ComSignals is available within Com module configuration.

Note: Even though the basic software (and the Com-Stack in particular) is distributed across several partitions, ComM and Nm Masters should reside in the same partition in order to keep mode interfaces between the two modules simple (for further information see chapter Master/Satellite-approach in [32] (Guide to BSW Distribution)).

7.12 Non functional requirements

[SWS_ComM_00459] It shall be possible to integrate the ComM module delivered as source or object code into the AUTOSAR stack.

Rationale:

- Allow IP protection and guaranteed test coverage: object code
- Allow high efficiency and configurability at system generation time (by integrator): source code.](SRS_BSW_00342)

8 API specification

8.1 Imported types

8.1.1 Standard types

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

[SWS_ComM_00820]

Module	Header File	Imported Type
Com	Com.h	Com_SignalIdType
ComStack_Types	ComStack_Types.h	NetworkHandleType
	ComStack_Types.h	PNCHandleType
NvM	Rte_NvM_Type.h	NvM_BlockIdType
	Rte_NvM_Type.h	NvM_RequestResultType
Std	Std_Types.h	Std_ReturnType
	Std_Types.h	Std_VersionInfoType

](SRS_BSW_00348, SRS_BSW_00357)

Note: Types from Com module are considered obsolete in R21-11 due to a concept part, which will be validated after the release of R21-11.

The ComM API uses the following extension to Std_ReturnType:

[SWS_ComM_91027]{DRAFT} [

Range	COMM_E_MODE_LIMITATION	2	Function call has been successful but mode can not be granted because of mode inhibition.
	COMM_E_MULTIPLE_PNC_ASSIGNED	3	Function could not provide the current mode of the PNC, since multiple PNCs are assigned to the affected user
	COMM_E_NO_PNC_ASSIGNED	4	Function could not provide the current mode of the PNC, since no PNC is assigned to the affected user
	COMM_E_LEARNING_ACTIVE	5	Function call has been successfully, but functionality cannot be executed because PNC learning phase is active.
Description	-- Tags:atp.Status=draft		
Available via	ComM.h		

](SRS_BSW_00331, SRS_BSW_00369, SRS_BSW_00377, SRS_BSW_00441)

8.2 Type definitions

[SWS_ComM_00863] [The following Data Types shall be used for the functions defined in this Specification.](SRS_BSW_00441)

8.2.1 ComM_InitStatusType

[SWS_ComM_00668][

Name	ComM_InitStatusType		
Kind	Enumeration		
Range	COMM_UNINIT	0x00	The COM Manager is not initialized or not usable. This shall be the default value after reset. This status shall have the value 0.
	COMM_INIT	0x01	The COM Manager is initialized and usable.
Description	Initialization status of ComM.		
Available via	ComM.h		

]()

8.2.2 ComM_PncModeType

[SWS_ComM_00673][

Name	ComM_PncModeType		
Kind	Enumeration		
Range	COMM_PNC_REQUESTED	0x00	PNC is requested by a local ComM user
	COMM_PNC_READY_SLEEP	0x01	PNC is requested by a remote ComM user
	COMM_PNC_PREPARE_SLEEP	0x02	PNC is active with no deadline monitoring
	COMM_PNC_NO_COMMUNICATION	0x03	PNC does not communicate
	COMM_PNC_REQUESTED_WITH_WAKEUP_REQUEST	0x04	PNC is requested by a local ComM user. The mode is used to indicate the BswM, that an active PNC request should trigger also a wake-up of the used communication hardware, if this is supported and configured (e.g. used for Ethernet switch port

			switching in combination with OA TC10 compliant Ethernet hardware).
Description	Current mode of a PNC		
Available via	ComM.h		

]()

8.2.3 ComM_StateType

[SWS_ComM_00674]

Name	ComM_StateType		
Kind	Type		
Derived from	uint8		
Range	COMM_NO_COM_NO_PENDING_REQUEST	0	--
	COMM_NO_COM_REQUEST_PENDING	1	--
	COMM_FULL_COM_NETWORK_REQUESTED	2	--
	COMM_FULL_COM_READY_SLEEP	3	--
	COMM_SILENT_COM	4	--
Description	State and sub-state of ComM state machine ComM states vs. Communication Modes: COMM_NO_COM* : Communication Mode='No Communication' COMM_FULL_COM* : Communication Mode='Full Communication' COMM_SILENT_COM: Communicatio Mode='Silent Communication'		
Available via	ComM.h		

]()

8.2.4 ComM_ConfigType

[SWS_ComM_00162]

Name	ComM_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	ComM.h

]()

8.3 Function definitions

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

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

- ComM_Init
- ComM_GetVersionInfo

8.3.1 ComM_Init

[SWS_ComM_00146]

Service Name	ComM_Init	
Syntax	<pre>void ComM_Init (const ComM_ConfigType* ConfigPtr)</pre>	
Service ID [hex]	0x01	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	ConfigPtr	Pointer to post-build configuration data
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Initializes the AUTOSAR Communication Manager and restarts the internal state machines.	
Available via	ComM.h	

](SRS_BSW_00101, SRS_BSW_00358, SRS_BSW_00414)

[SWS_ComM_00793] [Caveats of ComM_Init(): The NVRAM Manager module has to be initialized to have the possibility to "direct" access the ComM module's parameters.]()

[SWS_ComM_00864] [In `ComM_Init()` ComM shall read non-volatile parameters specified in [SWS_ComM_00103](#) from NVRAM. If no parameters are available, ComM shall use the default values in the ComM configuration.]()

8.3.2 ComM_DeInit

[SWS_ComM_00147]

Service Name	ComM_DeInit
Syntax	<pre>void ComM_DeInit (void)</pre>
Service ID [hex]	0x02
Sync/Async	Synchronous
Reentrancy	Non Reentrant
Parameters (in)	None
Parameters (inout)	None
Parameters (out)	None
Return value	None
Description	This API de-initializes the AUTOSAR Communication Manager.
Available via	ComM.h

](SRS_BSW_00336)

[SWS_ComM_00794] [De-initialization in `ComM_DeInit()` shall only be performed if all channels controlled by the ComM module are in `COMM_NO_COMMUNICATION` mode.]()

Rationale for [SWS_ComM_00794](#): Since the `ComM_DeInit()` API cannot return an error message, it must be assured that all channels are in `COMM_NO_COMMUNICATION` mode and `COMM_NO_COM_NO_PENDING_REQUEST` sub-state before `ComM_DeInit()` is called.

[SWS_ComM_00865] [In `ComM_DeInit` ComM shall store non-volatile parameters specified in [SWS_ComM_00103](#) to NVRAM.]()

8.3.3 ComM_GetStatus

[SWS_ComM_00242]

Service Name	ComM_GetStatus
---------------------	----------------

Syntax	Std_ReturnType ComM_GetStatus (ComM_InitStatusType* Status)	
Service ID [hex]	0x03	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	Status	COMM_UNINIT: The ComM is not initialized or not usable. Default value after startup or after ComM_Delnit() is called. COMM_INIT: The ComM is initialized and usable.
Return value	Std_ReturnType	E_OK: Successfully return of initialization status E_NOT_OK: Return of initialization status failed
Description	Returns the initialization status of the AUTOSAR Communication Manager. After a call to ComM_Delnit() ComM should have status COMM_UNINIT, and a new call to ComM_Init needed to make sure ComM restart internal state machines to default values.	
Available via	ComM.h	

](SRS_BSW_00406)

8.3.4 ComM_GetInhibitionStatus

[SWS_ComM_00619]

Service Name	ComM_GetInhibitionStatus	
Syntax	Std_ReturnType ComM_GetInhibitionStatus (NetworkHandleType Channel, ComM_InhibitionStatusType* Status)	
Service ID [hex]	0x04	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Channel	See NetworkHandleType
Parameters (inout)	None	
Parameters (out)	Status	See ComM_InhibitionStatusType
Return value	Std_ReturnType	E_OK: Successfully returned Inhibition Status E_NOT_OK: Return of Inhibition Status failed

Description	Returns the inhibition status of a ComM channel.
Available via	ComM.h

]()

8.3.5 ComM_RequestComMode

[SWS_ComM_00110]

Service Name	ComM_RequestComMode	
Syntax	Std_ReturnType ComM_RequestComMode (ComM_UserHandleType User, ComM_ModeType ComMode)	
Service ID [hex]	0x05	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	User	Handle of the user who requests a mode
	ComMode	COMM_FULL_COMMUNICATION COMM_NO_COMMUNICATION
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Successfully changed to the new mode E_NOT_OK: Changing to the new mode failed COMM_E_MODE_LIMITATION: Mode can not be granted because of mode inhibition.
Description	<p>Requesting of a Communication Mode by a user.</p> <p>Note: The following modes are no valid user requests, since they are used as internal modes:</p> <ul style="list-style-type: none"> - COMM_SILENT_COMMUNICATION (this mode is used for synchronization at shutdown) - COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST (this mode is used internally within the ComM channel statemachine to trigger the lower layers to request a wakeup on the network if the used hardware support such a feature. (e.g. Ethernet hardware which is compatible with OA TC10). <p>The following modes are valid user requests:</p> <ul style="list-style-type: none"> - COMM_NO_COMMUNICATION - COMM_FULL_COMMUNICATION. The communication request could also be released due to a ComM communication inhibition 	
Available via	ComM.h	

] (SRS_ModeMgm_09081)

[SWS_ComM_00795] [Configuration of ComM_RequestComMode: Relationship between users and channels. A user is statically mapped to one or more channels.](SRS_ModeMgm_09090)

8.3.6 ComM_GetMaxComMode

[SWS_ComM_00085]

Service Name	ComM_GetMaxComMode	
Syntax	Std_ReturnType ComM_GetMaxComMode (ComM_UserHandleType User, ComM_ModeType* ComMode)	
Service ID [hex]	0x06	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	User	Handle of the user who requests a mode
Parameters (inout)	None	
Parameters (out)	ComMode	See ComM_ModeType
Return value	Std_Return- Type	E_OK: Successfully returned maximum allowed Communication Mode E_NOT_OK: Return of maximum allowed Communication Mode failed
Description	Function to query the maximum allowed Communication Mode of the corresponding user.	
Available via	ComM.h	

l)

Use Case: This function provides the possibility to request the maximum possible mode (e.g. user wants to check if it is possible to get "Full Communication" mode or if a limitation/inhibition is active). This is needed for diagnosis/debugging..

[SWS_ComM_00374] [If more than one channel is linked to one user request and the maximum allowed modes of the channels are different, then the function ComM_GetMaxComMode shall return the lowest mode (see [SWS ComM_00867](#) and [SWS ComM_00868](#)).](SRS_ModeMgm_09149)

[SWS_ComM_00796] [Configuration of ComM_GetMaxComMode: Relationship between users and channels. A user is statically mapped to one or more channels.](SRS_ModeMgm_09090)

8.3.7 ComM_GetRequestedComMode

[SWS_ComM_00079]

Service Name	ComM_GetRequestedComMode	
Syntax	<pre>Std_ReturnType ComM_GetRequestedComMode (ComM_UserHandleType User, ComM_ModeType* ComMode)</pre>	
Service ID [hex]	0x07	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	User	Handle of the user who requests a mode
Parameters (inout)	None	
Parameters (out)	ComMode	Name of the requested mode
Return value	Std_ReturnType	E_OK: Successfully returned requested Communication Mode E_NOT_OK: Return of requested Communication Mode failed
Description	Function to query the currently requested Communication Mode of the corresponding user.	
Available via	ComM.h	

](SRS_ModeMgm_09149)

Rationale for [SWS ComM 00079](#): The requested user "Communication Mode" has to be stored volatile within the Communication Manager Module itself, to prevent redundant storage of status information by the users.

Comment: If the Communication Manager Module would not have this service every user has to store the status on its own --> redundant and possibly inconsistent storage of the same data.

Note: A user is statically mapped to one or more channels. The relationship between users and channels is reflected by the configuration (see [ECUC ComM 00658](#)).

8.3.8 ComM_GetCurrentComMode

[SWS_ComM_00083]

Service Name	ComM_GetCurrentComMode	
Syntax	<pre>Std_ReturnType ComM_GetCurrentComMode (ComM_UserHandleType User, ComM_ModeType* ComMode)</pre>	

Service ID [hex]	0x08	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	User	Handle of the user who requests a mode
Parameters (inout)	None	
Parameters (out)	ComMode	See ComM_ModeType
Return value	Std_ReturnType	E_OK: Successfully returned Communication Mode from Bus State Manager E_NOT_OK: Return of Communication Mode from Bus State Manager failed
Description	Function to query the current Communication Mode. ComM shall use the corresponding interfaces of the Bus State Managers to get the current Communication Mode of the network. (Call to Bus State Manager API: XXXSM_GetCurrentComMode(...))	
Available via	ComM.h	

](SRS_ModeMgm_09084)

[SWS_ComM_00176] [If more than one channel is linked to one user request and the modes of the channels are different, the function `ComM_GetCurrentComMode` shall return the lowest mode (see [SWS_ComM_00867](#) and [SWS_ComM_00868](#)).](SRS_ModeMgm_09172)

[SWS_ComM_00798] [Configuration of `ComM_GetCurrentComMode`: Relationship between users and channels. A user is statically mapped to one or more channels.](SRS_ModeMgm_09090)

8.3.9 ComM_GetCurrentPNCComMode

[SWS_ComM_91002]

Service Name	ComM_GetCurrentPNCComMode	
Syntax	<pre>Std_ReturnType ComM_GetCurrentPNCComMode (ComM_UserHandleType User, ComM_ModeType* ComMode)</pre>	
Service ID [hex]	0x6a	
Sync/Async	Synchronous	
Reentrancy	Reentrant	

Parameters (in)	User	Handle of the user who requests a mode
Parameters (inout)	None	
Parameters (out)	ComMode	See ComM_ModeType
Return value	Std_Return-Type	E_OK: Successfully returned Communication Mode from Bus State Manager E_NOT_OK: Return of Communication Mode from Bus State Manager failed COMM_E_MULTIPLE_PNC_ASSIGNED: Function could not provide the current mode of the PNC, since multiple PNCs are assigned to the affected user COMM_E_NO_PNC_ASSIGNED: Function could not provide the current mode of the PNC, since no PNC is assigned to the affected user
Description	The function returns the current Communication Mode of the corresponding PNC the affected user is assigned to.	
Available via	ComM.h	

]()

[SWS_ComM_01022] [If more than one PNC is assigned to the affected user, the function `ComM_GetCurrentPNCComMode` shall return `COMM_E_MULTIPLE_PNC_ASSIGNED` as `ComMode`.](SRS_ModeMgm_09149)

Comment to [SWS_ComM_01022]: For multiple PNCs it is not possible to return a consistent communication mode since the PNCs could have different communication modes.

[SWS_ComM_01023] [If no PNC is assigned to the affected user, the function `ComM_GetCurrentPNCComMode` shall return `COMM_E_NO_PNC_ASSIGNED` as `ComMode`.](SRS_ModeMgm_09149)

[SWS_ComM_01024] [If **[SWS_ComM_01022]** and **[SWS_ComM_01023]** do not apply, the function shall query for the current communication mode of the corresponding PNC statemachine the user is assigned to. If the corresponding PNC statemachine is in main state `COMM_PNC_FULL_COMMUNICATION`, then the function shall return `COMM_FULL_COMMUNICATION` as `ComMode`. If the corresponding PNC statemachine is main state `COMM_PNC_NO_COMMUNICATION`, then the function shall return `COMM_NO_COMMUNICATION` as `ComMode`.](SRS_ModeMgm_09149)

Note: The service interface `ComM_UserRequest` provides the possibility among others to query for the current mode of a channel and to query for the current mode of a PNC. Since the service interface has `ComM_ModeType` as a return value type, the main state of the `ComM` PNC statemachine has to be mapped to the main state of the `ComM` channel statemachine

[SWS_ComM_01025] [Configuration of ComM_GetCurrentPNCComMode: Relationship between users and PNCs. A user is statically mapped to one or more PNCs.](SRS_ModeMgm_09090, SRS_ModeMgm_09246)

8.3.10 ComM_GetPncToChannelMapping

[SWS_ComM_91013]{DRAFT} [

Service Name	ComM_GetPncToChannelMapping (draft)	
Syntax	Std_ReturnType ComM_GetPncToChannelMapping (boolean* MappingTable, uint8* ChannelCnt, uint8* PncCnt)	
Service ID [hex]	0x68	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	Mapping Table	Pointer to two-dimensional array with the current Pnc-to-channel-mapping of the PNC Gateway where the first dimension covers all relevant channels and the second all relevant PNCs.
	Channel Cnt	Pointer to number of ComM channels that are passed in the Mapping Table parameter.
	PncCnt	Pointer to number of PNCs, that are passed in the MappingTable parameter.
Return value	Std_Return-Type	E_OK: Successfully get PNC-to-channel-mapping entry E_NOT_OK: Getting of PNC-to-channel-mapping entry failed COMM_E_LEARNING_ACTIVE: Functionality cannot be executed because PNC learning phase is active.
Description	This function returns the current configuration of the ECUs PNC-to-channel-mapping. Tags: atp.Status=draft	
Available via	ComM.h	

](SRS_ModeMgm_09259)

[SWS_ComM_01034] {DRAFT} [Function ComM_GetPncMappingTable shall be only available if ComMPncGatewayEnabled and

ComMDynamicPncToChannelMappingSupport are set to TRUE.](SRS_ModeMgm_09258)

[SWS_ComM_01035] {DRAFT} [If
ComMDynamicPncToChannelMappingEnabled is set to TRUE on at least one channel and when PNC learning phase is active, then the function ComM_GetPncMappingTable shall return with COMM_E_LEARNING_ACTIVE.](SRS_ModeMgm_09259)

[SWS_ComM_01036] {DRAFT} [If
ComMDynamicPncToChannelMappingEnabled is set to TRUE on at least one channel and when PNC learning phase is not active, then the function ComM_GetPncMappingTable shall provide within MappingTable the current PNC-to-channel mapping as a two-dimensional array where on first dimension all ComM channels where ComMPncGatewayType is set are handled according to their derived order in ComM and on second dimension all configured ComMPnc according to their order given by their ComMPncId. ComM shall also set the parameter ChannelCnt and PncCnt accordingly and return with E_OK.](SRS_ModeMgm_09259)

Note: The content of this MappingTable can only be interpreted correctly by application or tester correctly if the number of Channels and PNCs and their order is known.

8.3.11 ComM_UpdatePncToChannelMapping

[SWS_ComM_91015]{DRAFT} [

Service Name	ComM_UpdatePncToChannelMapping (draft)	
Syntax	<pre>Std_ReturnType ComM_UpdatePncToChannelMapping (const boolean* MappingTable, uint8 channelCnt, uint8 PncCnt)</pre>	
Service ID [hex]	0x62	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Mapping Table	Pointer to two-dimensional array with the current Pnc-to-channel-mapping of the PNC Gateway where the first dimension covers all relevant channels and the second all relevant PNCs.
	channel Cnt	Number of physical channels passed in the MappingTable
	PncCnt	Number of PNCs passed in the MappingTable
Parameters (inout)	None	

Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Successfully set PNC-to-channel-mapping entry E_NOT_OK: Set of PNC-to-channel-mapping entry failed COMM_E_LEARNING_ACTIVE: Functionality cannot be executed because PNC learning phase is active.
Description	This function can be used to set entries within the the ECUs PNC-to-channel-mapping Tags: atp.Status=draft	
Available via	ComM.h	

](SRS_ModeMgm_09259)

[SWS_ComM_01037] {DRAFT} [Function `ComM_UpdatePncMappingTable` shall be only available if `ComMPncGatewayEnabled` and `ComMDynamicPncToChannelMappingSupport` are set to TRUE.](SRS_ModeMgm_09258)

[SWS_ComM_01038] {DRAFT} [If `ComMDynamicPncToChannelMappingEnabled` is set to TRUE on at least one channel and the function `ComM_UpdatePncMappingTable` is called, ComM shall check if `ChannelCnt` matches the number of ComM channels where `ComMPncGatewayType` is set and `PncCnt` matches the number of configured `ComMPnc`. If one parameter does not match and `ComMDevErrorDetect` is set to TRUE ComM shall call `Det_ReportError` with `COMM_E_WRONG_PARAMETERS`. If one parameter does not match ComM shall return with `E_NOT_OK`.](SRS_ModeMgm_09259)

[SWS_ComM_01039] {DRAFT} [If `ComMDynamicPncToChannelMappingEnabled` is set to TRUE on at least one channel, when passed parameters match (see **[SWS_ComM_01038]**) and when PNC learning phase is active, then the function `ComM_UpdatePncMappingTable` shall return with `COMM_E_LEARNING_ACTIVE`.](SRS_ModeMgm_09259)

[SWS_ComM_01040] {DRAFT} [If `ComMDynamicPncToChannelMappingEnabled` is set to TRUE on at least one channel, when passed parameters match (see **[SWS_ComM_01038]**) and PNC learning phase is not active, then the function `ComM_UpdatePncMappingTable` shall merge for all PNCs the provided information with their current PNC-to-channel mappings whereby `MappingTable` shall be interpreted as a two-dimensional array with on first dimension all ComM channels where `ComMPncGatewayType` is set are handled according to their derived order in ComM and on second dimension all configured `ComMPnc` according to their order given by their `ComMPncId`. Additionally it shall return with `E_OK`.](SRS_ModeMgm_09259)

8.3.12 ComM_ResetPncToChannelMapping

[SWS_ComM_91017]{DRAFT} [

Service Name	ComM_ResetPncToChannelMapping (draft)	
Syntax	Std_ReturnType ComM_ResetPncToChannelMapping (void)	
Service ID [hex]	0x63	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Successfully reset PNC-to-channel-mapping to default E_NOT_OK: Reset of PNC-to-channel-mapping to default failed COMM_E_LEARNING_ACTIVE: Functionality cannot be executed because PNC learning phase is active.
Description	This function resets dynamic entries within the ECUs PNC-to-channel-mapping to default values Tags: atp.Status=draft	
Available via	ComM.h	

](SRS_ModeMgm_09259)

[SWS_ComM_01041] {DRAFT} [Function ComM_ResetPncToChannelMapping shall be only available if ComMPncGatewayEnabled and ComMDynamicPncToChannelMappingSupport are set to TRUE.
](SRS_ModeMgm_09258)

[SWS_ComM_01042] {DRAFT} [If ComMDynamicPncToChannelMappingEnabled is set to TRUE on at least one channel and when PNC learning phase is active, then the function ComM_ResetPncToChannelMapping shall return with COMM_E_LEARNING_ACTIVE.](SRS_ModeMgm_09259)

[SWS_ComM_01043] {DRAFT} [If ComMDynamicPncToChannelMappingEnabled is set to TRUE on at least one channel and when PNC learning phase is not active, then the function ComM_ResetPncToChannelMapping shall set the PNC-to-channel mappings to the default values from the original configuration (i.e. static entries) and return with E_OK.](SRS_ModeMgm_09259)

8.3.13 ComM_PnLearningRequest

[SWS_ComM_91019]{DRAFT} [

Service Name	ComM_PnLearningRequest (draft)	
Syntax	Std_ReturnType ComM_PnLearningRequest (void)	
Service ID [hex]	0x64	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Successfully started PNC Learning algorithm E_NOT_OK: PNC Learning algorithm could not be started COMM_E_LEARNING_ACTIVE: Functionality cannot be executed because PNC learning phase is active.
Description	Triggers the NM to return into NM Repeat Message State and send the Partial Network Learning Bit (in order for receiving nodes to respond) together with the Repeat Message Request Bit (in order for receiving nodes to return into NM Repeat Message State). This function is used for the optional Dynamic PNC-to-channel-mapping feature. Tags: atp.Status=draft	
Available via	ComM.h	

](SRS_ModeMgm_09260)

[SWS_ComM_01044] {DRAFT} [Function ComM_PnLearningRequest shall be only available if ComMDynamicPncToChannelMappingSupport is set to TRUE.

](SRS_ModeMgm_09258)

[SWS_ComM_01045] {DRAFT} [If ComMDynamicPncToChannelMappingSupport is set to TRUE on at least one channel and when PNC learning phase is active, then the function ComM_PnLearningRequest shall return with COMM_E_LEARNING_ACTIVE.](SRS_ModeMgm_09260)

[SWS_ComM_01058] {DRAFT} [If `ComM_PnLearningRequest` is called, PNC learning phase is inactive and at least one `ComMChannel` resides in another state than `COMM_FULL_COMMUNICATION`, then the function `ComM_PnLearningRequest` shall return with `E_NOT_OK`.](SRS_ModeMgm_09260)

Note: When `ComM_PnLearningRequest` is called, all relevant communication channels need to be already in `COMM_FULL_COMMUNICATION` state. This could be achieved by requesting an active diagnostic session via call of `ComM_DCM_ActiveDiagnostic()`. The learning phase may be triggered by a diagnostic tester.

[SWS_ComM_01046] {DRAFT} [If `ComMDynamicPncToChannelMappingSupport` is set to `TRUE` on at least one channel and when the PNC learning phase is not active, then the function `ComM_PnLearningRequest` shall call the API `Nm_PnLearningRequest` on all channels where `ComMDynamicPncToChannelMappingEnabled` is set to `TRUE` and return with `E_OK`.](SRS_ModeMgm_09260)

8.3.14 ComM_UpdatePncMembership

[SWS_ComM_91021]{DRAFT} [

Service Name	ComM_UpdatePncMembership (draft)	
Syntax	<pre>Std_ReturnType ComM_UpdatePncMembership (boolean Control, const uint8* PncMembership)</pre>	
Service ID [hex]	0x65	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Control	Boolean Parameter: 0 = Unset the corresponding Bits in PncBitMask 1 = Set the corresponding Bits in PncBitMask
	PncMembership	Array of uint8 with <PNC Vector Length> Elements that holds the current PNC Membership of the node
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: ComM_PncMembership successfully updated E_NOT_OK: Error occurred while updating the PNC membership. COMM_E_LEARNING_ACTIVE: Functionality cannot be

	executed because PNC learning phase is active.
Description	This function is used by SWCs to update the PNC membership which is transmitted during PNC Learning. This function is used for the optional Dynamic PNC-to-channel-mapping feature. This function is used for the optional Dynamic PNC-to-channel-mapping feature. Tags: atp.Status=draft
Available via	ComM.h

](SRS_ModeMgm_09263)

[SWS_ComM_01047] {DRAFT} [Function `ComM_UpdatePncMembership` shall be only available if `ComMDynamicPncToChannelMappingSupport` is set to TRUE.](SRS_ModeMgm_09258)

[SWS_ComM_01048] {DRAFT} [If `ComMDynamicPncToChannelMappingSupport` is set to TRUE on at least on channel and when PNC learning phase is active, then the function `ComM_UpdatePncMembership` shall return with `COMM_E_LEARNING_ACTIVE`.](SRS_ModeMgm_09260)

[SWS_ComM_01049] {DRAFT} [If `ComMDynamicPncToChannelMappingEnabled` is set to TRUE on at least on channel and PNC Learning phase is not active, then the function `ComM_UpdatePncMembership` shall perform the following actions:

- When `Control = 0`, then the current PNC membership shall be applied with logical AND (conjunction) operation on the parameter `PncMembership` (This would only unset the bits out of the `PncBitMask`)
- When `Control = 1`, then the current PNC membership shall be applied with logical OR (disjunction) operation on the parameter `PncMembership` (This would only set the bits out of the `PncBitMask`)
- Return with `E_OK`.

](SRS_ModeMgm_09260)

8.3.15 ComM_PreventWakeUp

[SWS_ComM_00156]

Service Name	<code>ComM_PreventWakeUp</code>
Syntax	<pre>Std_ReturnType ComM_PreventWakeUp (NetworkHandleType Channel, boolean Status)</pre>
Service ID [hex]	0x09
Sync/Async	Synchronous

Reentrancy	Non Reentrant	
Parameters (in)	Channel	See NetworkHandleType
	Status	FALSE: Wake up inhibition is switched off TRUE: Wake up inhibition is switched on
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Successfully changed wake up status for the channel E_NOT_OK: Change of wake up status for the channel failed, e.g. ComMEcuGroupClassification disables the functionality (see ECUC_ComM_00563)
Description	Changes the inhibition status COMM_NO_WAKEUP for the corresponding channel.	
Available via	ComM.h	

](SRS_ModeMgm_09157)

[SWS_ComM_00799] [Configuration of ComM_PreventWakeUp: Configurable with ComMWakeupInhibitionEnabled (see [ECUC_ComM_00559](#)).](SRS_ModeMgm_09089)

8.3.16 ComM_LimitChannelToNoComMode

[SWS_ComM_00163][

Service Name	ComM_LimitChannelToNoComMode	
Syntax	Std_ReturnType ComM_LimitChannelToNoComMode (NetworkHandleType Channel, boolean Status)	
Service ID [hex]	0x0b	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Channel	See NetworkHandleType
	Status	FALSE: Limit channel to COMM_NO_COMMUNICATION disabled TRUE: Limit channel to COMM_NO_COMMUNICATION enabled
Parameters (inout)	None	
Parameters (out)	None	

Return value	Std_Return-Type	E_OK: Successfully changed inhibition status for the channel E_NOT_OK: Change of inhibition status for the channel failed, e.g. ComMEcuGroupClassification disables the functionality (see ECUC_ComM_00563)
Description	Changes the inhibition status for the channel for changing from COMM_NO_COMMUNICATION to a higher Communication Mode. (See also ComM_LimitECUtoNoComMode, same functionality but for all channels)	
Available via	ComM.h	

](SRS_ModeMgm_09157)

[SWS_ComM_00800] [Configuration of ComM_LimitChannelToNoComMode: Configurable with ComMModeLimitationEnabled (see [ECUC_ComM_00560](#)) and ComMResetAfterForcingNoComm (see [ECUC_ComM_00558](#)).](SRS_ModeMgm_09071)

8.3.17 ComM_LimitECUtoNoComMode

[SWS_ComM_00124]

Service Name	ComM_LimitECUtoNoComMode	
Syntax	Std_ReturnType ComM_LimitECUtoNoComMode (boolean Status)	
Service ID [hex]	0x0c	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Status	FALSE: Limit ECU to COMM_NO_COMMUNICATION disabled TRUE: Limit ECU to COMM_NO_COMMUNICATION enabled
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Successfully changed inhibition status for the ECU E_NOT_OK: Change of inhibition status for the ECU failed, e.g. ComMEcuGroupClassification disables the functionality (see ECUC_ComM_00563)
Description	Changes the inhibition status for the ECU (=all channels) for changing from COMM_NO_COMMUNICATION to a higher Communication Mode. (See also ComM_LimitChannelToNoComMode, same functionality but for a specific channels)	
Available via	ComM.h	

](SRS_ModeMgm_09157)

[SWS_ComM_00801] [Configuration of ComM_LimitECUToNoComMode: Configurable with ComMModeLimitationEnabled (see [ECUC ComM 00560](#)) and ComMResetAfterForcingNoComm (see [ECUC ComM 00558](#))].(SRS_ModeMgm_09071)

8.3.18 ComM_ReadInhibitCounter

[SWS_ComM_00224]

Service Name	ComM_ReadInhibitCounter	
Syntax	Std_ReturnType ComM_ReadInhibitCounter (uint16* CounterValue)	
Service ID [hex]	0x0d	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	CounterValue	Amount of rejected COMM_FULL_COMMUNICATION user requests.
Return value	Std_Return-Type	E_OK: Successfully returned Inhibition Counter E_NOT_OK: Return of Inhibition Counter failed
Description	This function returns the amount of rejected COMM_FULL_COMMUNICATION user requests.	
Available via	ComM.h	

](SRS_ModeMgm_09156)

[SWS_ComM_00802] [Configuration of ComM_ReadInhibitCounter: Configurable with ComMModeLimitationEnabled (see [ECUC ComM 00560](#)). Function will only be available if ComMModeLimitationEnabled (see [ECUC ComM 00560](#)) is enabled and ComMGlobalNvMBlockDescriptor is configured.](SRS_ModeMgm_09156)

8.3.19 ComM_ResetInhibitCounter

[SWS_ComM_00108]

Service Name	ComM_ResetInhibitCounter	
Syntax	Std_ReturnType ComM_ResetInhibitCounter (void)	

Service ID [hex]	0x0e	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Successfully reset of Inhibit COMM_FULL_COMMUNICATION Counter E_NOT_OK: Reset of Inhibit COMM_FULL_COMMUNICATION Counter failed
Description	This function resets the Inhibited COMM_FULL_COMMUNICATION request Counter.	
Available via	ComM.h	

](SRS_ModeMgm_09156)

[SWS_ComM_00803] [Configuration of ComM_ResetInhibitCounter: Configurable with ComMModeLimitationEnabled (see [ECUC ComM_00560](#)). Function will only be available if ComMModeLimitationEnabled (see [ECUC ComM_00560](#)) is enabled and ComMGlobalNvMBlockDescriptor is configured.](SRS_ModeMgm_09155)

8.3.20 ComM_SetECUGroupClassification

[SWS_ComM_00552]

Service Name	ComM_SetECUGroupClassification	
Syntax	Std_ReturnType ComM_SetECUGroupClassification (ComM_InhibitionStatusType Status)	
Service ID [hex]	0x0f	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Status	See ComM_InhibitionStatusType
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Successfully change the ECU Group Classification Status E_NOT_OK: Change of the ECU Group Classification Status

	failed
Description	Changes the ECU Group Classification status (see chapter 10.2.2)
Available via	ComM.h

]()

8.3.21 ComM_GetVersionInfo

[SWS_ComM_00370]

Service Name	ComM_GetVersionInfo	
Syntax	<pre>void ComM_GetVersionInfo (Std_VersionInfoType* Versioninfo)</pre>	
Service ID [hex]	0x10	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	Versioninfo	See Std_VersionInfoType
Return value	None	
Description	This function returns the version information of this module	
Available via	ComM.h	

](SRS_BSW_00407)

8.4 Callback notifications

[SWS_ComM_00620] [All the provided indication functions shall be implemented pre-compile time.]()

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

8.4.1 AUTOSAR Network Management Interface

8.4.1.1 ComM_Nm_NetworkStartIndication

[SWS_ComM_00383]

Service Name	ComM_Nm_NetworkStartIndication
---------------------	--------------------------------

Syntax	void ComM_Nm_NetworkStartIndication (NetworkHandleType Channel)	
Service ID [hex]	0x15	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	See NetworkHandleType
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Indication that a NM-message has been received in the Bus Sleep Mode, what indicates that some nodes in the network have already entered the Network Mode.	
Available via	ComM_Nm.h	

10)

8.4.1.2 ComM_Nm_NetworkMode [SWS_ComM_00390]

Service Name	ComM_Nm_NetworkMode	
Syntax	<pre>void ComM_Nm_NetworkMode (NetworkHandleType Channel)</pre>	
Service ID [hex]	0x18	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	Channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Notification that the network management has entered Network Mode.	
Available via	ComM_Nm.h	

]()

8.4.1.3 ComM_Nm_PrepareBusSleepMode [SWS_ComM_00391]

Service Name	ComM_Nm_PrepareBusSleepMode	
Syntax	<pre>void ComM_Nm_PrepareBusSleepMode (NetworkHandleType Channel)</pre>	
Service ID [hex]	0x19	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	Channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Notification that the network management has entered Prepare Bus-Sleep Mode. Reentrancy: Reentrant (but not for the same NM-Channel)	
Available via	ComM_Nm.h	

l()

8.4.1.4 ComM_Nm_BusSleepMode [SWS_ComM_00392]

Service Name	ComM_Nm_BusSleepMode	
Syntax	<pre>void ComM_Nm_BusSleepMode (NetworkHandleType Channel)</pre>	
Service ID [hex]	0x1a	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	Channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Notification that the network management has entered Bus-Sleep Mode. This callback function should perform a transition of the hardware and transceiver to bus-sleep mode.	
Available via	ComM_Nm.h	

l()

8.4.1.5 ComM_Nm_RestartIndication [SWS_ComM_00792]

Service Name	ComM_Nm_RestartIndication	
Syntax	<pre>void ComM_Nm_RestartIndication (NetworkHandleType Channel)</pre>	
Service ID [hex]	0x1b	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	Channel
Parameters (inout)	None	

Parameters (out)	None
Return value	None
Description	If NmIf has started to shut down the coordinated busses, AND not all coordinated busses have indicated bus sleep state, AND on at least on one of the coordinated busses NM is restarted, THEN the NM Interface shall call the callback function ComM_Nm_RestartIndication with the nmNetworkHandle of the channels which have already indicated bus sleep state.
Available via	ComM_Nm.h

I()

8.4.1.6 ComM_Nm_RepeatMessageLeftIndication

[SWS_ComM_91024]{DRAFT} [

Service Name	ComM_Nm_RepeatMessageLeftIndication (draft)	
Syntax	<pre>void ComM_Nm_RepeatMessageLeftIndication (NetworkHandleType Channel)</pre>	
Service ID [hex]	0x66	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	See NetworkHandleType
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Notification that the state of all <BusNm> has left RepeatMessage. This interface is used to indicate by the optional Dynamic PNC-to-channel-mapping feature to indicate that learning phase ends. Tags: atp.Status=draft	
Available via	ComM_Nm.h	

](SRS_ModeMgm_09265)

8.4.1.7 ComM_Nm_PncLearningBitIndication

[SWS_ComM_91026]{DRAFT} [

Service Name	ComM_Nm_PncLearningBitIndication (draft)	
Syntax	<pre>void ComM_Nm_PncLearningBitIndication (NetworkHandleType Channel)</pre>	

Service ID [hex]	0x69	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	See NetworkHandleType
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Service to indicate that an NM message with set PNC Learning Bit has been received. Tags: atp.Status=draft	
Available via	ComM_Nm.h	

](SRS_ModeMgm_09261)

8.4.1.8 ComM_Nm_ForwardSynchronizedPncShutdown

[SWS_ComM_91003]{OBSOLETE} [

Service Name	ComM_Nm_ForwardSynchronizedPncShutdown (obsolete)	
Syntax	<pre>void ComM_Nm_ForwardSynchronizedPncShutdown (NetworkHandleType Channel)</pre>	
Service ID [hex]	0x6b	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	Channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	If an ECU in role of an intermediate PNC coordinator receives a PN shutdown message via a <Bus>Nm, then ComM is immediately indicated via ComM_Nm_ForwardSynchronizedPncShutdown to forward the request for a synchronized PNC shutdown of the affected PNCs. Therefore, ComM will immediately release the affected PNC state machines and forward the PN information to the affected ComM Channels and the corresponding NM channels, respectively. Note: This supports a nearly synchronized PNC shutdown across the PN topology from the top-level PNC coordinator down to the subordinated PNC node.	

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

](SRS_ModeMgm_09269)

[SWS_ComM_91030]{DRAFT} [

Service Name	ComM_Nm_ForwardSynchronizedPncShutdown (draft)	
Syntax	<pre>void ComM_Nm_ForwardSynchronizedPncShutdown (NetworkHandleType Channel, const uint8* PncBitVectorPtr)</pre>	
Service ID [hex]	0x6b	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	Channel
	PncBitVectorPtr	Pointer to PNC Bit vector with all PNC bits set to "1" which are indicated for a synchronized PNC shutdown
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	<p>If an ECU in role of an intermediate PNC coordinator receives a PN shutdown message via a <Bus>Nm, then ComM is immediately indicated via ComM_Nm_ForwardSynchronizedPncShutdown to forward the request for a synchronized PNC shutdown of the affected PNCs given by PncBitVectorPtr. Therefore, ComM will immediately release the affected PNC state machines and forward the PN information to the affected ComM Channels and the corresponding NM channels, respectively. Note: This supports a nearly synchronized PNC shutdown across the PN topology from the top-level PNC coordinator down to the subordinated PNC node.</p> <p>Tags: atp.Status=draft</p>	
Available via	ComM_Nm.h	

](SRS_ModeMgm_09269)

8.4.1.9 ComM_Nm_UpdateEIRA

[SWS_ComM_91028]{DRAFT} [

Service Name	ComM_Nm_UpdateEIRA (draft)
Syntax	<pre>void ComM_Nm_UpdateEIRA (const uint8* PncBitVectorPtr</pre>

)	
Service ID [hex]	0x6c	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	PncBit VectorPtr	Pointer to the PNC bit vector which contain the current aggregated internal and external PNC requests (EIRA)
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Function to indicate the current aggregated external / internal PNC request called by Nm. Tags: atp.Status=draft	
Available via	ComM_Nm.h	

](SRS_ModeMgm_09248, SRS_ModeMgm_09250)

8.4.1.10 ComM_Nm_UpdateERA

[SWS_ComM_91029]{DRAFT} [

Service Name	ComM_Nm_UpdateERA (draft)	
Syntax	<pre>void ComM_Nm_UpdateERA (NetworkHandleType Channel, const uint8* PncBitVectorPtr)</pre>	
Service ID [hex]	0x6d	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	Channel
	PncBitVectorPtr	PNC bit vector which contain the current external PNC requests (ERA) received on the given channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Function to indicate the current external PNC request per channel called by Nm. Tags: atp.Status=draft	

Available via	ComM_Nm.h
----------------------	-----------

](SRS_ModeMgm_09248, SRS_ModeMgm_09250)

8.4.2 AUTOSAR Diagnostic Communication Manager Interface

8.4.2.1 ComM_DCM_ActiveDiagnostic

[SWS_ComM_00873]

Service Name	ComM_DCM_ActiveDiagnostic	
Syntax	<pre>void ComM_DCM_ActiveDiagnostic (NetworkHandleType Channel)</pre>	
Service ID [hex]	0x1f	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	Channel needed for Diagnostic communication
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Indication of active diagnostic by the DCM.	
Available via	ComM_Dcm.h	

]()

8.4.2.2 ComM_DCM_InactiveDiagnostic

[SWS_ComM_00874]

Service Name	ComM_DCM_InactiveDiagnostic	
Syntax	<pre>void ComM_DCM_InactiveDiagnostic (NetworkHandleType Channel)</pre>	
Service ID [hex]	0x20	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	Channel no longer needed for Diagnostic communication
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	

Description	Indication of inactive diagnostic by the DCM.
Available via	ComM_Dcm.h

l()

8.4.3 AUTOSAR ECU State Manager Interface

8.4.3.1 ComM_EcuM_WakeUpIndication

[SWS_ComM_00275][

Service Name	ComM_EcuM_WakeUpIndication	
Syntax	<pre>void ComM_EcuM_WakeUpIndication (NetworkHandleType Channel)</pre>	
Service ID [hex]	0x2a	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	Channel
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Notification of a wake up on the corresponding channel.	
Available via	ComM_EcuM.h	

l()

8.4.3.2 ComM_EcuM_PNCWakeUpIndication

[SWS_ComM_91001][

Service Name	ComM_EcuM_PNCWakeUpIndication	
Syntax	<pre>void ComM_EcuM_PNCWakeUpIndication (PNCHandleType PNCid)</pre>	
Service ID [hex]	0x37	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	PNCid	Identifier of the partial network cluster
Parameters (inout)	None	
Parameters (out)	None	

Return value	None
Description	Notification of a wake up on the corresponding partial network cluster.
Available via	ComM_EcuM.h

]()

8.4.4 AUTOSAR ECU State Manager and Basic Software Mode Manager Interface

8.4.4.1 ComM_CommunicationAllowed [SWS_ComM_00871]

Service Name	ComM_CommunicationAllowed	
Syntax	<pre>void ComM_CommunicationAllowed (NetworkHandleType Channel, boolean Allowed)</pre>	
Service ID [hex]	0x35	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Channel	Channel
	Allowed	TRUE: Communication is allowed FALSE: Communication is not allowed
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	EcuM or BswM shall indicate to ComM when communication is allowed. If EcuM/ Flex is used: BswM	
Available via	ComM_BswM.h	

]()

8.4.5 Bus State Manager Interface

8.4.5.1 ComM_BusSM_ModeIndication [SWS_ComM_00675]

Service Name	ComM_BusSM_ModeIndication
Syntax	<pre>void ComM_BusSM_ModeIndication (NetworkHandleType Channel,</pre>

	<pre>ComM_ModeType ComMode)</pre>	
Service ID [hex]	0x33	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	See NetworkHandleType
	ComMode	See ComM_ModeType
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Indication of the actual bus mode by the corresponding Bus State Manager. ComM shall propagate the indicated state to the users with means of the RTE and BswM.	
Available via	ComM.h	

l()

8.4.5.2 ComM_BusSM_BusSleepMode [SWS_ComM_91000]

Service Name	ComM_BusSM_BusSleepMode	
Syntax	<pre>void ComM_BusSM_BusSleepMode (NetworkHandleType Channel)</pre>	
Service ID [hex]	0x34	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	Channel	Identification of the channel
	None	
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Notification of the corresponding Bus State Manager that the actual bus mode is Bus-Sleep. Only applicable for ComM channels with ComMNmVariant set to SLAVE_ACTIVE or	

	SLAVE_PASSIVE. E.g. LIN slaves (ComMNMVariant = SLAVE_ACTIVE) or Ethernet channels with OA TC10 compliant Ethernet hardware which act as passive communication slave (ComMNMVariant = SLAVE_PASSIVE and EthTrcvActAsSlavePassiveEnabled set to TRUE)
Available via	ComM.h

]()

8.4.6 COM Interface (OBSOLETE)

[SWS_ComM_00819]{OBSOLETE} [

Service Name	ComM_COMCbk_<sn> (obsolete)
Syntax	void ComM_COMCbk_<sn> (void)
Service ID [hex]	0x36
Sync/Async	Synchronous
Reentrancy	Non Reentrant
Parameters (in)	None
Parameters (inout)	None
Parameters (out)	None
Return value	None
Description	This callback is called when the EIRA or ERA was updated in COM. The call only informs the ComM about ERA and EIRA changes. The actual handling is done in the next call to ComM_MainFunction_<ComMChannel.ShortName> with changing the corresponding PN State machine. Tags: atp.Status=obsolete
Available via	ComM_Com.h

]()

8.5 Scheduled functions

These functions are directly called by Basic Software Scheduler. The following functions shall have no return value and no parameter. All functions shall be non reentrant.

8.5.1 ComM_MainFunction

[SWS_ComM_00429]

Service Name	ComM_MainFunction_<ComMChannel.ShortName>
Syntax	void ComM_MainFunction_<ComMChannel.ShortName> (void)
Service ID [hex]	0x60
Description	This function shall perform the processing of the AUTOSAR ComM activities that are not directly initiated by the calls e.g. from the RTE. There shall be one dedicated Main Function for each channel of ComM. Precondition: ComM shall be initialized
Available via	SchM_ComM.h

](SRS_BSW_00373)

[SWS_ComM_00818] [Channel.ShortName shall be used to configure ComM_MainFunction_<ComMChannel.ShortName> (see section 10.2.2) .]()

Note: ComMChannel.ShortName is the short name of the ComMChannel container that will be managed by the ComM_MainFunction_<ComMChannel.ShortName> function

8.6 Expected interfaces

In this chapter all interfaces required from other modules are shown. An overview of the required interfaces is shown in Figure 1.

8.6.1 Mandatory Interfaces

This chapter defines all interfaces which are required to fulfil the core functionality of the module.

[SWS_ComM_00828] [

API function	Header File	Description
Nm_PassiveStartUp	Nm	This function calls the <BusNm>_PassiveStartUp function (e.g. CanNm_PassiveStartUp function is called if channel is configured as CAN).
Nm_NetworkRequest	Nm	This function calls the <BusNm>_NetworkRequest (e.g. CanNm_NetworkRequest function is called if channel is configured as CAN).
Nm_NetworkRelease	Nm	This function calls the <BusNm>_NetworkRelease bus specific function (e.g. CanNm_NetworkRelease

		function is called if channel is configured as CAN).
Dcm_ComM_NoComModeEntered	Dcm	This call informs the Dcm module about a ComM mode change to COMM_NO_COMMUNICATION.
Dcm_ComM_SilentComModeEntered	Dcm	This call informs the Dcm module about a ComM mode change to COMM_SILENT_COMMUNICATION.
Dcm_ComM_FullComModeEntered	Dcm	This call informs the Dcm module about a ComM mode change to COMM_FULL_COMMUNICATION.
Rte_Ports_UserMode_P()[n].Switch_currentMode(RTE_MODE_ComMMode_COMM_NO_COMMUNICATION)	Rte	Indicate COMM_NO_COMMUNICATION mode to RTE
Rte_Ports_UserMode_P()[n].Switch_currentMode(RTE_MODE_ComMMode_COMM_SILENT_COMMUNICATION)	Rte	Indicate COMM_SILENT_COMMUNICATION mode to RTE
Rte_Ports_UserMode_P()[n].Switch_currentMode(RTE_MODE_ComMMode_COMM_FULL_COMMUNICATION)	Rte	Indicate COMM_FULL_COMMUNICATION mode to RTE
BswM_ComM_CurrentMode	BswM	Indicate Communication Mode to BswM
NvM_ReadBlock	NvM	NVRAM manager API for Read block
NvM_WriteBlock	NvM	NVRAM manager API for Write block
NvM_GetErrorStatus	NvM	NVRAM manager API for Get status
<BusSM>_GetCurrentComMode	<BusSM>	Function to query the actual communication mode from the <Bus> State Manager.
<BusSM>_RequestComMode	<BusSM>	Function to request a communication mode from the <Bus> State Manager.

()

8.6.1.1 AUTOSAR NVRAM Manager module

[SWS_ComM_00103] [The ComM module shall use the corresponding standardized services of the NVRAM Manager module (see [SWS_ComM_00828](#)) for storing and reading non-volatile configuration data `ComMNoWakeup` (see [ECUC_ComM_00569](#)), `ComMEcuGroupClassification` (see [ECUC_ComM_00563](#)), inhibition status (see [SWS_ComM_00157](#)), the Inhibit counter (see [SWS_ComM_00140](#)), the PNC-to-channel Mapping (see **[SWS_ComM_01040]**) and the PNC membership (see **[SWS_ComM_01049]**).]()

Comment: See [SWS_ComM_00864](#) and [SWS_ComM_00865](#) when configuration data shall be read and stored

For details refer to the AUTOSAR NVRAM Manager module Specification [7].

8.6.1.2 AUTOSAR Bus State Manager

[SWS_ComM_00962] [The prefix for the StateManager APIs ("<BusSm>") shall be CanSM, LinSM, FrSM, EthSM if the Parameter `ComMBusType` is `COMM_BUS_TYPE_CAN`, `COMM_BUS_TYPE_LIN`, `COMM_BUS_TYPE_FR` or `COMM_BUS_TYPE_ETH` accordingly.](SRS_ModeMgm_09155)

[SWS_ComM_00957] [If `ComMBusType` = "`COMM_BUS_TYPE_CDD`" the API prefix ("<BusSm>") shall be configured in the Parameter "`ComMCDDBusPrefix`".](SRS_ModeMgm_09207)

[SWS_ComM_00963] [The Communication Manager module shall use <BusSm>_GetCurrentComMode() from the State Manager to query the current communication mode if necessary.]()

[SWS_ComM_00958] [The Communication Manager module shall use <BusSm>_RequestComMode() from the State Manager to request a dedicated communication mode.]()

When it is necessary to request a dedicated communication mode depends on the current status of each instance of the channel state machine (see above).

For details of the functionality of the Bus State Manager modules refer to their Specification [\[23\]](#), [\[24\]](#), [\[25\]](#), [\[28\]](#).

Comment: Those APIs can be called re-entrant, as long as different channel & controller numbers are used.

8.6.1.3 AUTOSAR Network Management Interface

[SWS_ComM_00261] [The ComM module shall use the corresponding functions to synchronize the bus start-up and shutdown of the Network Management (see [SWS_ComM_00828](#)).

For details refer to the AUTOSAR NM Interface Specification [9].]()

8.6.1.4 AUTOSAR Diagnostic Communication Manager Module

[SWS_ComM_00266] [The ComM module shall use the corresponding functions provided by DCM (see [SWS_ComM_00828](#)) to control the communication capabilities of the DCM module.]()

Comment: DCM provides no functions to start/stop transmission and reception. DCM ensures to control communication according the indicated Communication Manager Module states.

For details refer to the AUTOSAR DCM Specification [11].

8.6.1.5 AUTOSAR RTE interface provided by RTE to ComM for the SW-C

[SWS_ComM_00091] [The ComM module shall use the corresponding function provided by RTE to indicate modes to the users. There shall be one indication per user. Fan-out in case of a mode indication related to more than one user shall be done by the Communication Manager Module.](SRS_ModeMgm_09085)

[SWS_ComM_00663] [If more than one channel is linked to one user request and the modes of the channels are different, the ComM module shall indicate the lowest mode to the user.](SRS_ModeMgm_09085)

[SWS_ComM_00662] [The sequence of users shall start with user 0 up to user N and the name of the mode ports shall be UM000, UM001, ... UM<N>.](SRS_ModeMgm_09090)

Rationale for [SWS_ComM_00662](#): It shall be possible to use the port based API also to address specific users directly.

Comment: Within the array of ports, the ports are named alphabetically.

[SWS_ComM_00778] [The ComM module shall explicitly indicate changes in modes to each individual user, to which a SW-C is connected. The ComM module shall do this by calling the right API on the RTE through the ports “UMnnn”.](SRS_ModeMgm_09085)

Comment: There is one such port per configured user to which a SW-C is connected. For users not used by SW-Cs (e.g. the users created due to [ECUC ComM 00840](#)) no mode port will be created.

Implementation Hint: An implementation of the ComM module could use any of the normal RTE-mechanisms to signal changes in the mode to the users. Given the specific configurability of the Communication Manager Module, using the RTE “Indirect API” seems most appropriate. This works as follows (consult the RTE specification for details).

An implementation of the Communication Manager Module can use the “Rte_Ports” API to obtain an array of the “UMnnn” ports at run-time:

```
/* Return an array of all ports that provide the interface ComM_CurrentMode. Because of the specific naming conventions chosen, the element n in this array of ports will reference to the port UM<nnn>. For example userModePorts[1] will be a handle on port UM001 */
```

```
userModePorts = Rte_Ports_ComM_CurrentMode_P();
```

The number of such userModePorts can be obtained through the call `Rte_NPorts_ComM_CurrentMode_P()`. This value corresponds to the size of the `COMM_USER_LIST` array.

To signal that a user `n` is in a new mode, the Communication Manager Module should:
`userModePorts[n].Switch_currentMode(newMode)`

For details refer to the AUTOSAR RTE specification [8] and AUTOSAR Services Mode Management specification [21].

8.6.1.6 Basic Software Mode Manager (BswM)

[SWS_ComM_00861] [The ComM module shall use the corresponding function provided by BswM to report the states of Communication Manager Module channels (see [SWS ComM 00828](#)).]()

For details refer to AUTOSAR Basic Software Mode Manager module [29] .

8.6.2 Optional Interfaces

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

[SWS_ComM_00829]

API Function	Header File	Description
BswM_ComM_CurrentPNCMode	BswM_ComM.h	Function is called by ComM to indicate the current mode of the PNC.

BswM_ComM_-InitiateReset	BswM_ComM.h	Function is called by ComM to signal a shutdown.
Com_ReceiveSignal	Com.h	Com_ReceiveSignal copies the data of the signal identified by SignalId to the location specified by SignalDataPtr.
Com_SendSignal	Com.h	The service Com_SendSignal updates the signal object identified by SignalId with the signal referenced by the SignalDataPtr parameter.
Nm_PnLearningRequest (draft)	Nm.h	Set Repeat Message Request Bit and Partial Network Learning Bit for NM messages transmitted next on the bus. For that purpose <BusNm>_PnLearningRequest shall be called (e.g. CanNm_PnLearningRequest function if channel is configured as CAN). This will force all nodes to enter the PNC Learning Phase and re-enter Repeat Message Stat. This is needed for the optional Dynamic PNC-to-channel-mapping feature. Tags: atp.Status=draft
Nm_RequestSynchronizedPncShutdown (draft)	Nm.h	This function forward the request for a synchronized PNC shutdown of a particular PNC given by PncId to the affected <Bus>Nm by calling <Bus>Nm_RequestSynchronizedPncShutdown The function call is only valid if NmStandardBusType is not set to NM_BUSNM_LOCALNM (e.g. CanNm_RequestSynchronizedPncShutdown function is called for NM_BUSNM_CANNM). Tags: atp.Status=draft
Nm_UpdateIRA (draft)	Nm.h	Indication by ComM of internal PNC requests. This is used to aggregate the internal PNC requests. Tags: atp.Status=draft

J()

Note: Optional interfaces from Com module are considered obsolete in R21-11 due to a concept part, which will be validated after the release of R21-11.

8.6.2.1 AUTOSAR DET

The Communication Manager module shall use Det_ReportError from the Default Error Tracer Module to report development errors.

8.6.3 Configurable Interfaces

None.

8.7 Service Interfaces

8.7.1 Sender-Receiver-interfaces

8.7.1.1 ComM_CurrentChannelRequest

[SWS_ComM_00904]

Name	ComM_CurrentChannelRequest_{channel_name}
Comment	Array of ComMUserIdentifier, that currently hold FULL_COM requests for this channel.

	The size of the attribute fullComRequestors.handleArray is NUM_COMM_USER_PER_CHANNEL	
IsService	true	
Variation	{ecuc(ComM/ComMConfigSet/ComMChannel/ComMFullCommRequestNotificationEnabled)} == true channel_name = {ecuc(ComM/ComMConfigSet/ComMChannel.SHORT-NAME)}	
Data Elements	fullComRequestors	
	Type	ComM_UserHandleArrayType_{channel_name}
	Variation	channel_name = {ecuc(ComM/ComMConfigSet/ComMChannel.SHORT-NAME)}

]()

8.7.1.2 ComM_UpdatePncMembership

[SWS_ComM_91100]{OBSOLETE} [

Name	ComM_UpdatePncMembership (obsolete)	
Comment	Clear (Control = FALSE) or Add (Control = TRUE) values of the PNC Membership Tags: atp.Status=obsolete	
IsService	true	
Variation	{ecuc(ComM/ComMGeneral/ComMDynamicPncToChannelMappingSupport)} == true	
Data Elements	Control	
	Type	boolean
	Variation	--
	PncMemberShip	
	Type	ComM_PncMembershipType
	Variation	--

](SRS_ModeMgm_09263)

8.7.1.3 ComM_PnLearningRequest

[SWS_ComM_91101]{OBSOLETE} [

Name	ComM_PnLearningRequest (obsolete)	
Comment	-- Tags: atp.Status=obsolete	
IsService	true	
Variation	{ecuc(ComM/ComMGeneral/ComMDynamicPncToChannelMappingSupport)} == true	
Data Elements	NetworkHandle	
	Type	NetworkHandleType

	Variation	--
--	------------------	----

](SRS_ModeMgm_09260)

8.7.2 Client-Server-interfaces

8.7.2.1 ComM_ChannelLimitation

[SWS_ComM_00743]

Name	ComM_ChannelLimitation		
Comment	A SW-C playing the role of a "Mode Manager" can use this interface to configure the Communication Manager Module to inhibit communication mode for a given channel.		
IsService	true		
Variation	{ecuc(ComM/ComMGeneral.ComMModeLimitationEnabled)} == true		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed

Operation	GetInhibitionStatus		
Comment	returns the inhibition status of a channel		
Variation	--		
Parameters	Status		
	Type	ComM_InhibitionStatusType	
	Direction	OUT	
	Comment	--	
	Variation	--	
Possible Errors	E_OK E_NOT_OK		

Operation	LimitChannelToNoComMode		
Comment	Changes the inhibition status for the channel for changing from COMM_NO_COMMUNICATION to a higher Communication Mode. (See also ComM_LimitECUToNoComMode, same functionality but for all channels)		
Variation	--		
Parameters	Status		
	Type	boolean	
	Direction	IN	

	Comment	FALSE: Limit channel to COMM_NO_COMMUNICATION disabled TRUE: Limit channel to COMM_NO_COMMUNICATION enabled
	Variation	--
Possible Errors	E_OK E_NOT_OK	

l()

8.7.2.2 ComM_ChannelWakeup [SWS_ComM_00742]

Name	ComM_ChannelWakeup		
Comment	A SW-C playing the role of a "Mode Manager" can use this interface to configure the Communication Manager Module to take precautions against awakening other ECU's by starting the communication.		
IsService	true		
Variation	{ecuc(ComM/ComMGeneral.ComMWakeupInhibitionEnabled)} == true		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed

Operation	GetInhibitionStatus		
Comment	returns the inhibition status of a channel		
Variation	--		
Parameters	Status		
	Type	ComM_InhibitionStatusType	
	Direction	OUT	
	Comment	--	
	Variation	--	
Possible Errors	E_OK E_NOT_OK		

Operation	PreventWakeUp		
Comment	Changes the inhibition status COMM_NO_WAKEUP for the corresponding channel.		
Variation	--		
Parameters	Status		
	Type	boolean	

	Direction	IN
	Comment	--
	Variation	--
Possible Errors	E_OK E_NOT_OK	

]()

8.7.2.3 ComM_ECUModeLimitation

[SWS_ComM_00741]

Name	ComM_ECUModeLimitation		
Comment	A SW-C which plays the role of a "Mode Manager" can use this interface to change the behavior of the entire ECU.		
IsService	true		
Variation	{ecuc(ComM/ComMGeneral.ComMModeLimitationEnabled)} == true		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed

Operation	LimitECUtoNoComMode		
Comment	Changes the inhibition status for the ECU (=all channels) for changing from COMM_NO_COMMUNICATION to a higher Communication Mode. (See also ComM_LimitChannelToNoComMode, same functionality but for a specific channels)		
Variation	--		
Parameters	Status		
	Type	boolean	
	Direction	IN	
	Comment	FALSE: Limit ECU to COMM_NO_COMMUNICATION disabled TRUE: Limit ECU to COMM_NO_COMMUNICATION enabled	
	Variation	--	
Possible Errors	E_OK E_NOT_OK		

Operation	ReadInhibitCounter		
Comment	returns the value of the 'inhibited full communication request counter'		
Variation	{ecuc(ComM/ComMGeneral.ComMGlobalInVMBlockDescriptor)} != NULL		
Parameters	CounterValue		

	Type	uint16
	Direction	OUT
	Comment	--
	Variation	--
Possible Errors	E_OK E_NOT_OK	

Operation	ResetInhibitCounter
Comment	reset the "inhibited full communication request counter"
Variation	{ecuc(ComM/ComMGeneral.ComMGlobalInVMBlockDescriptor)} != NULL
Possible Errors	E_OK E_NOT_OK

Operation	SetECUGroupClassification	
Comment	changes the ECU group classification status	
Variation	--	
Parameters	Status	
	Type	ComM_InhibitionStatusType
	Direction	IN
	Comment	--
	Variation	--
Possible Errors	E_OK E_NOT_OK	

]()

8.7.2.4 ComM_UserRequest [SWS_ComM_01000]

Name	ComM_UserRequest		
Comment	A SW-C that wants to explicitly direct the local Communication Manager Module of the ECU towards a certain state requires the client-server interface ComM_UserRequest. Through this interface, the SW-C could either set the desired state of all communication channels (if the user is mapped to one or more channels) or of all PNCs (if the user is mapped to one or more PNCs) that are relevant for that component to "No Communication" or "Full Communication".		
IsService	true		
Variation	--		
Possible	0	E_OK	Operation successful

Errors	1	E_NOT_OK	Operation failed
	2	E_MODE_LIMITATION	ComMMode cannot be granted because of ComMMode inhibition
	3	E_MULTIPLE_PNC_ASSIGNED	Operation is not possible since multiple PNCs are assigned to the affected ComMUser
	4	E_NO_PNC_ASSIGNED	Operation is not possible since no PNC is assigned to the affected ComMUser

Operation	GetCurrentComMode	
Comment	Returns the current Communication Manager Module mode for the SW-C-Return the current Communication Manager Modul channel mode to the SW-C. Please note: the channel mode is returned. Even though the affected user is assigned to a PNC. (see ComM_GetCurrentComMode)	
Variation	--	
Parameters	ComMode	
	Type	ComM_ModeType
	Direction	OUT
	Comment	--
	Variation	--
Possible Errors	E_OK E_NOT_OK	

Operation	GetCurrentPNCComMode	
Comment	Return the current Communication Manager Modul PNC mode to the SW-C. Please note: the PNC mode is returned as ComM_ModeType (COMM_NO_COMMUNICATION == COMM_PNC_NO_COMMUNICATIO, COMM_FULL_COMMUNICATION == COMM_PNC_FULL_COMMUNICATION). If the affected ComM user is mapped to multiple PNCs than the operation shall return E_MULTIPLE_PNC_ASSIGNED. If the affected ComM user is mapped to no PNC than the operation shall return E_NO_PNC_ASSIGNED.	
Variation	--	
Parameters	ComMode	
	Type	ComM_ModeType
	Direction	OUT
	Comment	--
	Variation	--
Possible Errors	E_OK E_NOT_OK E_MULTIPLE_PNC_ASSIGNED	

	E_NO_PNC_ASSIGNED
--	-------------------

Operation	GetMaxComMode	
Comment	Returns the current Communication Manager Module mode for the SW-C	
Variation	--	
Parameters	ComMode	
	Type	ComM_ModeType
	Direction	OUT
	Comment	--
	Variation	--
Possible Errors	E_OK E_NOT_OK	

Operation	GetRequestedComMode	
Comment	Returns that last Communication Manager Module Mode requested by the SW-C	
Variation	--	
Parameters	ComMode	
	Type	ComM_ModeType
	Direction	OUT
	Comment	--
	Variation	--
Possible Errors	E_OK E_NOT_OK	

Operation	RequestComMode	
Comment	The SW-C requests that all communication channels it needs are in the provided Communication Manager Module mode	
Variation	--	
Parameters	ComMode	
	Type	ComM_ModeType
	Direction	IN
	Comment	--
	Variation	--

Possible Errors	E_OK E_NOT_OK E_MODE_LIMITATION
------------------------	---------------------------------------

]()

8.7.2.5 ComM_PncToChannelMapping [SWS_ComM_91102]

Name	ComM_PncToChannelMapping		
Comment	Client-server interface to get, update or clear the PNC-to-channel-mapping		
IsService	true		
Variation	{ecuc(ComM/ComMGeneral/ComMDynamicPncToChannelMappingSupport)} == true		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	5	E_LEARNING_ACTIVE	Operation not possible as PNC Learning Phase is active

Operation	GetPncToChannelMapping		
Comment	Returns the current PNC-to-channel-mapping Tags: atp.Status=draft		
Variation	--		
Parameters	MappingTable		
	Type	boolean*	
	Direction	OUT	
	Comment	Pointer to two-dimensional array with the current Pnc-to-channel-mapping of the PNC Gateway where the first dimension covers all relevant channels and the second all relevant PNCs.	
	Variation	--	
	ChannelCnt		
	Type	uint8	
	Direction	OUT	
	Comment	--	
	Variation	--	
	PncCnt		
	Type	uint8	
Direction	OUT		

	Comment	--
	Variation	--
Possible Errors	E_OK E_NOT_OK E_LEARNING_ACTIVE	

Operation	ResetPncToChannelMapping	
Comment	Resets the current PNC-to-channel mapping to its static configured default Tags: atp.Status=draft	
Variation	--	
Possible Errors	E_OK E_NOT_OK E_LEARNING_ACTIVE	

Operation	UpdatePncToChannelMapping	
Comment	Updates the current PNC-to-channel-mapping Tags: atp.Status=draft	
Variation	--	
Parameters	MappingTable	
	Type	const boolean*
	Direction	IN
	Comment	Pointer to two-dimensional array with the current Pnc-to-channel-mapping of the PNC Gateway where the first dimension covers all relevant channels and the second all relevant PNCs.
	Variation	--
	channelCnt	
	Type	uint8
	Direction	IN
	Comment	--
	Variation	--
	PncCnt	
	Type	uint8
	Direction	IN
	Comment	--
Variation	--	

Possible Errors	E_OK E_NOT_OK E_LEARNING_ACTIVE
------------------------	---------------------------------------

](SRS_ModeMgm_09259)

8.7.2.6 ComM_DynamicPncToChannelMapping [SWS_ComM_91108]

Name	ComM_DynamicPncToChannelMapping		
Comment	A SW-C can use this interface in order to update during runtime the PNC membership and trigger a learning request by sending NM messages with Partial Network Learning and Repeat Message Request bits set.		
IsService	true		
Variation	{ecuc(ComM/ComMGeneral/ComMDynamicPncToChannelMappingSupport)} == true		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	5	E_LEARNING_ACTIVE	Operation not possible as PNC Learning Phase is active

Operation	ComM_PnLearningRequest
Comment	Triggers a learning request
Variation	--
Possible Errors	E_OK E_NOT_OK E_LEARNING_ACTIVE

Operation	ComM_UpdatePncMembership		
Comment	Used by SWCs to update the PNC membership which is transmitted during PNC Learning		
Variation	--		
Parameters	Control		
	Type	boolean	
	Direction	IN	
	Comment	--	
	Variation	--	
	PncMembership		
Type	const uint8*		

	Direction	IN
	Comment	Array of uint8 with <PNC Vector Length> Elements that holds the current PNC Membership of the node.
	Variation	--
Possible Errors	E_OK E_NOT_OK E_LEARNING_ACTIVE	

]()

8.7.3 Mode-Switch-Interfaces

8.7.3.1 ComM_CurrentMode

[SWS_ComM_01001]

Name	ComM_CurrentMode	
Comment	A SW-C that wants to get informed about its current Communication Manager Module Mode requires the ModeSwitchInterface ComM_CurrentMode.	
IsService	true	
Variation	--	
Mode Group	currentMode	ComMMode

]()

8.7.4 Implementation Data Types

8.7.4.1 ComM_InhibitionStatusType

[SWS_ComM_00669]

Name	ComM_InhibitionStatusType			
Kind	Bitfield			
Derived from	uint8			
Elements	Kind	Name	Mask	Description
	bit	WakeupInhibitionActive	0x01	Bit 0 (LSB): Wake Up inhibition active
	bit	LimitedToNoCom	0x02	Bit 1: Limit to COMM_NO_COMMUNICATION mode
Description	Defines whether a mode inhibition is active or not. Inhibition status of ComM. e.g. status=00000011 -> Wake up inhibition and limitation to COMM_NO_			

	COMMUNICATION mode active
Variation	--
Available via	Rte_ComM_Type.h

]()

8.7.4.2 ComM_ModeType

[SWS_ComM_00672]

Name	ComM_ModeType		
Kind	Type		
Derived from	uint8		
Range	COMM_NO_COMMUNICATION	0	ComM state machine is in "No Communication" mode. Configured channel shall have no transmission or reception capability.
	COMM_SILENT_COMMUNICATION	1	ComM state machine is in "Silent Communication" mode. Configured channel shall have only reception capability, no transmission capability.
	COMM_FULL_COMMUNICATION	2	ComM state machine is in "Full Communication" mode. Configured channel shall have both transmission and reception capability.
	COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST	3	ComM state machine is in "Full Communication" mode. Configured channel shall have both transmission and reception towards the lower layer (e.g. Ethernet hardware compliant to OA TC10). This is only for internal use within the ComM channel statemachine.
Description	Current mode of the Communication Manager (main state of the state machine).		
Variation	--		
Available via	Rte_ComM_Type.h		

]()

8.7.4.3 ComM_UserHandleType

[SWS_ComM_00670]

Name	ComM_UserHandleType
Kind	Type
Derived from	uint8
Description	Handle to identify a user. For each user, a unique value must be defined at system generation time. Maximum number of users is 255. Legal user IDs are in the range 0 .. 254; user ID 255 is reserved and shall have the symbolic representation COMM_NOT_USED_USER_ID.

Variation	--
Available via	Rte_ComM_Type.h

}]()

8.7.4.4 ComM_UserHandleArrayType

[SWS_ComM_00906]

Name	ComM_UserHandleArrayType_{channel_name}	
Kind	Structure	
Elements	numberOfRequesters	
	Type	uint8
	Comment	--
	handleArray	
	Type	ComM_UserHandleSubArrayType_{channel_name}
	Comment	--
	Variation	channel_name = {ecuc(ComM/ComMConfigSet/ComMChannel.SHORT-NAME)}
Description	numberOfRequesters contains the number of valid user handle entries in the "handle Array" member. If no user keeps the channel requested, this is zero {LOWER-LIMIT=0, UPPER-LIMIT= MAX_CHANNEL_REQUESTER }	
Variation	channel_name = {ecuc(ComM/ComMConfigSet/ComMChannel.SHORT-NAME)}	
Available via	Rte_ComM_Type.h	

}]()

8.7.4.5 ComM_UserHandleSubArrayType

[SWS_ComM_01005]

Name	ComM_UserHandleSubArrayType_{channel_name}		
Kind	Array	Element type	ComM_UserHandleType
Size	COUNT{ecuc(ComM/ComMConfigSet/ComMChannel/ComMUserPerChannel)} Elements		
Description	This element contains the user handles of the users which keep the channel requested (if any), starting in its first entries. The size of the array MAX_CHANNEL_REQUESTERS is the maximum of the number of users requesting a channel.		
Variation	channel_name = {ecuc(ComM/ComMConfigSet/ComMChannel.SHORT-NAME)}		
Available via	Rte_ComM_Type.h		

}]()

8.7.4.6 ComM_PncMembershipType

[SWS_ComM_91103]{OBSOLETE} [

Name	ComM_PncMembershipType (obsolete)
Kind	Const Pointer
Type	const uint8*
Description	Array of uint8 with <PNC Vector Length> Elements that holds the current PNC Membership of the node. Tags: atp.Status=obsolete
Variation	{ecuc(ComM/ComMGeneral/ComMDynamicPncToChannelMappingSupport)} == true
Available via	Rte_ComM_Type.h

]()

8.7.5 Ports

8.7.5.1 ComM_CL

[SWS_ComM_01006][

Name	CL_{channel_name}		
Kind	Provided Port	Interface	ComM_ChannelLimitation
Description	--		
Port Defined Argument Value(s)	Type	NetworkHandleType	
	Value	{ecuc(ComM/ComMConfigSet/ComMChannel/ComMChannelId.value)}	
Variation	{ecuc(ComM/ComMGeneral.ComMModelLimitationEnabled)} == true channel_name = {ecuc(ComM/ComMConfigSet/ComMChannel)}		

]()

8.7.5.2 ComM_CR

[SWS_ComM_01007][

Name	CR_{channel_name}		
Kind	ProvidedPort	Interface	ComM_CurrentChannelRequest_{channel_name}
Description	--		
Variation	{ecuc(ComM/ComMConfigSet/ComMChannel/ComMFullCommRequestNotificationEnabled)} == true channel_name = {ecuc(ComM/ComMConfigSet/ComMChannel.SHORT-NAME)}		

]()

8.7.5.3 ComM_CW

[SWS_ComM_01008]

Name	CW_{channel_name}		
Kind	Provided Port	Interface	ComM_ChannelWakeup
Description	--		
Port Defined Argument Value(s)	Type	NetworkHandleType	
	Value	{ecuc(ComM/ComMConfigSet/ComMChannel/ComMChannelId.value)}	
Variation	{ecuc(ComM/ComMGeneral.ComMWakeupInhibitionEnabled)} == true channel_name = {ecuc(ComM/ComMConfigSet/ComMChannel)}		

]()

8.7.5.4 ComM_modeLimitation

[SWS_ComM_01009]

Name	modeLimitation		
Kind	ProvidedPort	Interface	ComM_ECUModeLimitation
Description	--		
Variation	{ecuc(ComM/ComMGeneral.ComMModeLimitationEnabled)} == true		

]()

8.7.5.5 ComM_UM

[SWS_ComM_01010]

Name	UM_{user_name}		
Kind	ProvidedPort	Interface	ComM_CurrentMode
Description	--		
Variation	user_name = {ecuc(ComM/ComMConfigSet/ComMUser.SHORT-NAME)}		

]()

8.7.5.6 ComM_UR

[SWS_ComM_01011]

Name	UR_{user_name}		
Kind	Provided Port	Interface	ComM_UserRequest
Description	--		
Port Defined Argument Value(s)	Type	ComM_UserHandleType	
	Value	ecuc(ComM/ComMConfigSet/ComMUser/ComMUserIdentifier.value)}	

Variation	user_name = {ecuc(ComM/ComMConfigSet/ComMUser.SHORT-NAME)}
------------------	--

]()

8.7.5.7 ComM_UpdatePncMembership

[SWS_ComM_91105]{OBSOLETE} [

Name	UpdatePncMembership (obsolete)		
Kind	ProvidedPort	Interface	ComM_UpdatePncMembership
Description	-- Tags: atp.Status=obsolete		
Variation	{ecuc(ComM/ComMGeneral/ComMDynamicPncToChannelMappingSupport)} == true		

] (SRS_ModeMgm_09259)

8.7.5.8 ComM_PnLearningRequest

[SWS_ComM_91106]{OBSOLETE} [

Name	PnLearningRequest (obsolete)		
Kind	ProvidedPort	Interface	ComM_PnLearningRequest
Description	-- Tags: atp.Status=obsolete		
Variation	{ecuc(ComM/ComMGeneral/ComMDynamicPncToChannelMappingSupport)} == true		

] (SRS_ModeMgm_09260)

8.7.5.9 ComM_PncToChannelMapping

[SWS_ComM_91107][

Name	PncToChannelMapping		
Kind	ProvidedPort	Interface	ComM_PncToChannelMapping
Description	--		
Variation	{ecuc(ComM/ComMGeneral/ComMDynamicPncToChannelMappingSupport)} == true		

] (SRS_ModeMgm_09259)

8.7.5.10 ComM_DynamicPncToChannelMapping

[SWS_ComM_91109][

Name	ComM_DynamicPncToChannelMapping		
Kind	ProvidedPort	Interface	ComM_DynamicPncToChannelMapping
Description	--		
Variation	{ecuc(ComM/ComMGeneral/ComMDynamicPncToChannelMappingSupport)} == true		

]()

8.7.6 ModeDeclarationGroups

8.7.6.1 ComMMode

[SWS_ComM_01012]

Name	ComMMode	
Kind	ModeDeclarationGroup	
Category	ALPHABETIC_ORDER	
Initial mode	COMM_NO_COMMUNICATION	
On transition value	--	
Modes	COMM_FULL_COMMUNICATION	--
	COMM_NO_COMMUNICATION	--
	COMM_SILENT_COMMUNICATION	--
Description	--	

l()

9 Sequence diagrams

9.1 Transmission and Reception start (CAN)

Figure 13 shows the sequence for starting transmission and reception on CAN. The behaviour is equal for LIN, FlexRay and Ethernet just with different API names.

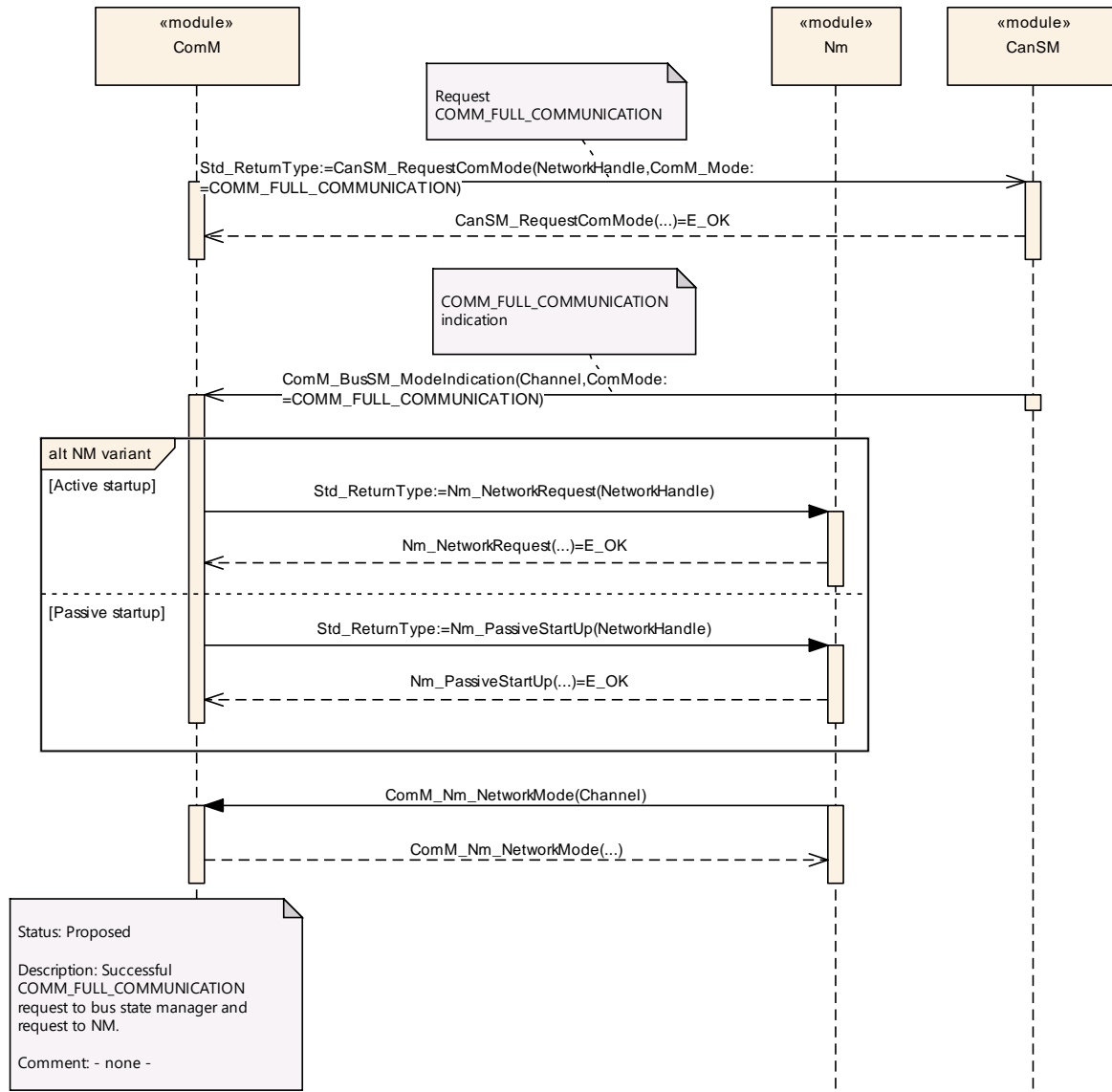


Figure 13: Starting transmission and reception on CAN

9.2 Passive Wake-up (CAN)

Figure 14 shows the behaviour after a wake-up indicated by the ECU State Manager module, or the Nm module for a CAN channel. The behaviour is equal for LIN, FlexRay and Ethernet just with different API names.

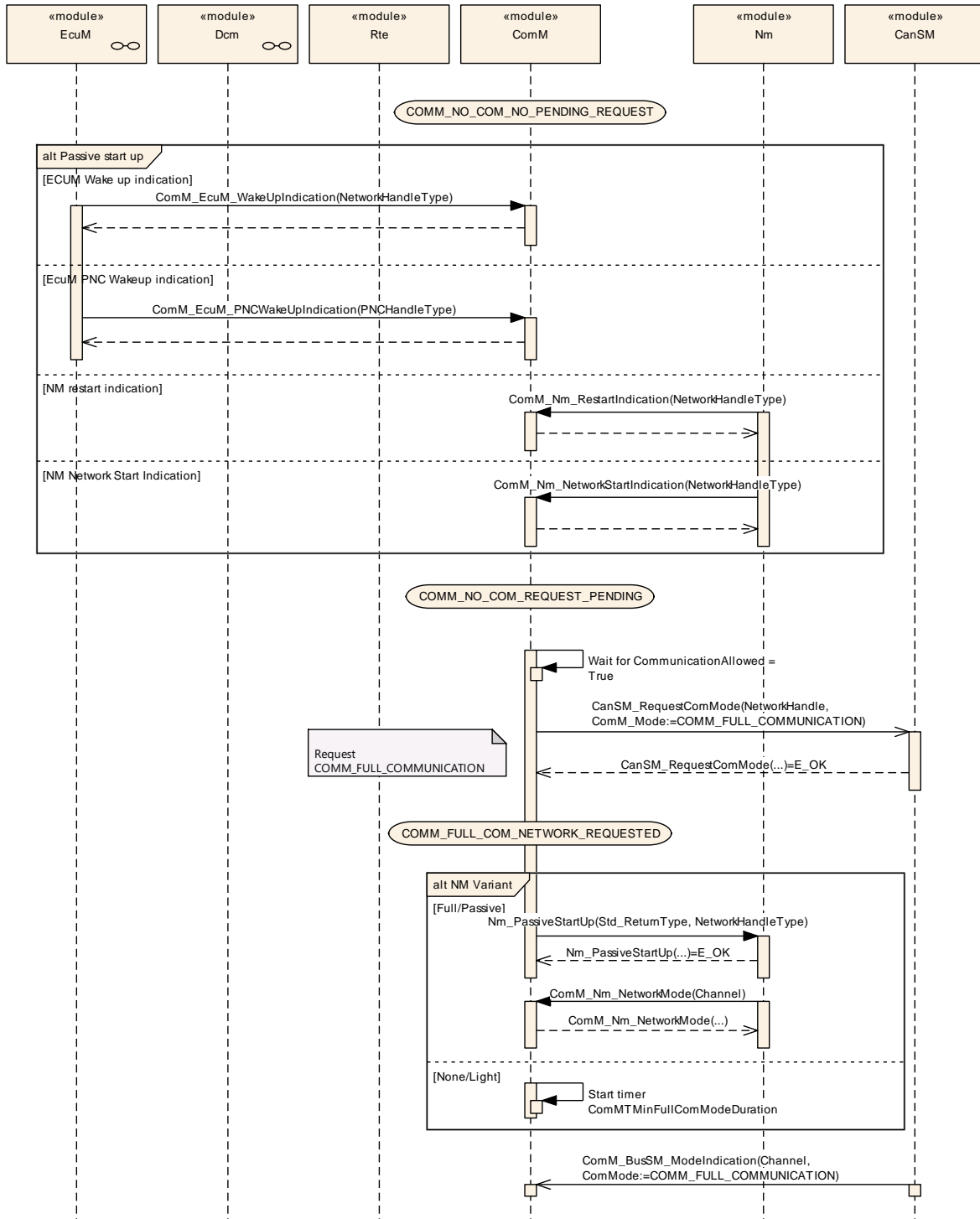


Figure 14: Reaction on a wake-up indicated by the ECU State Manager module

9.3 Network shutdown (CAN)

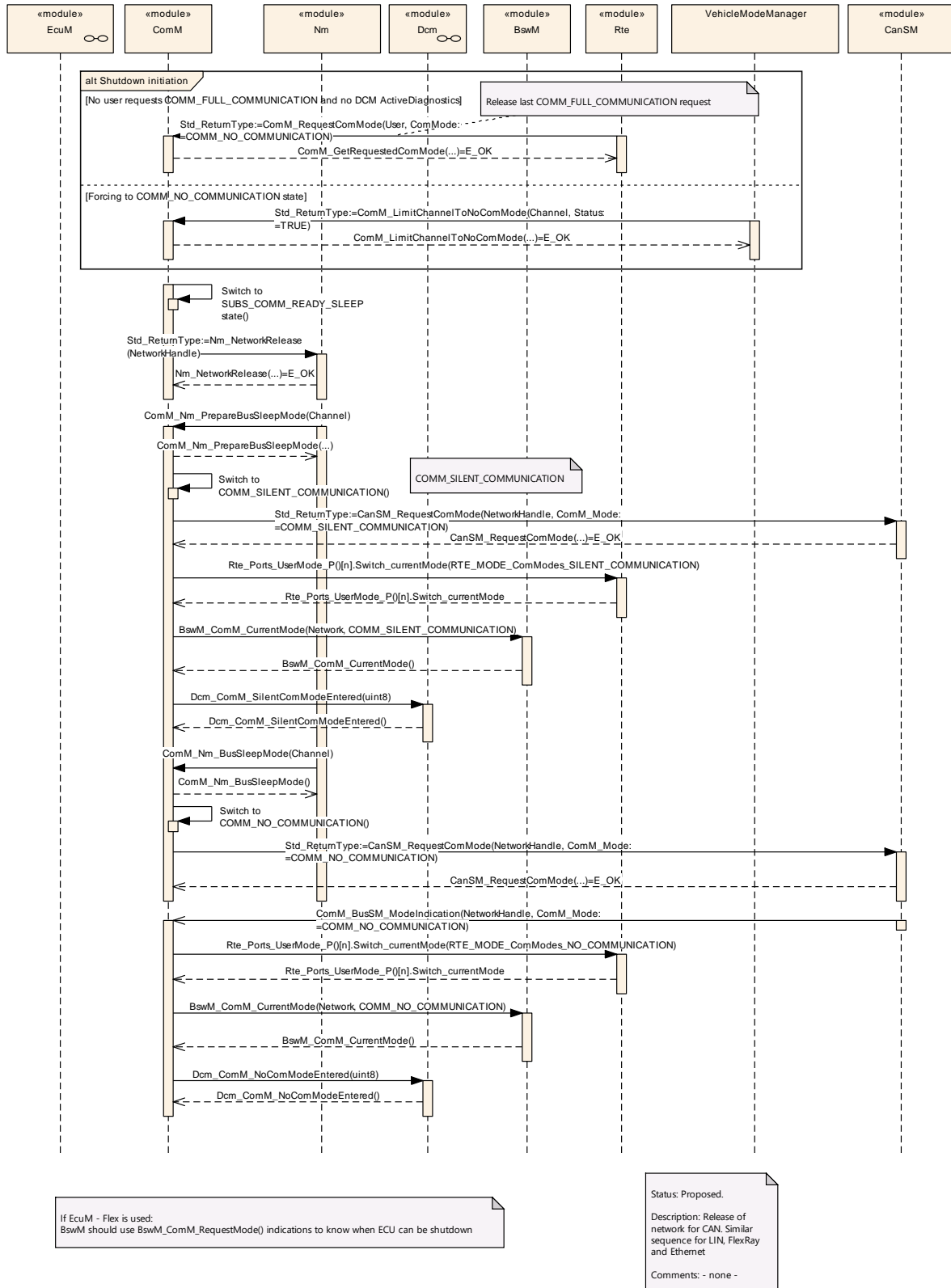


Figure 15 shows the possibilities to shutdown the CAN network. It can be either initiated if the last user releases his `COMM_FULL_COMMUNICATION` request or
156 of 192 Document ID 79: AUTOSAR_SWS_COMManager

ComM_LimitChannelToNoComMode(...) (see [SWS_ComM_00163](#)) is called. The behaviour is equal for LIN, FlexRay and Ethernet just with different API names.

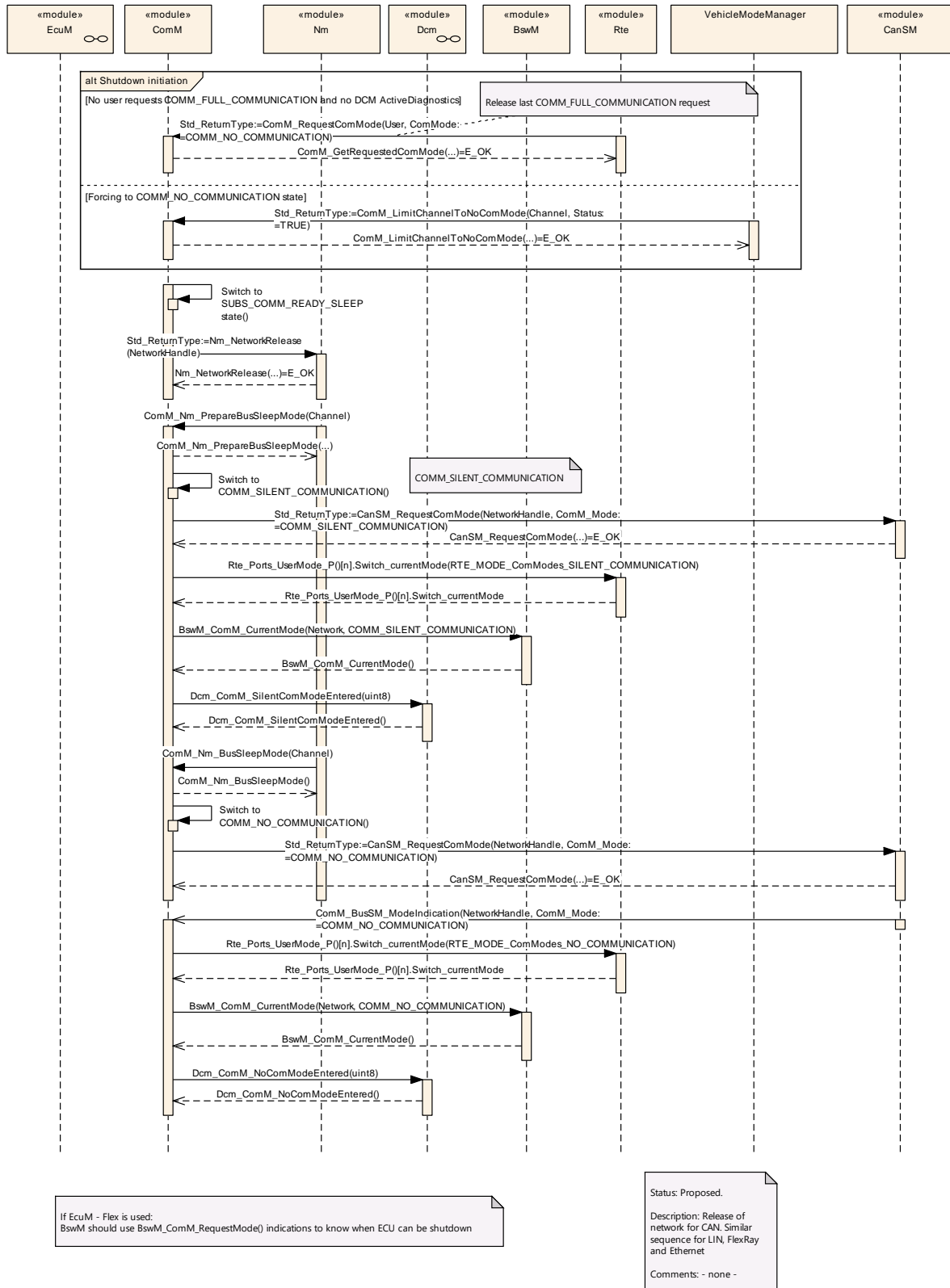


Figure 15: Network shutdown (CAN)

9.4 Communication request

Figure 16 shows the possibilities to start `COMM_FULL_COMMUNICATION` on CAN. It can be either initiated if a user requests `COMM_FULL_COMMUNICATION` request or DCM indicates `ComM_DCM_ActiveDiagnostic` (see [SWS ComM_00873](#)). The behaviour is equal for LIN, FlexRay and Ethernet just with different API names.

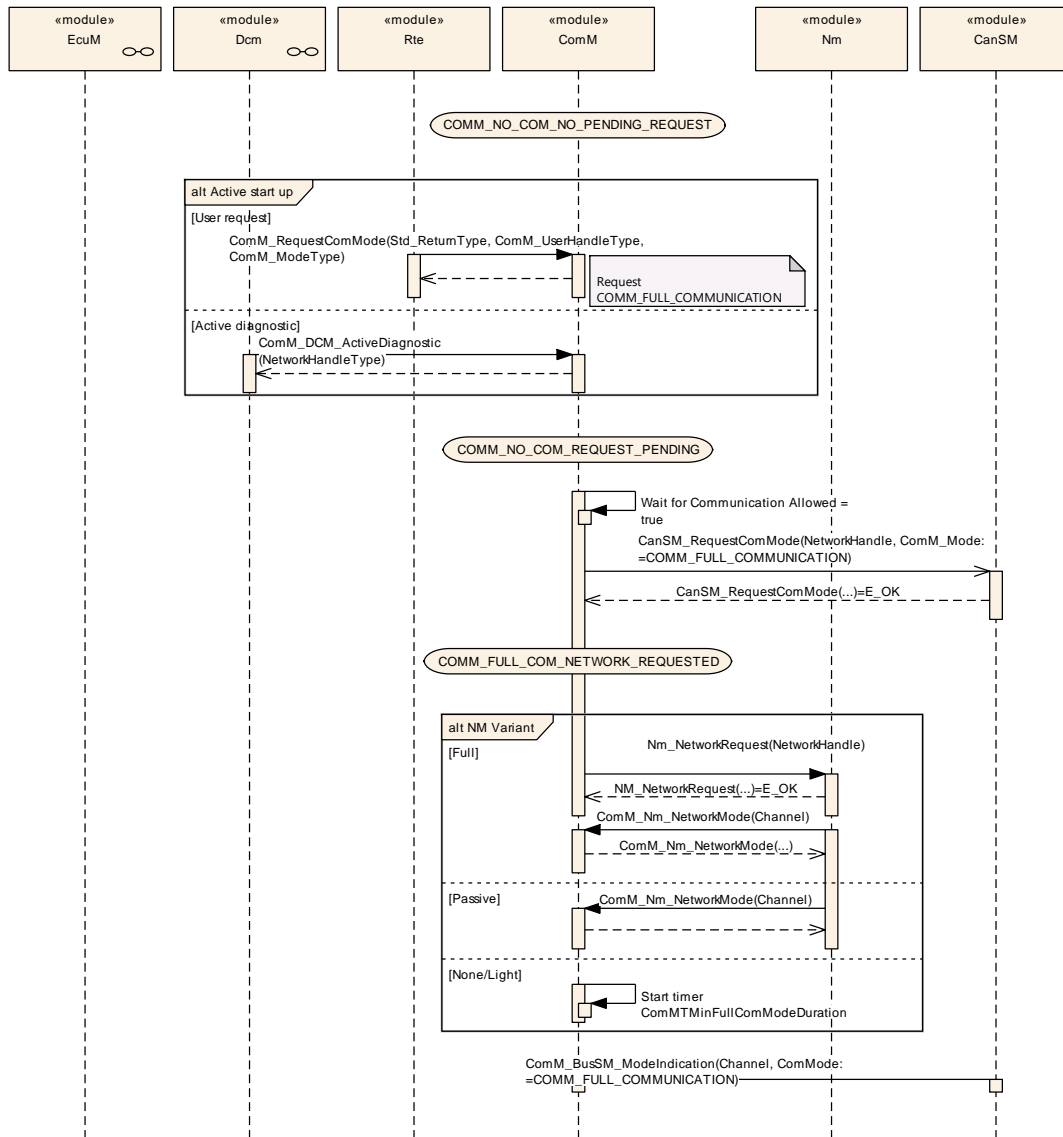


Figure 16: Request Communication

9.5 Synchronized PNC shutdown

Note: The sequence diagrams shows the expected behaviour, but not the implementation

Figure 17 shows the request for a synchronized PNC shutdown if an ECU in the role of a top-level PNC coordinator detects a release of a PNC.

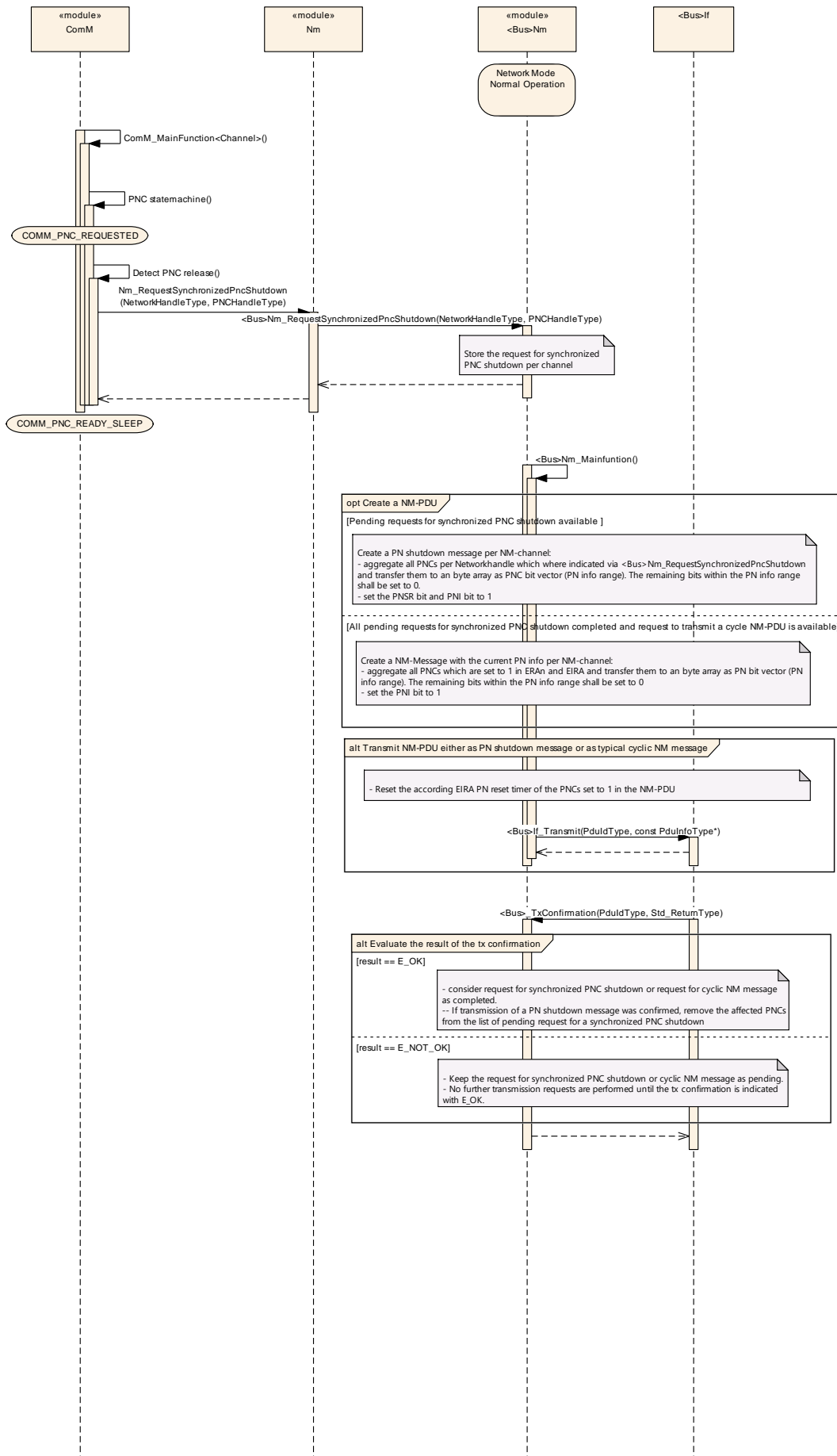


Figure 17: Request for a synchronized PNC shutdown in the role of a top-level PNC coordinator (TLPC)

Figure 18 shows the request to forward a received synchronized PNC shutdown if an ECU in role of an intermediate PNC coordinator receives a PN shutdown message.

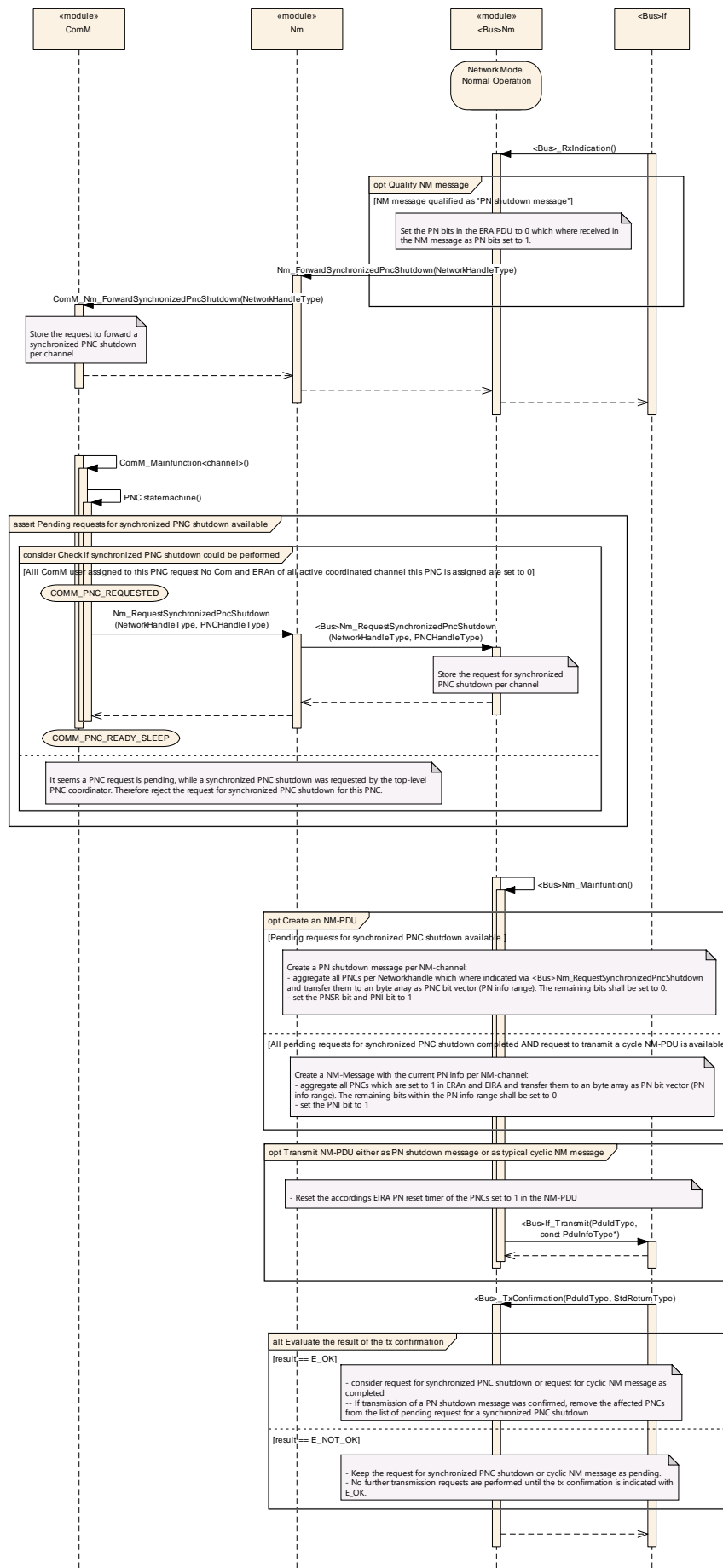


Figure 18: Request to forward a synchronized PNC shutdown in the role of an intermediate PNC coordinator

10 Configuration specification

In general, this chapter defines configuration parameters and their clustering into containers. In order to support the specification Chapter 10.1 describes fundamentals.

Chapter 10.2 specifies the structure (containers) and the parameters of the module Communication Manager Module.

Chapter 10.3 specifies published information of the Communication Manager Module.

10.1 How to read this chapter

For details refer to the chapter 10.1 “Introduction to configuration specification” in *SWS_BSWGeneral*.

10.2 Containers and configuration parameters

[SWS_ComM_00419] [The ComM module pre-compile time and link time configuration parameters shall be checked statically (at the latest during link time) for correctness.](SRS_BSW_00167)

[SWS_ComM_00322] [The ComM module configuration shall support configuration of bus type for each channel.](SRS_BSW_00167)

Rationale for [SWS ComM 00322](#): Interfaces for controlling the communication stack depends on the bus type.

[SWS_ComM_00464] [The ComM module shall strictly separate configuration from implementation.](SRS_BSW_00158)

Rationale for [SWS ComM 00464](#): Easy and clear configuration.

10.2.1 ComM

SWS Item	ECUC_ComM_00890 :
Module Name	ComM
Module Description	Configuration of the ComM (Communications Manager) module.
Post-Build Variant Support	true
Supported Config Variants	VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

Included Containers		
Container Name	Multiplicity	Scope / Dependency
ComMConfigSet	1	This container contains the configuration parameters and sub containers of the AUTOSAR ComM module.
ComMGeneral	1	General configuration parameters of the Communication Manager.

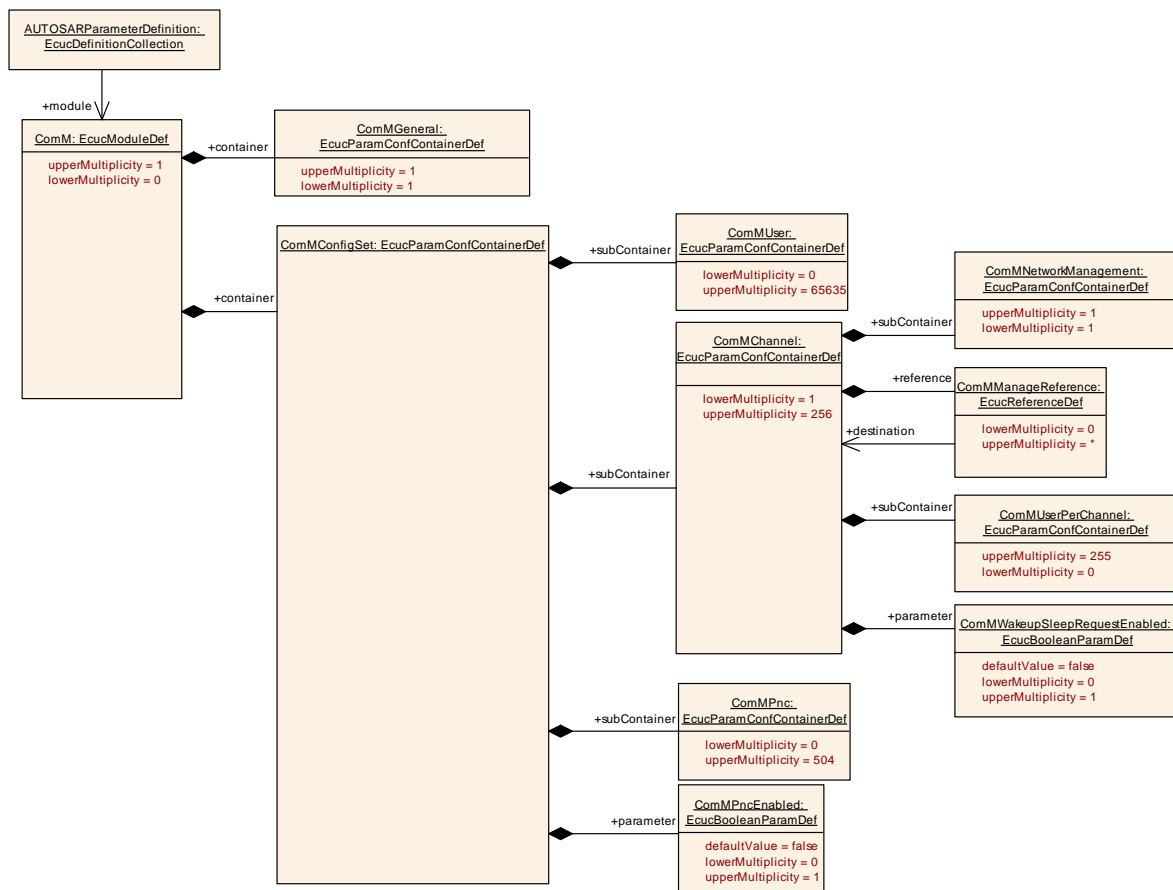


Figure 19: Configuration ComM

10.2.2 ComMGeneral

SWS Item	ECUC_ComM_00554 :
Container Name	ComMGeneral
Parent Container	ComM

Description	General configuration parameters of the Communication Manager.
Configuration Parameters	

SWS Item	ECUC_ComM_00892 :		
Name	ComM0PncVectorAvoidance		
Parent Container	ComMGeneral		
Description	This parameter avoids sending of 0-PNC-Vectors in case ComMPncGatewayEnabled is enabled.		
Multiplicity	0..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 dependency: ComMPncGatewayEnabled is enabled		

SWS Item	ECUC_ComM_00555 :		
Name	ComMDevErrorDetect		
Parent Container	ComMGeneral		
Description	Switches the development error detection and notification on or off. <ul style="list-style-type: none"> • 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_ComM_00840 :		
Name	ComMDirectUserMapping		
Parent Container	ComMGeneral		
Description	If this parameter is set to true the configuration tool shall automatically create a ComMUser per ComMPnc and a ComMUser per ComMChannel. The shortName of the generated ComMUsers shall follow the following naming convention: PNCUser_ComMPncId, e.g. PNCUser_13 ChannelUser_ComMChannelId, e.g. ChannelUser_25 Restriction: ComMUser, which are created due to this configuration parameter, shall not be used by SWCs (only available for BswM).		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	

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

SWS Item	ECUC_ComM_00895 :		
Name	ComMDynamicPncToChannelMappingSupport		
Parent Container	ComMGeneral		
Description	Precompile time switch to enable the dynamic PNC-to-channel-mapping handling. False: Dynamic PNC-to-channel-mapping is disabled True: Dynamic PNC-to-channel-mapping is enabled Tags: atp.Status=draft		
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: ECU dependency: shall only be TRUE if ComMPncSupport = TRUE		

SWS Item	ECUC_ComM_00563 :		
Name	ComMEcuGroupClassification		
Parent Container	ComMGeneral		
Description	Defines whether a mode inhibition affects the ECU or not. Examples: 000: No mode inhibition can be activated 001: Wake up inhibition can be enabled		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	3		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: Shall be stored non volatile (value must be kept during a reset) at least if Wake up inhibition is enabled/allowed. Can be changed during runtime with ComM_SetECUGroupClassification() thus the default values shall be set only once (first ECU initialization).		

SWS Item	ECUC_ComM_00560 :		
Name	ComMModeLimitationEnabled		
Parent Container	ComMGeneral		
Description	true if mode limitation functionality shall be enabled. true: Enabled false: 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_ComM_00887 :		
Name	ComMPncGatewayEnabled		
Parent Container	ComMGeneral		
Description	Enables or disables support of Partial Network Gateway. False: Partial Networking Gateway is disabled True: Partial Networking Gateway is enabled		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_ComM_00841 :		
Name	ComMPncPrepareSleepTimer		
Parent Container	ComMGeneral		
Description	Time in seconds the PNC state machine shall wait in COMM_PNC_PREPARE_SLEEP.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[0 .. 63]		
Default value	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: DRAFT: #CanNm: (NmPnResetTime + ComMPncPrepareSleepTimer) < CanNmTimeoutTime # FrNm: (NmPnResetTime + ComMPncPrepareSleepTimer) < ((FrNmReadySleepCnt +1) * FrNmRepetitionCycle * "Duration of one FlexRay Cycle") # UdpNm: (NmPnResetTime + ComMPncPrepareSleepTimer) < UdpNmTimeoutTime OBSOLETE:		

	#CanNm: (CanNmPnResetTime + ComMPncPrepareSleepTimer) < CanNmTimeoutTime # FrNm: (FrNmPnResetTime + ComMPncPrepareSleepTimer) < (FrNmReadySleepCnt + 1) * FrNmRepetitionCycle * "Duration of one FlexRay Cycle") # UdpNm: (UdpNmPnResetTime + ComMPncPrepareSleepTimer) < UdpNmTimeoutTime
--	---

SWS Item	ECUC_ComM_00839 :		
Name	ComMPncSupport		
Parent Container	ComMGeneral		
Description	Enables or disables support of partial networking. False: Partial Networking is disabled True: Partial Networking is enabled		
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_ComM_00558 :		
Name	ComMResetAfterForcingNoComm		
Parent Container	ComMGeneral		
Description	ComM shall perform a reset after entering "No Communication" mode because of an active mode limitation to "No Communication" mode. true: Enabled false: 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_ComM_00897 :		
Name	ComMSynchronizedPncShutdownEnabled		
Parent Container	ComMGeneral		
Description	Enables or disables support of synchronized PNC shutdown. FALSE: synchronized PNC shutdown is disabled TRUE: synchronized PNC shutdown is enabled NOTE: This is only possible for ECU that has the role of an top-level PNC coordinator or intermediate PNC within the PNC network Tags: atp.Status=draft		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		

Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: Parameter can only be set to TRUE if ComMPncGatewayEnabled is set to TRUE.		

SWS Item	ECUC_ComM_00695 :		
Name	ComMSynchronousWakeUp		
Parent Container	ComMGeneral		
Description	Wake up of one channel shall lead to a wake up of all channels if true. true: Enabled false: Disabled		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
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_ComM_00557 :		
Name	ComMTMinFullComModeDuration		
Parent Container	ComMGeneral		
Description	Minimum time duration in seconds, spent in the COMM_FULL_COMMUNICATION sub-state COMM_FULL_COM_NETWORK_REQUESTED.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0.001 .. 65]		
Default value	5		
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_ComM_00622 :		
Name	ComMVersionInfoApi		
Parent Container	ComMGeneral		
Description	Switches the possibility to read the version information with the service ComM_GetVersionInfo(). true: Enabled false: 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_ComM_00559 :		
Name	ComMWakeupInhibitionEnabled		
Parent Container	ComMGeneral		
Description	true if wake up inhibition functionality enabled. true: Enabled false: 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_ComM_00783 :		
Name	ComMGlobalNvMBlockDescriptor		
Parent Container	ComMGeneral		
Description	Reference to NVRAM block containing the none volatile data. If this parameter is not configured it means that no NVRam is used at all.		
Multiplicity	0..1		
Type	Symbolic name reference to [NvMBlockDescriptor]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: ECU dependency: Derived from NvM configuration		

No Included Containers

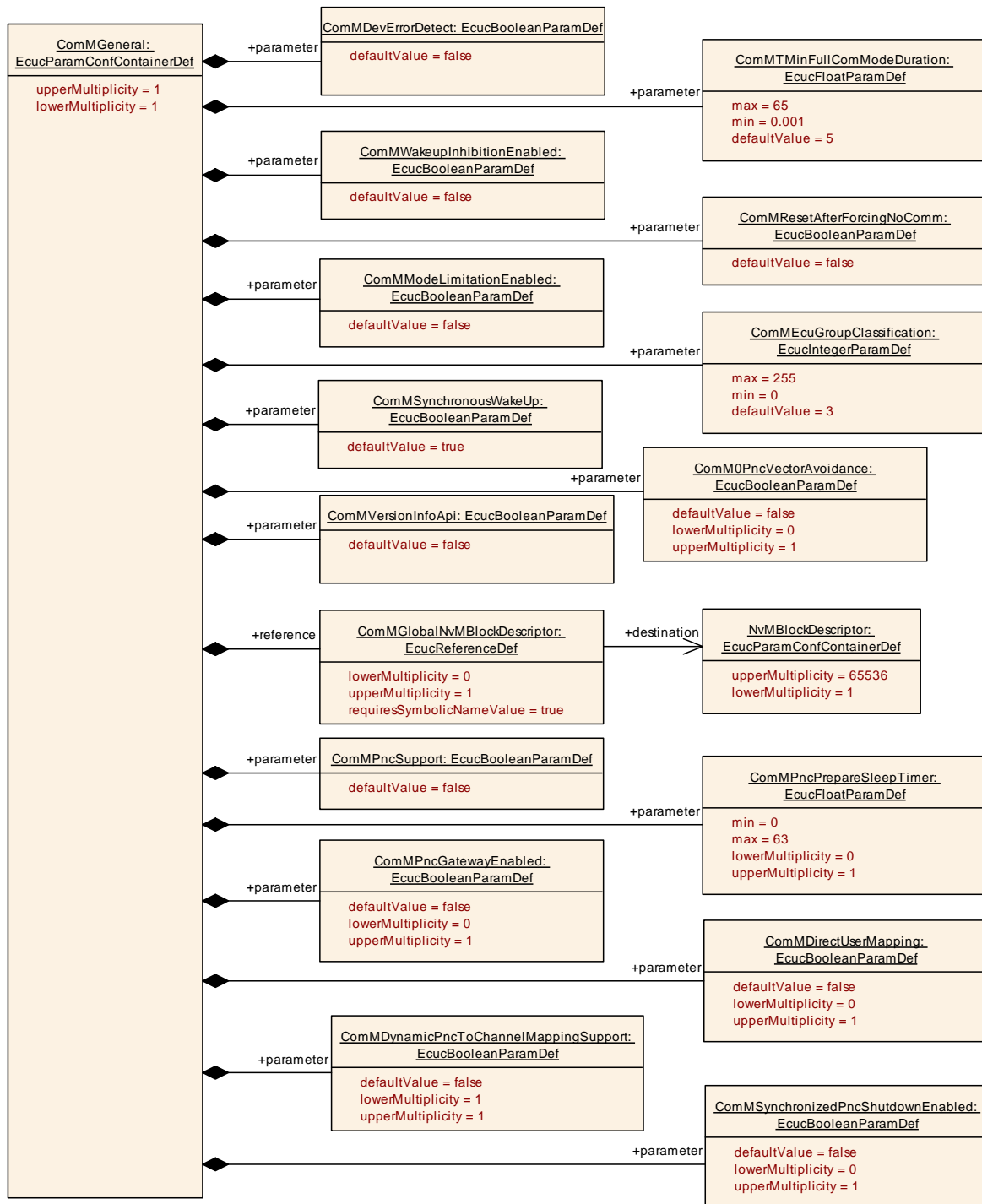


Figure 20: Configuration ComMGeneral

10.2.3 ComMConfigSet

SWS Item	ECUC_ComM_00879 :
Container Name	ComMConfigSet
Parent Container	ComM
Description	This container contains the configuration parameters and sub containers of

	the AUTOSAR ComM module.		
Configuration Parameters			
SWS Item	ECUC_ComM_00878 :		
Name	ComMPncEnabled		
Parent Container	ComMConfigSet		
Description	Defines whether in this configuration set the partial networking is enabled. true: Enabled false: Disabled		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU dependency: ComMPncSupport		

Included Containers		
Container Name	Multiplicity	Scope / Dependency
ComMChannel	1..256	This container contains the configuration (parameters) of the bus channel(s). The channel parameters shall be harmonized within the whole communication stack.
ComMPnc	0..504	This container contains the configuration of the partial network cluster (PNC).
ComMUser	0..65635	This container contains a list of identifiers that are needed to refer to a user in the system which is designated to request Communication modes.

10.2.4 ComMUser

SWS Item	ECUC_ComM_00653 :
Container Name	ComMUser
Parent Container	ComMConfigSet
Description	This container contains a list of identifiers that are needed to refer to a user in the system which is designated to request Communication modes.
Configuration Parameters	

SWS Item	ECUC_ComM_00654 :	
Name	ComMUserIdentifier	
Parent Container	ComMUser	
Description	An identifier that is needed to refer to a user in the system which is designated to request Communication Modes. ImplementationType: ComM_UserHandleType	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 254	

Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: EcuMUser: The concept of users is very similar to the concept of requestors in the ECU State Manager specification. These two parameters shall be harmonized during the configuration process.		

SWS Item	ECUC_ComM_00786 :		
Name	ComMUserEcucPartitionRef		
Parent Container	ComMUser		
Description	Denotes in which "EcucPartition" the requester is executed. When the partition is stopped, the communication request shall be cancelled in the ComM to avoid a stay-awake situation of the bus due to a stopped partition.		
Multiplicity	0..1		
Type	Reference to [EcucPartition]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: ECU		

No Included Containers

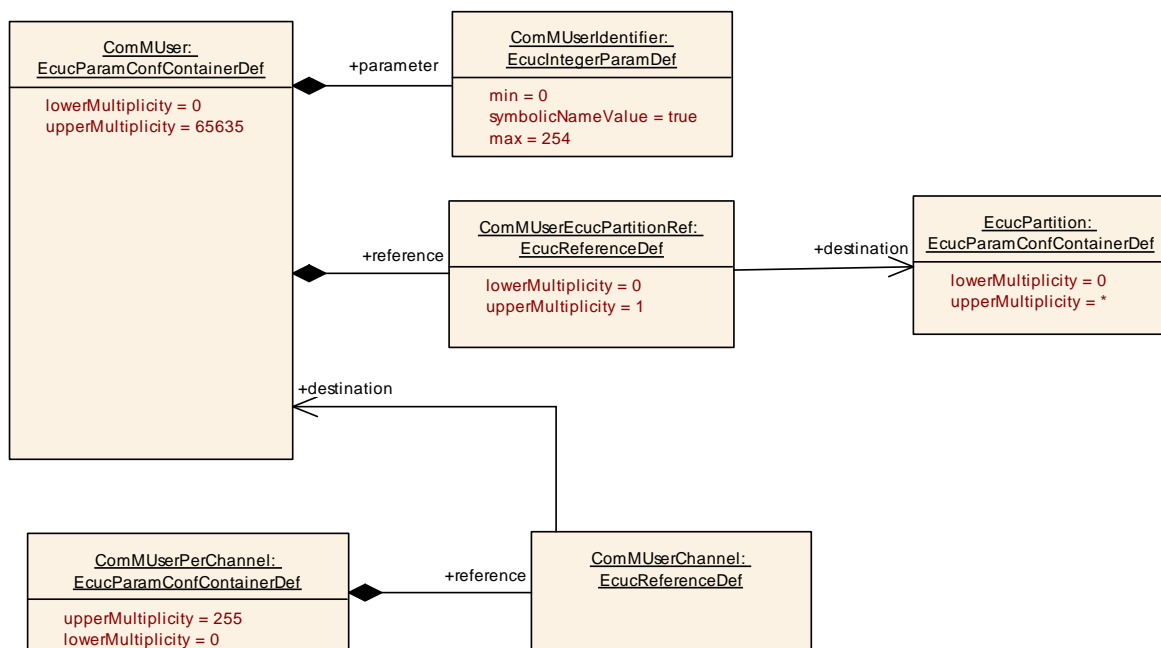


Figure 21: Configuration ComMUser

10.2.5 ComMChannel

SWS Item	ECUC_ComM_00565 :
Container Name	ComMChannel
Parent Container	ComMConfigSet
Description	This container contains the configuration (parameters) of the bus channel(s). The channel parameters shall be harmonized within the whole communication stack.
Configuration Parameters	

SWS Item	ECUC_ComM_00567 :		
Name	ComMBusType		
Parent Container	ComMChannel		
Description	Identifies the bus type of the channel.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	COMM_BUS_TYPE_CAN	--	
	COMM_BUS_TYPE_CDD	--	
	COMM_BUS_TYPE_ETH	--	
	COMM_BUS_TYPE_FR	--	
	COMM_BUS_TYPE_INTERNAL	--	
	COMM_BUS_TYPE_LIN	--	
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_ComM_00888 :		
Name	ComMCDDBusPrefix		
Parent Container	ComMChannel		
Description	Prefix to be used for API calls to CDD.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: Only applicable if ComMBusType equals		

	COMM_BUS_TYPE_CDD.
--	--------------------

SWS Item	ECUC_ComM_00635 :		
Name	ComMChannelId		
Parent Container	ComMChannel		
Description	Channel identification number of the corresponding channel.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
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 dependency: Shall be harmonized with channel IDs of networkmanagement and the bus interfaces.		

SWS Item	ECUC_ComM_00896 :		
Name	ComMDynamicPncToChannelMappingEnabled		
Parent Container	ComMChannel		
Description	Channel-specific parameter to enable the dynamic PNC-to-channel-mapping feature. False: Dynamic PNC-to-channel-mapping is disabled True: Dynamic PNC-to-channel-mapping is enabled Tags: atp.Status=draft		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-POST-BUILD
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local dependency: Shall only be TRUE if ComMDynamicPncToChannelMappingSupport is TRUE and ComMNmVariant is set to FULL for this ComMChannel.		

SWS Item	ECUC_ComM_00787 :		
Name	ComMFullCommRequestNotificationEnabled		
Parent Container	ComMChannel		
Description	Defines if the optional SenderReceiver Port of Interface ComM_CurrentChannelRequest will be provided for this channel. True means enabled. False means 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 dependency: Shall be stored none volatile (value must be kept during a reset).		

SWS Item	ECUC_ComM_00556 :		
Name	ComMMainFunctionPeriod		
Parent Container	ComMChannel		
Description	Specifies the period in seconds that the MainFunction has to be triggered with. Comment: ComM scheduling shall be at least as fast as the communication stack and a schedule longer than 100ms makes no sense for communication.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range]0 .. INF[
Default value	0.02		
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_ComM_00571 :		
Name	ComMNoCom		
Parent Container	ComMChannel		
Description	Not allowed to change state of ComM channel to COMM_SILENT_COMMUNICATION or COMM_FULL_COMMUNICATION. true: Enabled - Not allowed to switch to Communication Modes above. false: Disabled - Allowed to switch Communication Modes above. Shall be possible to change parameter during runtime with ComM API's. ECU/All channels: ComM_LimitECUToNoComMode(). Separate channels: ComM_LimitChannelToNoComMode().		
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 dependency: ComMModeLimitationEnabled		

SWS Item	ECUC_ComM_00569 :		
Name	ComMNoWakeup		
Parent Container	ComMChannel		
Description	Defines if an ECU is not allowed to wake-up the channel. true: Enabled (not allowed to wake-up)) false: Disabled This is the default/init value of a runtime variable that can be changed during runtime using ComM_PreventWakeup().		
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 dependency: Shall be stored none volatile (value must be kept during a reset).		

SWS Item	ECUC_ComM_00789 :		
Name	ComMNoWakeUpInhibitionNvmStorage		
Parent Container	ComMChannel		
Description	If this parameter is set to "true", the NoWakeUp inhibition state of the channel shall be stored (in some implementation specific way) in the block pointed to by ComMGlobalNvmBlockDescriptor.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: If the parameter is set to true, a valid Nvm block reference must be given in the (existing, i.e. multiplicity 1) ComMGlobalNvmBlockDescriptor pointing to a sufficiently big Nvm block.		

SWS Item	ECUC_ComM_00842 :		
Name	ComMPncGatewayType		
Parent Container	ComMChannel		
Description	Identifies the Partial Network Gateway behaviour of a ComMChannel.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	COMM_GATEWAY_TYPE_ACTIVE	--	
	COMM_GATEWAY_TYPE_PASSIVE	--	
Default value	COMM_GATEWAY_TYPE_ACTIVE		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: ECU dependency: Parameter shall not be used for managed channel (shall neither be set to COMM_GATEWAY_TYPE_ACTIVE nor COMM_GATEWAY_TYPE_PASSIVE).		

SWS Item	ECUC_ComM_00898 :		
Name	ComMWakeupSleepRequestEnabled		
Parent Container	ComMChannel		
Description	Used for communication channels where the corresponding hardware support wake-up and/or sleep request capability on the network, e.g. OA TC10 compatible PHYs for Ethernet.		

Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: Only applicable if ComMBusType equals COMM_BUS_TYPE_ETH and the used Ethernet hardware (e.g. PHY, Ethernet switch) is compatible with the OA TC10 specification.		

SWS Item	ECUC_ComM_00894 :		
Name	ComMChannelPartitionRef		
Parent Container	ComMChannel		
Description	Reference to EcucPartition, where the according ComMChannel is assigned to.		
Multiplicity	0..1		
Type	Reference to [EcucPartition]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_ComM_00893 :		
Name	ComMManageReference		
Parent Container	ComMChannel		
Description	Represents the reference between a ComMChannel with role managing channel and a ComMChannel with role managed channel.		
Multiplicity	0..*		
Type	Reference to [ComMChannel]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

Included Containers		
Container Name	Multiplicity	Scope / Dependency
ComMNetworkManagement	1	This container contains the configuration parameters of the

		networkmanagement.
ComMUserPerChannel	0..255	This container contains a list of identifiers that are needed to refer to a user in the system which is linked to a channel.

[SWS_ComM_00690] [Configuration parameter `ComMNoCom` (see [ECUC_ComM_00571](#)) need not to be evaluated in case `ComMModeLimitationEnabled` = FALSE = Disabled (see [ECUC_ComM_00560](#)) thus it can be removed in that case to reduce/optimize the configuration.](
(

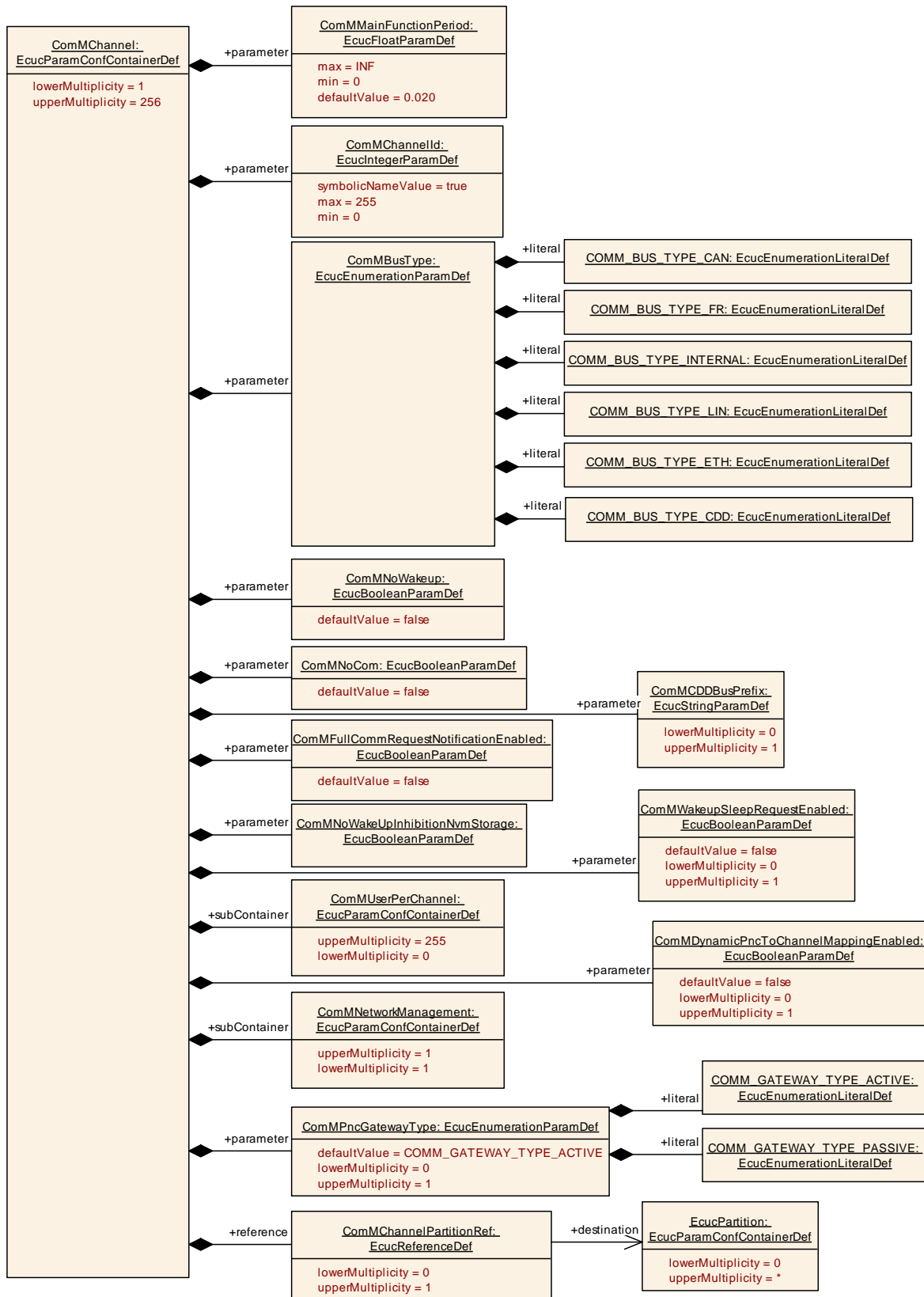


Figure 22: Configuration ComMChannel

10.2.6 ComMNetworkManagement

SWS Item	ECUC_ComM_00607 :
Container Name	ComMNetworkManagement
Parent Container	ComMChannel
Description	This container contains the configuration parameters of the networkmanagement.
Configuration Parameters	

SWS Item	ECUC_ComM_00606 :		
Name	ComMNMLightTimeout		
Parent Container	ComMNetworkManagement		
Description	Defines the timeout (in seconds) after COMM_FULL_COMMUNICATION sub-state COMM_FULL_COM_READY_SLEEP is left. The range shall be greater than 0.0 and less or equal to 255.0.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[0 .. 255]		
Default value	10		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: Only used if ComMNMVariant is configured as ComMLight		

SWS Item	ECUC_ComM_00568 :		
Name	ComMNMVariant		
Parent Container	ComMNetworkManagement		
Description	Defines the functionality of the networkmanagement. Shall be harmonized with NM configuration.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	FULL		AUTOSAR NM is available (default).
	LIGHT		No AUTOSAR NM is available, but functionality to shut down a channel.
	NONE		No NM available
	PASSIVE		AUTOSAR NM running in passive mode available.
	SLAVE_ACTIVE		No NM is available. This is used for e.g. LIN slaves.
	SLAVE_PASSIVE		No NM is available. This used for e.g. Ethernet communication channels with OA TC10 compliant hardware.
Default value	FULL		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope /	scope: local		

Dependency	dependency: ComMNmVariant shall be NONE if ComMbusType = COMM_BUS_TYPE_INTERNAL. ComMNmVariant shall be LIGHT for managed channels. ComMNmVariant shall be FULL for managing channels.
-------------------	--

SWS Item	ECUC ComM 00886 :		
Name	ComMPncNmRequest		
Parent Container	ComMNetworkManagement		
Description	If this parameter equals true then every time a FULL Communication is requested due to a change in the PNC state machine to COMM_PNC_REQUESTED Nm shall be called using the API Nm_NetworkRequest.		
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 dependency: It shall only be possible to set ComMPncNmRequest to TRUE, if ComMNmVariant is FULL.		

No Included Containers

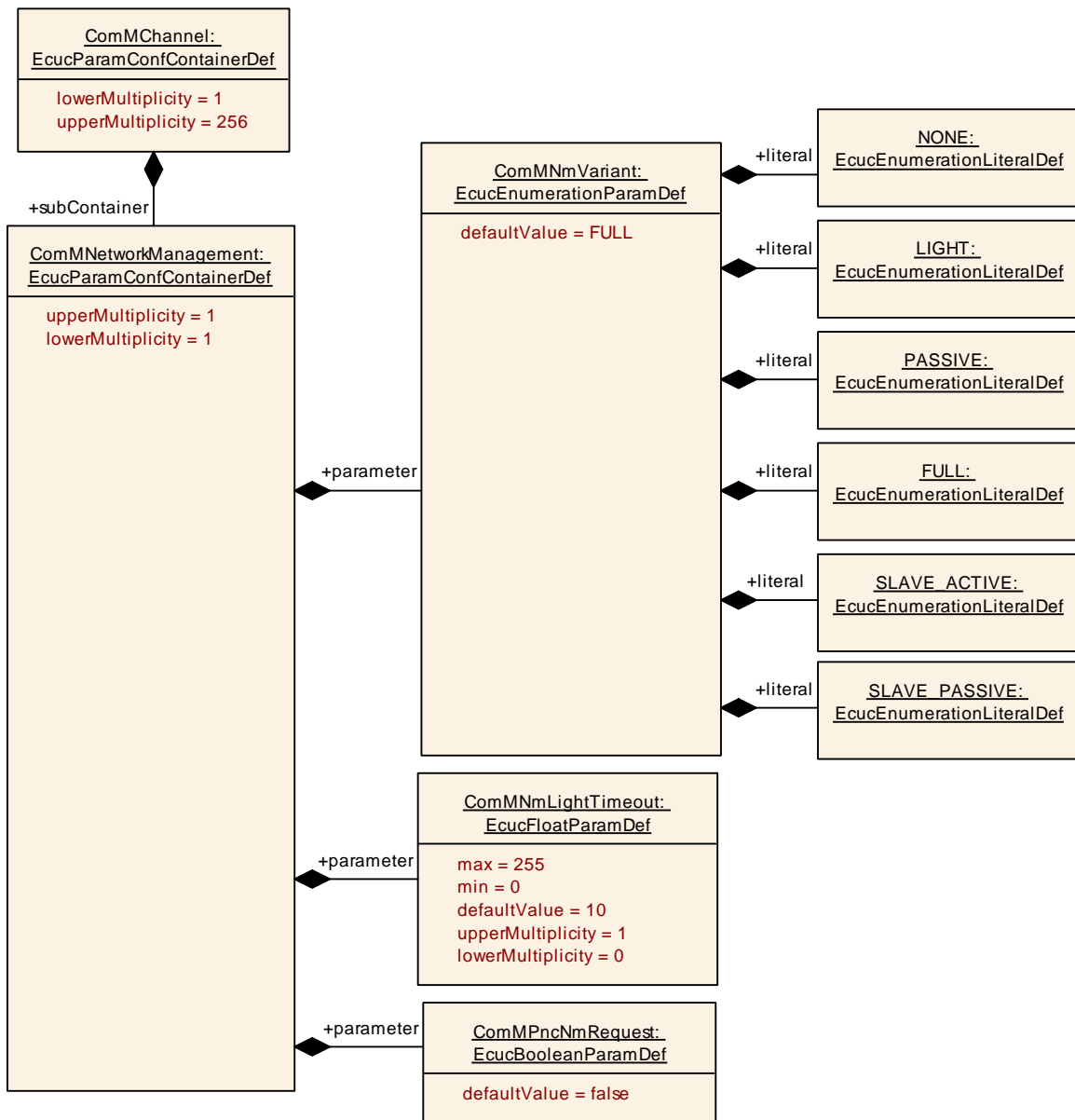


Figure 23: Configuration ComMNetworkManagement

10.2.7 ComMUserPerChannel

SWS Item	ECUC_ComM_00657 :
Container Name	ComMUserPerChannel
Parent Container	ComMChannel
Description	This container contains a list of identifiers that are needed to refer to a user in the system which is linked to a channel.
Configuration Parameters	

SWS Item	ECUC_ComM_00658 :
Name	ComMUserChannel
Parent Container	ComMUserPerChannel
Description	Reference to the ComMUser that corresponds to this channel user. ImplementationType: COMM_UserHandleType

Multiplicity	1		
Type	Reference to [ComMUser]		
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

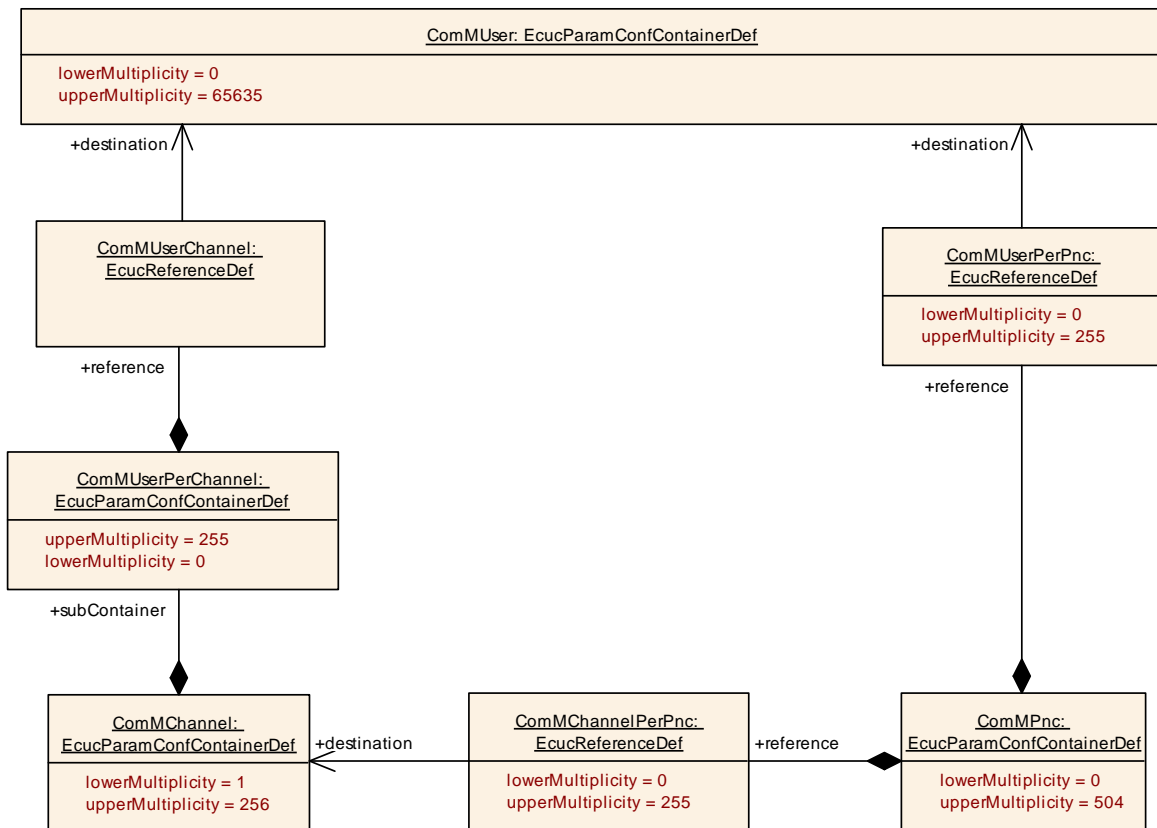


Figure 24: Configuration ComMUserPerChannel and ComUserPerPNC

10.2.8 ComMPnc

SWS Item	ECUC_ComM_00843 :
Container Name	ComMPnc
Parent Container	ComMConfigSet
Description	This container contains the configuration of the partial network cluster (PNC).
Configuration Parameters	

SWS Item	ECUC_ComM_00874 :
Name	ComMPncId
Parent Container	ComMPnc
Description	Partial network cluster identification number.
Multiplicity	1
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)

Range	8 .. 511		
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_ComM_00899 :		
Name	ComMPncWakeupSleepRequestEnabled		
Parent Container	ComMPnc		
Description	Used for PNCs where a requested PNC shall report an active communication request towards the BswM. The BswM forward the active communication request to the lower layer communication channels where the used hardware support wake-up and/or sleep request capability on the network, e.g. OA TC10 compatible PHYs for Ethernet. This is used e.g. for Ethernet Switch port group switching.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	ECUC_ComM_00880 :		
Name	ComMChannelPerPnc		
Parent Container	ComMPnc		
Description	Reference to the ComMChannel that is required for this PNC. ImplementationType: NetworkHandleType		
Multiplicity	0..255		
Type	Reference to [ComMChannel]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC_ComM_00900 :		
Name	ComMChannelPerTxOnlyPnc		
Parent Container	ComMPnc		
Description	Reference to the ComMChannel that is required for this PNC. This PNC is considered to be only transmitted on this channel as internal PNC request. ImplementationType: NetworkHandleType Tags:		

	atp.Status=draft		
Multiplicity	0..255		
Type	Reference to [ComMChannel]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	ECUC ComM 00891 :		
Name	ComMPncEthIfSwitchPortGroupRef		
Parent Container	ComMPnc		
Description	Reference to the PortGroups that correspond to this PNC. Note: This is only for documentation.		
Multiplicity	0..255		
Type	Symbolic name reference to [EthIfSwitchPortGroup]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-POST-BUILD
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	ECUC ComM 00876 :		
Name	ComMUserPerPnc		
Parent Container	ComMPnc		
Description	Reference to the ComMUsers that correspond to this PNC. ImplementationType: COMM_UserHandleType		
Multiplicity	0..255		
Type	Reference to [ComMUser]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

Included Containers		
Container Name	Multiplicity	Scope / Dependency
ComMPncComSignal	0..*	Represents the PncComSignals which are used to communicate the EIRA and ERA status of this PNC. Tags: atp.Status=obsolete

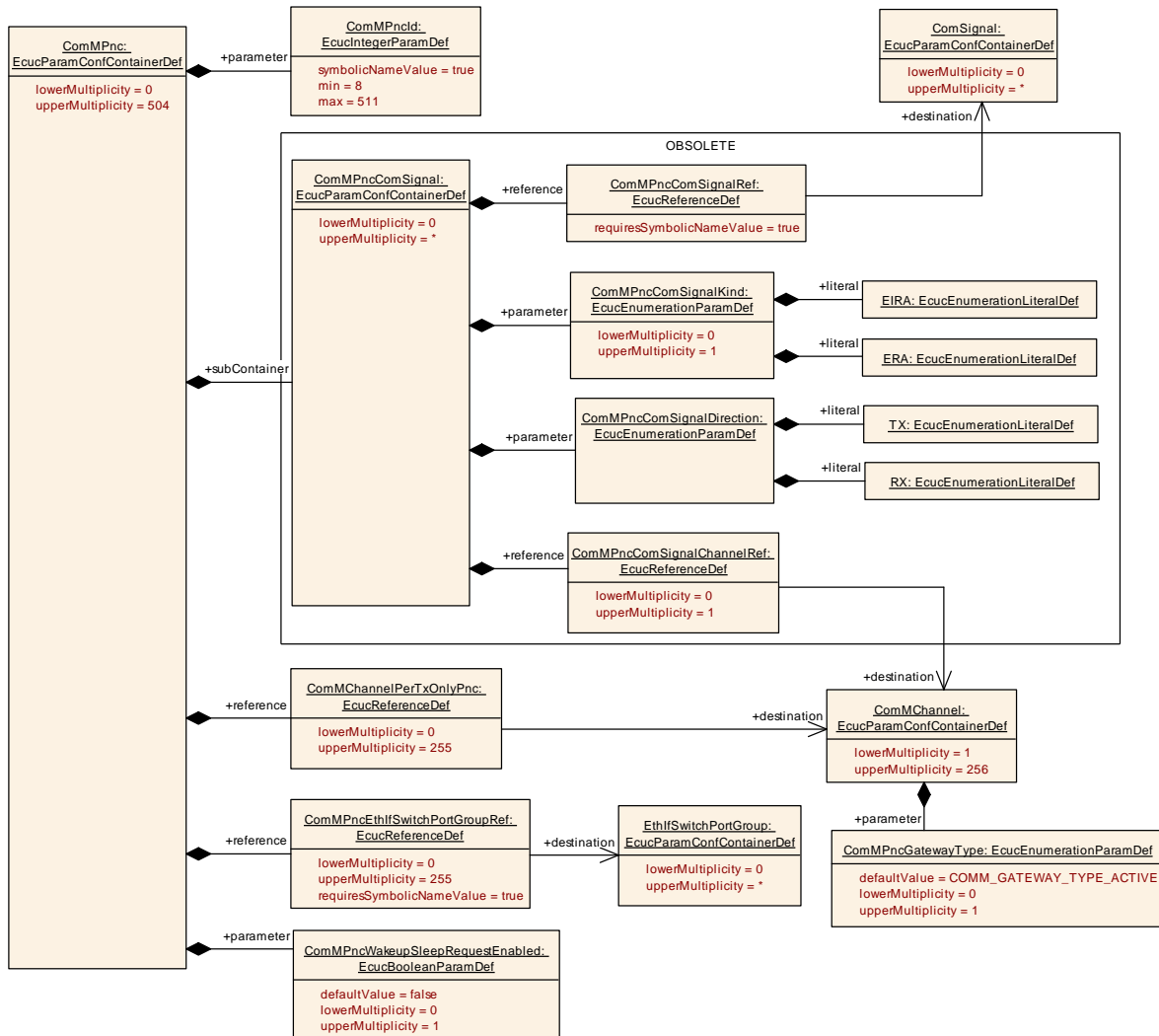


Figure 25: Configuration ComMPnc

10.2.9 ComMPncComSignal

SWS Item	ECUC_ComM_00881 : (Obsolete)
Container Name	ComMPncComSignal
Parent Container	ComMPnc
Description	Represents the PncComSignals which are used to communicate the EIRA and ERA status of this PNC. Tags: atp.Status=obsolete
Configuration Parameters	

SWS Item	ECUC_ComM_00885 : (Obsolete)
Name	ComMPncComSignalDirection
Parent Container	ComMPncComSignal
Description	Indicates the communication direction of this PncComSignal. Tags: atp.Status=obsolete
Multiplicity	1

Type	EcucEnumerationParamDef		
Range	RX	--	Tags: atp.Status=obsolete
	TX	--	Tags: atp.Status=obsolete
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_ComM_00883 : (Obsolete)		
Name	ComMPncComSignalKind		
Parent Container	ComMPncComSignal		
Description	Indicates whether this PncComSignal represents EIRA or ERA PNC information. This parameter ComMPncComSignalKind is optional and shall be ignored when ComMPncComSignalDirection equals TX. Tags: atp.Status=obsolete		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	EIRA	--	Tags: atp.Status=obsolete
	ERA	--	Tags: atp.Status=obsolete
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter ComMPncComSignalKind shall be ignored when ComMPncComSignalDirection equals TX.		

SWS Item	ECUC_ComM_00884 : (Obsolete)		
Name	ComMPncComSignalChannelRef		
Parent Container	ComMPncComSignal		
Description	Reference to the ComMChannel which is used to determine whether this PncComSignal shall participate in the active or passive role (via the parameter ComMPncGatewayType of the ComMChannel). Tags: atp.Status=obsolete		
Multiplicity	0..1		
Type	Reference to [ComMChannel]		
Post-Build Variant Multiplicity	false		

Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: ComMPncGatewayEnabled		

SWS Item	ECUC_ComM_00882 : (Obsolete)		
Name	ComMPncComSignalRef		
Parent Container	ComMPncComSignal		
Description	Reference to the ComSignal which is used to transport the partial network channel request information. Tags: atp.Status=obsolete		
Multiplicity	1		
Type	Symbolic name reference to [ComSignal]		
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

10.3 Published information

[SWS_ComM_00418] [The version information in the module header and source files shall be validated and consistent (e.g. by comparing the version information in the module header and source files with a pre-processor macro).](SRS_BSW_00004)

11 Not applicable requirements

[SWS_ComM_00499] [These requirements are not applicable to this specification.]
(SRS_BSW_00005, SRS_BSW_00009, SRS_BSW_00010, SRS_BSW_00161,
SRS_BSW_00162, SRS_BSW_00164, SRS_BSW_00168, SRS_BSW_00170,
SRS_BSW_00314, SRS_BSW_00325, SRS_BSW_00341, SRS_BSW_00343,
SRS_BSW_00344, SRS_BSW_00353, SRS_BSW_00361, SRS_BSW_00375,
SRS_BSW_00378, SRS_BSW_00398, SRS_BSW_00404, SRS_BSW_00405,
SRS_BSW_00413, SRS_BSW_00416, SRS_BSW_00417, SRS_BSW_00422,
SRS_BSW_00423, SRS_BSW_00424, SRS_BSW_00425, SRS_BSW_00426,
SRS_BSW_00427, SRS_BSW_00428, SRS_BSW_00429, SRS_BSW_00432,
SRS_BSW_00433, SRS_BSW_00437, SRS_BSW_00438, SRS_BSW_00439)