Document Title	Specification of Chinese Vehicle-2-X Network
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	991

Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R22-11

Document Change History			
Date	**Release**	**Changed by**	**Description**
2022-11-24	R22-11	AUTOSAR Release Management	• Initial release

Disclaimer

Contents

Known Limitations

1 Introduction and functional overview

This document specifies the functionality, APIs and the configuration of the AUTOSAR Basic Software module Chinese Vehicle-2-X Network (CnV2xNet).

The Chinese Vehicle-2-X Network (CnV2xNet) together with the Chinese Vehicle-2-X Message (CnV2xMsg), Chinese Vehicle-2-X Management (CnV2xMgt), Chinese Vehicle-2-X Security (CnV2xSec) and AUTOSAR BSW module Ethernet Interface (EthIf) forms the Chinese V2X stack within the AUTOSAR architecture.

The bases for this document are the Chinese LTE-V2X based standards [1] [2]. It is assumed that the reader is familiar with these standards.

1.1 Architecture Overview

CnV2xNet module provides services to upper module CnV2xMsg to transmit or receive V2X messages (i.e. DSMP SDUs), and gets services from the EthIf module to realize the data exchanging with LTE-V2X hardware. It also responsible for source Layer-2 address and destination layer-2 address selection and maintain QoS related mapping relationships between upper layer and lower layer from protocol perspective, the details are explained in chapter 1.2 and chapter 7 of this document.

Positioning of the CnV2xNet module within the AUTOSAR BSW and the Layered Software architecture is shown in below.

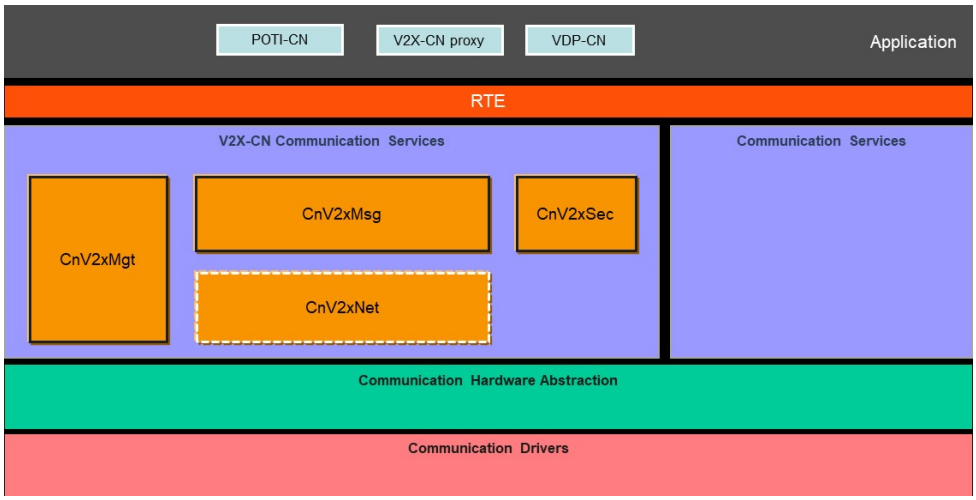

Figure 1.1: AUTOSAR BSW software architecture - CnV2xNet scope

1.2 Functional Overview

The functionality of CnV2xNet module should comply with the standard of Chinese LTE-V2X based Network protocol [2] and Technical Requirements of Vehicular Com-

munication System based on LTE-V2X Direct Communication [1]. The module provides services to the upper CnV2xMsg module specified in [3] for data transmission and reception. In order to communicate with LTE-V2X hardware to realize packet transport services, it relies on the lower EthIf module [4].

From protocol perspective, CnV2xNet module includes DSMP sub-layer and Adaptation sub-layer. Adaptation sub-layer provides parameter adaptation function between LTE-V2X access layer and DSMP sub-layer. DSMP sub-layer is responsible for exchanging data with message layer and Adaptation layer.

CnV2xNet module function also includes: mapping between application layer identifier and destination layer-2 ID, generation/change/maintenance of the source layer-2 ID, mapping between the unicast/multicast address and the layer-2 ID, mapping between the message priority and PPPP, indicating the service period to the lower layer, indicating the channel busy rate or maximum data rate to the upper layer, and so on.

2 Acronyms and Abbreviations

Abbreviation / Acronym:	Description:
API	Application programming Interface
BSW	Basic Software
BSM	Basic safety Message
C-V2X	Cellular based Vehicle to Everything
CCSA	China Communications Standards Association
CnV2xMsg	Chinese Vehicle-2-X Message
CnV2xNet	Chinese Vehicle-2-X Network
CnV2xSec	Chinese Vehicle-2-X Security
DEM	Diagnostic Event Manager
DET	Default Error Tracer
DSMP	Dedicated Short Message Protocol
EcuM	Electronic Control Unit Manager
IF	Interface
NTCAS	National Technical Committee of Auto Standardization
PPPP	ProSe Per-Packet Priority

3 Related documentation

3.1 Input documents & related standards and norms

[1] GB/T: Technical requirements and test methods of vehicular communication system based on LTE-V2X direct communication (Draft Edition: 2022-04-01)
http://www.catarc.org.cn/

[2] YD/T 3707-2020: Technical requirements of network layer of LTE-based vehicular communication
http://www.ccsa.org.cn/

[3] Specification of Chinese Vehicle-2-X Message
AUTOSAR_SWS_ChineseV2XMessage

[4] Specification of Ethernet Interface
AUTOSAR_SWS_EthernetInterface

[5] Specification of Default Error Tracer
AUTOSAR_SWS_DefaultErrorTracer

[6] Specification of ECU State Manager
AUTOSAR_SWS_ECUStateManager

4 Constraints and assumptions

4.1 Limitations

Wireless communication supports LTE-V2X only. Other cellular based wireless communication can be extended in future release of AUTOSAR standard. CnV2xNet module support non-IP (i.e. DSMP) transmission only and mainly focus on broadcast based packet transport services in R20-11. The V2X modules follow the guidance regarding the Day-1 V2X allocations defined in [1] [2], which are by NTCAS and CCSA respectively.

4.2 Applicability to car domains

This specification is applicable to all car domains.

5 Dependencies to other modules

This section describes the relations of CnV2xNet module to other modules within the AUTOSAR basic software architecture. It outlines the modules that are required or optional for the realization of CnV2xNet module and services.

5.1 AUTOSAR Default Error Tracer (DET)

In development mode, CnV2xNet module reports errors through the Det_ReportError function of DET Module [5].

5.2 AUTOSAR Ecu State Manager (EcuM)

The EcuM [6] initializes the CnV2xNet module by calling CnV2xNet_Init specified in 8.3.1 in this document.

5.3 AUTOSAR Ethernet Interface (EthIf)

The Ethernet Interface [4] is the lower layer module of CnV2xNet module.

5.4 AUTOSAR Chinese Vehicle-2-X Message (CnV2xMsg)

CnV2xMsg is the upper layer module of CnV2xNet module. The callback services called by CnV2xNet module are declared and placed inside the CnV2xMsg module. These callbacks provide receive indication and transmit confirmation services for the CnV2xMsg Module.

6 Requirements Tracing

Requirement	Description	Satisfied by
[CP_SRS_CnV2X_-00401]	The network layer of Chinese V2X communication shall support a CCSA compliant Network layer protocol of LTE-based vehicular communication	[CP_SWS_CnV2xNet_00100] [CP_SWS_CnV2xNet_00101] [CP_SWS_CnV2xNet_00102] [CP_SWS_CnV2xNet_00103] [CP_SWS_CnV2xNet_00104] [CP_SWS_CnV2xNet_00105] [CP_SWS_CnV2xNet_00106] [CP_SWS_CnV2xNet_00109] [CP_SWS_CnV2xNet_00110] [CP_SWS_CnV2xNet_00111] [CP_SWS_CnV2xNet_00113] [CP_SWS_CnV2xNet_00114] [CP_SWS_CnV2xNet_00115] [CP_SWS_CnV2xNet_00116] [CP_SWS_CnV2xNet_01002] [CP_SWS_CnV2xNet_01003] [CP_SWS_CnV2xNet_01004] [CP_SWS_CnV2xNet_01006] [CP_SWS_CnV2xNet_01007] [CP_SWS_CnV2xNet_01008] [CP_SWS_CnV2xNet_01009] [CP_SWS_CnV2xNet_01010] [CP_SWS_CnV2xNet_01011] [CP_SWS_CnV2xNet_01012] [CP_SWS_CnV2xNet_02001] [CP_SWS_CnV2xNet_02003] [CP_SWS_CnV2xNet_02005] [CP_SWS_CnV2xNet_02009] [CP_SWS_CnV2xNet_02012] [CP_SWS_CnV2xNet_02015] [CP_SWS_CnV2xNet_02018] [CP_SWS_CnV2xNet_02019] [CP_SWS_CnV2xNet_02020]
[CP_SRS_CnV2X_-00402]	The network layer of Chinese V2X communication shall Select and maintain Source Layer-2 ID and Destination Layer-2 ID	[CP_SWS_CnV2xNet_00119] [CP_SWS_CnV2xNet_00120] [CP_SWS_CnV2xNet_00121]
[CP_SRS_CnV2X_-00403]	The network layer of Chinese V2X communication shall provide the mapping between packet priority and PPPP	[CP_SWS_CnV2xNet_00108] [CP_SWS_CnV2xNet_00117]
[CP_SRS_CnV2X_-00404]	The network layer of Chinese V2X communication shall provide CBR or Max data rate to message Layer	[CP_SWS_CnV2xNet_00118]
[CP_SRS_CnV2X_-00605]	The Chinese V2X communication shall randomize the identifiers related to BSM to in order to support privacy	[CP_SWS_CnV2xNet_00112]

Requirement	Description	Satisfied by
[CP_SRS_CnV2X_-00606]	The Chinese V2X communication shall change pseudonym certificates in order to support privacy	[CP_SWS_CnV2xNet_00122] [CP_SWS_CnV2xNet_00123] [CP_SWS_CnV2xNet_00124] [CP_SWS_CnV2xNet_00125]
[SRS_BSW_00345]	BSW Modules shall support pre-compile configuration	[SWS_CnV2xNet_03001]

7 Functional specification

7.1 General Functionality

[CP_SWS_CnV2xNet_00100]{DRAFT} ⌈The CnV2xNet module shall implement the network Layer protocols defined in [2] unless specified otherwise in this document.⌋ *(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_00101]{DRAFT} ⌈The network protocol shall meet the network layer related requirements defined in [1].⌋ *(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_00102]{DRAFT} ⌈The CnV2xNet Module shall encapsulate the payload from the CnV2xMsg module with a DSMP header and Adaptation layer header as per [2].⌋ *(CP_SRS_CnV2X_00401)*

7.2 Message Transmission

[CP_SWS_CnV2xNet_00103]{DRAFT} ⌈The CnV2xNet module shall provide the API CnV2xNet_Transmit () to enable transmit requests from the CnV2xMsg module.⌋ *(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_00104]{DRAFT} ⌈The CnV2xNet module shall use EthIf_ProvideTxBuffer to acquire a buffer within the C-V2X Driver for a V2X Packet to be transmitted. This has to be done during the CnV2xNet_Transmit context.⌋ *(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_00105]{DRAFT} ⌈The CnV2xNet module shall only support DSMP based service to CnV2xMsg Module, and employ DSMP and Adaptation layer headers.⌋ *(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_00106]{DRAFT} ⌈The DSMP protocol version shall be set to zero.⌋ *(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_00108]{DRAFT} ⌈The CnV2xNet module shall select a PPPP value for a V2X packet to be transmitted based on Priority indicated in CnV2xNet_Transmit () and the mapping relationship defined in Table A.2 of [2].⌋ *(CP_SRS_CnV2X_00403)*

[CP_SWS_CnV2xNet_00109]{DRAFT} ⌈The CnV2xNet module shall provide traffic period indicated in CnV2xNet_Transmit () (if provided by CnV2xMsg Module) to lower layer using EthIf_SetBufCv2xPC5TxParams().⌋ *(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_00110]{DRAFT} ⌈The CnV2xNet module shall provide transmission parameters to the C-V2X Driver for a V2X Packet to be transmitted via an API call to EthIf_SetBufCv2xPC5TxParams. This has to be done during the CnV2xNet_Transmit context.⌋ *(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_00111]{DRAFT} ⌈The CnV2xNet module shall transmit packets using the EthIf_Transmit () provided by the EthIf Module. This has to be done during the CnV2xNet_Transmit context.⌋*(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_00112]{DRAFT} ⌈The CnV2xNet module shall suspend transmission of V2X packet when a pseudonym certificate changes is in preparation.⌋*(CP_SRS_CnV2X_00605)*

[CP_SWS_CnV2xNet_00113]{DRAFT} ⌈If the configuration parameter CnV2xNet_TxConfirmation is enabled, the CnV2xNet module shall provide information about the status of the transmission with an associated ID (generated by the CnV2xMsg module and handed down to track the status of the packet) the CnV2xMsg Module via the CnV2xMsg_TxConfirmation () (for details see chapter 8.4.1 in [3])callback.⌋*(CP_SRS_CnV2X_00401)*

NOTE: A dedicated EtherType value can be considered for Chinese V2X network layer during in-vehicle communication. It is up to implementation for whether a private value or a registered value is used.

7.3 Message Reception

[CP_SWS_CnV2xNet_00114]{DRAFT} ⌈The CnV2xNet module shall create a unique TransactionId for each received packet. This TransactionId is handed up to track the received packets and is used for verification.⌋*(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_00115]{DRAFT} ⌈The CnV2xNet module shall indicate received packets via the CnV2xMsg_RxIndication () callback to the CnV2xMsg module.⌋*(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_00116]{DRAFT} ⌈The CnV2xNet module shall get the reception status of a received packet during the CnV2xNet_RxIndication () from the EthIf module with a call to EthIf_GetBufCv2xPC5RxParams ().⌋*(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_00117]{DRAFT} ⌈The CnV2xNet module shall map PPPP indicated in CnV2xNet_RxIndication () to priority and provides to upper layer through CnV2xMsg_RxIndication ().⌋*(CP_SRS_CnV2X_00403)*

[CP_SWS_CnV2xNet_00118]{DRAFT} ⌈The CnV2xNet module shall provide the received CBR or max data rate indicated in CnV2xNet_RxIndication () to upper layer through CnV2xMsg_RxIndication ().⌋*(CP_SRS_CnV2X_00404)*

7.4 Layer-2 ID selection

[CP_SWS_CnV2xNet_00119]{DRAFT} ⌈The CnV2xNet module shall implement the source Layer-2 ID and destination Layer-2 ID selection and maintenance.⌋*(CP_SRS_CnV2X_00402)*

[CP_SWS_CnV2xNet_00120]{DRAFT} ⌈The CnV2xNet module shall randomly select a source layer-2 ID within [0x010001, 0xFFFFFE] during initialization as per [1]. It shall randomly reselect the source layer-2 ID during the application layer ID changing context.⌋*(CP_SRS_CnV2X_00402)*

[CP_SWS_CnV2xNet_00121]{DRAFT} ⌈The CnV2xNet module shall select a destination layer-2 ID for a V2X packet to be transmitted based on Aid indicated in CnV2xNet_Transmit () and the mapping relationship defined in Table 3 of [1].⌋*(CP_SRS_CnV2X_00402)*

[CP_SWS_CnV2xNet_00122]{DRAFT} ⌈The CnV2xNet shall perform the changing of the source Layer-2 ID in two phases, which consist of the prepare phase and the commit/abort phase. The second phase depends on the result of all called modules within the first phase. If the first phase was successful, the commit phase shall be performed. Otherwise, the abort phase shall be triggered.⌋*(CP_SRS_CnV2X_00606)*

[CP_SWS_CnV2xNet_00123]{DRAFT} ⌈In the prepare phase, the API CnV2xNet_PrepareAppLayerIdChange() shall be called by CnV2xMsg.

⌋*(CP_SRS_CnV2X_00606)*

[CP_SWS_CnV2xNet_00124]{DRAFT} ⌈In the commit phase, the API CnV2xNet_CommitAppLayerIdChange() shall be called by CnV2xMsg.

⌋*(CP_SRS_CnV2X_00606)*

[CP_SWS_CnV2xNet_00125]{DRAFT} ⌈In the commit phase, the API CnV2xNet_CommitAppLayerIdChange() shall be called by CnV2xMsg.

⌋*(CP_SRS_CnV2X_00606)*

7.5 Error Classification

This chapter lists and classifies all errors that can be detected within this software module. Each error is classified according to relevance (development / production) and related error code. For development errors, a value is defined.

7.5.1 Development Errors

[CP_SWS_CnV2xNet_00126]{DRAFT} **Development Error Types** ⌈The following table lists development errors that shall be distinguished by the CnV2xNet module. CnV2xNet shall report them to the DET, if development error detection (CnV2xNetDevErrorDetect) is enabled⌋*()*

Type of error	*Related error code*	*Value [hex]*
API service called with invalid parameter	CNV2XNET_E_PARAM	0x01
API service called with invalid pointer	CNV2XNET_E_PARAM_POINTER	0x02
API function called before the CnV2xNet module has been fully initialized	CNV2XNET_E_UNINIT	0x03
CnV2xNet initialization failed	CNV2XNET_E_INIT_FAILED	0x04

Table 7.1: Development Error Types for CnV2xNet

7.5.2 Runtime Errors

There are no runtime errors.

7.5.3 Transient Faults

There are no transient faults.

7.5.4 Production Errors

There are no production errors.

7.5.5 Extended Production Errors

There are no extended production errors.

8 API specification

8.1 Imported types

In this chapter all types included from the following files are listed.

[CP_SWS_CnV2xNet_01001] ⌈

Module	*Header File*	*Imported Type*
CnV2xMsg	CnV2xMsg.h	CnV2xMsg_RxParamsPresenceType (draft)
	CnV2xMsg.h	CnV2xMsg_RxParamsType (draft)
ComStack_Types	ComStack_Types.h	BufReq_ReturnType
CV2x	CV2x_GeneralTypes.h	CV2x_BufCV2xPC5RxParamIdType (draft)
	CV2x_GeneralTypes.h	CV2x_BufCV2xPC5TxParamIdType (draft)
Eth	Eth_GeneralTypes.h	Eth_BufIdxType
	Eth_GeneralTypes.h	Eth_FrameType
Std	Std_Types.h	Std_ReturnType
	Std_Types.h	Std_VersionInfoType

⌋()

8.2 Type definitions

8.2.1 CnV2xNet_TxParamsType

[CP_SWS_CnV2xNet_01002]{DRAFT} ⌈

Name	CnV2xNet_TxParamsType (draft)	
Kind	Structure	
Elements	presence	
	Type	CnV2xNet_TxParamsPresenceType
	Comment	Mark optional child present or not
	Aid	
	Type	uint64
	Comment	The value of the AID (Application Identifier)
	ProtocolType	
	Type	CnV2x_NetworkProtocolType
	Comment	Network layer protocol type. Value 4 is used for DSMP protocol and other values are reserved.
	priority	
	Type	uint8
	Comment	Priority of V2X-CN message
	SourceLayer2Id	
	Type	CnV2x_Layer2IdType

▽

△

	Comment	Source Layer 2 ID of V2X-CN packet(24-bit)
	DestinationLayer2Id	
	Type	CnV2x_Layer2IdType
	Comment	Destination Layer-2 ID of V2X-CN packet(24-bit)
	TrafficPeriod	
	Type	CnV2x_TrafficPeriodType
	Comment	Indicate Traffic Period
	AppLayerIdChangedCount16	
	Type	uint16
	Comment	The order of the Application layer Id Changed. This value is created in the CnV2xMsg module and shall be mapped with parameter pseudonymCount16.
	DsmpHeaderExtensionPtr	
	Type	uint8*
	Comment	Ptr of Dsmp header Extension
	DsmpHeaderExtensionLength	
	Type	uint16
	Comment	Length of Dsmp header Extension
Description	Wraps Message layer parameters from CnV2xMsg **Tags:** atp.Status=draft	
Available via	CnV2xNet.h	

⌋(*CP_SRS_CnV2X_00401*)

8.2.2 CnV2xNet_TxParamsPresenceType

[CP_SWS_CnV2xNet_01003]{DRAFT} ⌈

Name	CnV2xNet_TxParamsPresenceType (draft)			
Kind	Bitfield			
Derived from	uint8			
Elements	***Kind***	***Name***	***Mask***	***Description***
	bit	SourceLayer2Id	0x08	Bit 3: Optional child present
	bit	DestinationLayer2Id	0x04	Bit 2: Optional child present
	bit	TrafficPeriod	0x02	Bit 1: Optional child present
	bit	DsmpHeaderExtension	0x01	Bit 0 (LSB): Optional child(Dsmp HeaderExtensionPtr and Dsmp HeaderExtensionLength) present
Description	Presence flags for CnV2xNet_TxParamsType **Tags:** atp.Status=draft			
Available via	CnV2x_GeneralTypes.h			

⌋(*CP_SRS_CnV2X_00401*)

8.2.3 CnV2xNet_RxParamsType

[CP_SWS_CnV2xNet_01004]{DRAFT} ⌈

Name	CnV2xNet_RxParamsType (draft)	
Kind	Structure	
Elements	presence	
	Type	CnV2xNet_RxParamsPresenceType
	Comment	Mark optional child present or not
	Sourcelayer2Id	
	Type	CnV2x_Layer2IdType
	Comment	Source Layer 2 ID of V2X-CN packet(24bit)
	DestinationLayer2Id	
	Type	CnV2x_Layer2IdType
	Comment	Destination Layer 2 ID of V2X-CN packet(24bit)
	pppp	
	Type	CnV2x_PPPPType
	Comment	ProSe per-packet priority
	cbr	
	Type	CnV2x_CbrType
	Comment	Channel busy rate
	MaxDataRate	
	Type	CnV2x_MaxDataRateType
	Comment	Max data rate
Description	Structure containing Access layer parameters related to a received C-V2X packet. **Tags:** atp.Status=draft	
Available via	CnV2xNet.h	

⌋*(CP_SRS_CnV2X_00401)*

8.2.4 CnV2xNet_RxParamsPresenceType

[CP_SWS_CnV2xNet_01005]{DRAFT} ⌈

Name	CnV2xNet_RxParamsPresenceType (draft)			
Kind	Bitfield			
Derived from	uint8			
Elements	***Kind***	***Name***	***Mask***	***Description***
	bit	Cbr	0x02	Bit 1: Optional child present
	bit	maxDataRate	0x01	Bit 0 (LSB): Optional child present
Description	Presence flags for CnV2xNet_RxParamsType **Tags:** atp.Status=draft			
Available via	CnV2x_GeneralTypes.h			

⌋*()*

8.2.5 CnV2x_Layer2IdType

[CP_SWS_CnV2xNet_01006]{DRAFT} ⌈

Name	CnV2x_Layer2IdType (draft)		
Kind	Type		
Derived from	uint32		
Range	0..16777215	–	–
Description	The Layer 2 Id (24bit) **Tags:** atp.Status=draft		
Variation	–		
Available via	CnV2x_GeneralTypes.h		

⌋(*CP_SRS_CnV2X_00401*)

8.2.6 CnV2x_PPPPType

[CP_SWS_CnV2xNet_01007]{DRAFT} ⌈

Name	CnV2x_PPPPType (draft)		
Kind	Type		
Derived from	uint8		
Range	1..8	–	–
Description	Prose per-packet priority of V2X message **Tags:** atp.Status=draft		
Available via	CnV2x_GeneralTypes.h		

⌋(*CP_SRS_CnV2X_00401*)

8.2.7 CnV2x_NetworkProtocolType

[CP_SWS_CnV2xNet_01008]{DRAFT} ⌈

Name	CnV2x_NetworkProtocolType (draft)		
Kind	Type		
Derived from	uint8		
Range	CNV2X_DSMP_ PROTOCOL	0x04	DSMP protocol type
Description	Enumeration Type as defined in CCSA YD/T 3709-2020. **Tags:** atp.Status=draft		
Available via	CnV2x_GeneralTypes.h		

⌋(*CP_SRS_CnV2X_00401*)

8.2.8 CnV2x_TrafficPeriodType

[CP_SWS_CnV2xNet_01009]{DRAFT} ⌈

Name	CnV2x_TrafficPeriodType (draft)		
Kind	Type		
Derived from	uint8		
Range	CNV2X_TRAFFIC_PERIOD_20	0x00	Traffic Period: 20ms
	CNV2X_TRAFFIC_PERIOD_50	0x01	Traffic Period: 50ms
	CNV2X_TRAFFIC_PERIOD_100	0x02	Traffic Period: 100ms
	CNV2X_TRAFFIC_PERIOD_200	0x03	Traffic Period: 200ms
	CNV2X_TRAFFIC_PERIOD_300	0x04	Traffic Period: 300ms
	CNV2X_TRAFFIC_PERIOD_400	0x05	Traffic Period: 400ms
	CNV2X_TRAFFIC_PERIOD_500	0x06	Traffic Period: 500ms
	CNV2X_TRAFFIC_PERIOD_600	0x07	Traffic Period: 600ms
	CNV2X_TRAFFIC_PERIOD_700	0x08	Traffic Period: 700ms
	CNV2X_TRAFFIC_PERIOD_800	0x09	Traffic Period: 800ms
	CNV2X_TRAFFIC_PERIOD_900	0x0a	Traffic Period: 900ms
	CNV2X_TRAFFIC_PERIOD_1000	0x0b	Traffic Period: 1000ms
Description	Enumeration Type as defined in CCSA YD/T 3709-2020. **Tags:** atp.Status=draft		
Available via	CnV2x_GeneralTypes.h		

⌋(*CP_SRS_CnV2X_00401*)

8.2.9 CnV2x_CbrType

[CP_SWS_CnV2xNet_01010]{DRAFT} ⌈

Name	CnV2x_CbrType (draft)		
Kind	Type		
Derived from	uint8		
Range	0..100	–	–
Description	Channel busy rate % **Tags:** atp.Status=draft		
Variation	–		
Available via	CnV2x_GeneralTypes.h		

⌋(*CP_SRS_CnV2X_00401*)

8.2.10 CnV2x_MaxDataRateType

[CP_SWS_CnV2xNet_01011]{DRAFT} ⌈

Name	CnV2x_MaxDataRateType (draft)		
Kind	Type		
Derived from	uint32		
Range	0..1585200	–	–
Description	Max date Rate uint: bps **Tags:** atp.Status=draft		
Variation	–		
Available via	CnV2x_GeneralTypes.h		

⌋(*CP_SRS_CnV2X_00401*)

8.2.11 CnV2x_NetTxResultType

[CP_SWS_CnV2xNet_01012]{DRAFT} ⌈

Name	CnV2x_NetTxResultType (draft)		
Kind	Type		
Derived from	uint8		
Range	CNV2X_NETTX_ ACCEPTED	0x00	–
	CNV2X_NETTX_E_ MAXSDUSIZEOVF	0x01	Traffic Period: 50ms
	CNV2X_NETTX_E_AID	0x02	transmit has been rejected due to unsupported AID
	CNV2X_NETTX_E_ PRTOCOLTYPE	0x03	transmit has been rejected due to maximum length exceedance
	CNV2X_NETTX_E_ PRIORITY	0x04	transmit has been rejected due to unsupported priority
	CNV2X_NETTX_E_ LAYER2ID_S	0x05	transmit has been rejected due to uncorrected source Layer 2 ID
	CNV2X_NETTX_E_ LAYER2ID_D	0x06	transmit has been rejected due to uncorrected destination Layer 2 ID
	CNV2X_NETTX_E_TP	0x07	transmit has been rejected due to unsupported traffic period
	CNV2X_NETTX_E_ UNSPECIFIED	0x08	transmit has been rejected due to unspecified reasons
Description	Return Types of API CnV2xNet_Transmit. **Tags:** atp.Status=draft		
Available via	CnV2x_GeneralTypes.h		

⌋(*CP_SRS_CnV2X_00401*)

8.3 Function definition

8.3.1 CnV2xNet_Init

[CP_SWS_CnV2xNet_02001]{DRAFT} ⌈

Service Name	CnV2xNet_Init (draft)	

Syntax	`void CnV2xNet_Init (`
	`  void* CfgPtr`
	`)`	

Service ID [hex]	0x1	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CfgPtr	Points to a null pointer
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Initialize the CnV2xNet module **Tags:** atp.Status=draft	
Available via	CnV2xNet.h	

⌋*(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_02030]{DRAFT} ⌈The CfgPtr must be a Null pointer and as a rationale, that initialization at runtime (post build init) is not possible.⌋*()*

[CP_SWS_CnV2xNet_02002]{DRAFT} ⌈If development error detection is enabled: The function shall check the parameter CfgPtr for containing a valid configuration. If the check fails, the function shall raise the development error CNV2XNET_E_INIT_FAILED.⌋*()*

8.3.2 CnV2xNet_GetVersionInfo

[CP_SWS_CnV2xNet_02003]{DRAFT} ⌈

Service Name	CnV2xNet_GetVersionInfo (draft)	

Syntax	`void CnV2xNet_GetVersionInfo (`
	`  Std_VersionInfoType* VersionInfoPtr`
	`)`	

Service ID [hex]	0x2	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	VersionInfoPtr	Pointer to where to store the version information of this module.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	

▽

△

Description	Returns the version information of this module. **Tags:** atp.Status=draft
Available via	CnV2xNet.h

⌋(*CP_SRS_CnV2X_00401*)

[CP_SWS_CnV2xNet_02004]{DRAFT} ⌈If development error detection is enabled: the function shall check the parameter VersionInfoPtr for being valid. If the check fails, the function shall raise the development error CNV2XNET_E_PARAM_POINTER.⌋*()*

8.3.3 CnV2xNet_Transmit

[CP_SWS_CnV2xNet_02005]{DRAFT} ⌈

Service Name	CnV2xNet_Transmit (draft)	
Syntax	`CnV2x_NetTxResultType CnV2xNet_Transmit ( uint16 TransactionId16, const CnV2xNet_TxParamsType* TxParams, uint16 Length, const uint8* DataPtr )`	
Service ID [hex]	0x3	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	TransactionId16	ID identifying the payload to be transmitted. This ID is generated by the CnV2xMsg module and is used later to indicate the status of the transmission of the message having this ID to the CnV2x Msg module.
	TxParams	Structure containing all the Network layer parameters used for the transmit request.
	Length	Length of the data pointed by DataPtr.
	DataPtr	Payload of the Message Layer packet to be transmitted
Parameters (inout)	None	
Parameters (out)	None	
Return value	CnV2x_NetTxResultType	CNV2X_NETTX_ACCEPTED if no error occurred. CNV2X_NETTX_E_MAXSDUSIZEOVFL transmit has been rejected due to maximum length exceedance CNV2X_NETTX_E_AID transmit has been rejected due to unsupported AID CNV2X_NETTX_E_PRTOCOLTYPE transmit has been rejected due to unsupported protocol type CNV2X_NETTX_E_PRIORITY transmit has been rejected due to unsupported priority CNV2X_NETTX_E_LAYER2ID_S transmit has been rejected due to uncorrected source Layer 2 ID CNV2X_NETTX_E_LAYER2ID_D transmit has been rejected due to uncorrected destination Layer 2 ID CNV2X_NETTX_E_TP transmit has been rejected due to unsupported traffic period CNV2X_NETTX_E_UNSPECIFIED transmit has been rejected due to unspecified reasons
Description	This API is called by the CvxMsgCN module to request sending a Network Layer V2X PDU to the peer Network entity. **Tags:** atp.Status=draft	
Available via	CnV2xNet.h	

⌋(*CP_SRS_CnV2X_00401*)

[CP_SWS_CnV2xNet_02006]{DRAFT} ⌈If development error detection is enabled: the function shall check that the service CnV2xNet_Init was previously called. If the check fails, the function shall raise the development error CNV2XNET_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK.⌋*()*

[CP_SWS_CnV2xNet_02007]{DRAFT} ⌈If development error detection is enabled: the function shall check the parameter TxParams for being valid. If the check fails, the function shall raise the development error CNV2XNET_E_PARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.⌋*()*

[CP_SWS_CnV2xNet_02008]{DRAFT} ⌈If development error detection is enabled: the function shall check the parameter DataPtr for being valid. If the check fails, the function shall raise the development error CNV2XNET_E_PARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.⌋*()*

8.3.4 CnV2xNet_PrepareAppLayerIdChange

[CP_SWS_CnV2xNet_02009]{DRAFT} ⌈

Service Name	CnV2xNet_PrepareAppLayerIdChange (draft)	
Syntax	`Std_ReturnType CnV2xNet_PrepareAppLayerIdChange ( uint8 TransmissionClass, uint16 ApplayerIdChangedCount16 )`	
Service ID [hex]	0x4	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	TransmissionClass	Transmission Class Indication
	ApplayerIdChanged Count16	Order of the Application layer identifier changed correspond to specific message type. This count value is created in the CnV2x Sec module.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: operation successful E_NOT_OK: operation failed
Description	By this API primitive the CnV2xNet module gets an indication that Application Layer Id is about to change and hereby source Layer-2 ID is about to be changed. **Tags:** atp.Status=draft	
Available via	CnV2xNet.h	

⌋*(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_02010]{DRAFT} ⌈The function shall prepare the setting of source Layer-2 ID used for packet transmission.⌋*()*

[CP_SWS_CnV2xNet_02011]{DRAFT} ⌈If development error detection is enabled: the function shall check that the service CnV2xNet_Init was previously called. If the check fails, the function shall raise the development error CNV2XNET_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK.⌋*()*

8.3.5 CnV2xNet_CommitAppLayerIdChange

[CP_SWS_CnV2xNet_02012]{DRAFT} ⌈

Service Name	CnV2xNet_CommitAppLayerIdChange (draft)	
Syntax	`Std_ReturnType CnV2xNet_CommitAppLayerIdChange ( uint8 TransmissionClass, uint16 ApplayerIdChangedCount16 )`	
Service ID [hex]	0x5	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	TransmissionClass	Transmission Class Indication
	ApplayerIdChanged Count16	Order of the Application layer identifier changed correspond to specific message type. This count value is created in the CnV2x Sec module.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: operation successful E_NOT_OK: operation failed
Description	The CnV2xMsg module calls this function when all modules are OK with the pseudonym certificate change and the change is to be committed. **Tags:** atp.Status=draft	
Available via	CnV2xNet.h	

⌋*(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_02013]{DRAFT} ⌈The function shall update the new source Layer-2 ID.⌋*()*

[CP_SWS_CnV2xNet_02014]{DRAFT} ⌈If development error detection is enabled: the function shall check that the service CnV2xNet_Init was previously called. If the check fails, the function shall raise the development error CNV2XNET_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK.⌋*()*

Note: The function requires previous preparation of the pseudonym certificate via an API call to CnV2xNet_PrepareAppLayerIdChange.

8.3.6 CnV2xNet_AbortAppLayerIdChange

[CP_SWS_CnV2xNet_02015]{DRAFT} ⌈

Service Name	CnV2xNet_AbortAppLayerIdChange (draft)
Syntax	`Std_ReturnType CnV2xNet_AbortAppLayerIdChange ( uint8 TransmissionClass, uint16 ApplayerIdChangedCount16 )`
Service ID [hex]	0x6
Sync/Async	Synchronous
Reentrancy	Non Reentrant

▽

△

Parameters (in)	TransmissionClass	Transmission Class Indication
	ApplayerIdChanged Count16	Order of the Application layer identifier changed correspond to specific message type. This count value is created in the CnV2x Sec module.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: operation successful E_NOT_OK: operation failed
Description	The CnV2xMsg module calls this function when not all modules are OK with the pseudonym certificate change and the change is to be rolled back. **Tags:** atp.Status=draft	
Available via	CnV2xNet.h	

⌋(*CP_SRS_CnV2X_00401*)

[CP_SWS_CnV2xNet_02016]{DRAFT} ⌈The function shall roll back the prepared source Layer-2 ID change.⌋*()*

[CP_SWS_CnV2xNet_02017]{DRAFT} ⌈If development error detection is enabled: the function shall check that the service CnV2xNet_Init was previously called. If the check fails, the function shall raise the development error CnV2xNet_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK.⌋*()*

Note: The function requires previous preparation of the pseudonym certificate via an API call to CnV2xNet_PrepareAppLayerIdChange.

8.4 Callback notifications

8.4.1 CnV2xNet_RxIndication

[CP_SWS_CnV2xNet_02018]{DRAFT} ⌈

Service Name	CnV2xNet_RxIndication (draft)	

Syntax	`void CnV2xNet_RxIndication (`
	`  uint8 CtrlIdx,`
	`  Eth_FrameType FrameType,`
	`  boolean IsBroadcast,`
	`  const uint8* PhysAddrPtr,`
	`  uint8* DataPtr,`
	`  uint16 LenByte`
	`)`	

Service ID [hex]	0x7	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	FrameType	frame type of received Ethernet frame
	IsBroadcast	parameter to indicate a broadcast frame

▽

△

	PhysAddrPtr	pointer to Physical source address (MAC address in network byte order) of received Ethernet frame
	DataPtr	Pointer to payload of the received Ethernet frame (i.e. Ethernet header is not provided).
	LenByte	Length of received data.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Indicates the reception of an Ethernet frame **Tags:** atp.Status=draft	
Available via	CnV2xNet.h	

⌋(*CP_SRS_CnV2X_00401*)

[CP_SWS_CnV2xNet_02023]{DRAFT} ⌈The function CnV2xNet_RxIndication shall get reception parameters of the C-V2X Driver for a C-V2X Packet received via an API call to EthIf_GetBufCv2xPC5RxParams.⌋*()*

[CP_SWS_CnV2xNet_02024]{DRAFT} ⌈If development error detection is enabled: the function shall check that the service CnV2xNet_Init was previously called. If the check fails, the function shall raise the development error CNV2XNET_E_UNINIT.⌋*()*

[CP_SWS_CnV2xNet_02025]{DRAFT} ⌈If development error detection is enabled: the function shall check the parameter DataPtr for being valid. If the check fails, the function shall raise the development error CNV2XNET_E_PARAM_POINTER.⌋*()*

Note: Eth_FrameType and PhysAddrPtr are not used but reserved in CnV2xNet_RxIndication.

8.4.2 CnV2xNet_TxConfirmation

[CP_SWS_CnV2xNet_02019]{DRAFT} ⌈

Service Name	CnV2xNet_TxConfirmation (draft)	

Syntax	`void CnV2xNet_TxConfirmation (`
	`  uint8 CtrlIdx,`
	`  uint8 BufIdx`
	`)`	

Service ID [hex]	0x8	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	BufIdx	Index of the buffer resource
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	

▽

△

Description	Confirms the transmission of an Ethernet frame **Tags:** atp.Status=draft
Available via	CnV2xNet.h

⌋*(CP_SRS_CnV2X_00401)*

[CP_SWS_CnV2xNet_02026]{DRAFT} ⌈If development error detection is enabled: the function shall check that the service CnV2xNet_Init was previously called. If the check fails, the function shall raise the development error CNV2XNET_E_UNINIT.⌋*()*

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 CnV2xNetMainFunction

[CP_SWS_CnV2xNet_02020]{DRAFT} ⌈

Service Name	CnV2xNet_MainFunction (draft)

Syntax	`void CnV2xNet_MainFunction (`
	`  void`
	`)`

Service ID [hex]	0x9
Description	Main function of the CnV2xNet module for periodical execution of protocol operations. **Tags:** atp.Status=draft
Available via	SchM_CnV2xNet.h

⌋*(CP_SRS_CnV2X_00401)*

8.6 Expected interfaces

8.6.1 Mandatory interfaces

This section defines all external interfaces, which are required to fulfill the core functionality of the module.

[CP_SWS_CnV2xNet_02021] ⌈

API Function	*Header File*	*Description*
CnV2xMsg_RxIndication (draft)	CnV2xMsg.h	By this API primitive the CnV2xMsg module gets a confirmation that the V2X message with a certain ID was send successfully. This API primitive is called by the CnV2xNet module providing the data and the Network parameters of a received DSMP packet to CnV2xMsg module. **Tags:** atp.Status=draft
EthIf_GetBufCV2xPC5RxParams	EthIf.h	Read out values related to the receive direction of the Cellular V2X for a received packet. For example, this could be CBR belonging to one single packet.
EthIf_GetBufCV2xPC5TxParams	EthIf.h	Read out values related to the transmit direction of the Cellular V2X for a transmitted packet. For example, this could be transaction ID belonging to one single packet.
EthIf_ProvideTxBuffer	EthIf.h	Provides access to a transmit buffer of the specified Ethernet controller.
EthIf_SetBufCV2xPC5TxParams	EthIf.h	Set values related to the transmit direction of the Cellular V2X for a specific buffer (packet to be sent). For example, this can be the desired ProSe per-packet priority belonging to one single packet.
EthIf_Transmit	EthIf.h	Triggers transmission of a previously filled transmit buffer

⌋()

8.6.2 Optional interfaces

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

[CP_SWS_CnV2xNet_02022] ⌈

API Function	*Header File*	*Description*
CnV2xMsg_TxConfirmation (draft)	CnV2xMsg.h	By this API primitive, the CnV2xMsg module gets a confirmation that the V2X message with a certain ID was send successfully. **Tags:** atp.Status=draft
Det_ReportError	Det.h	Service to report development errors.

⌋()

9 Sequence diagrams

9.1 RxIndication

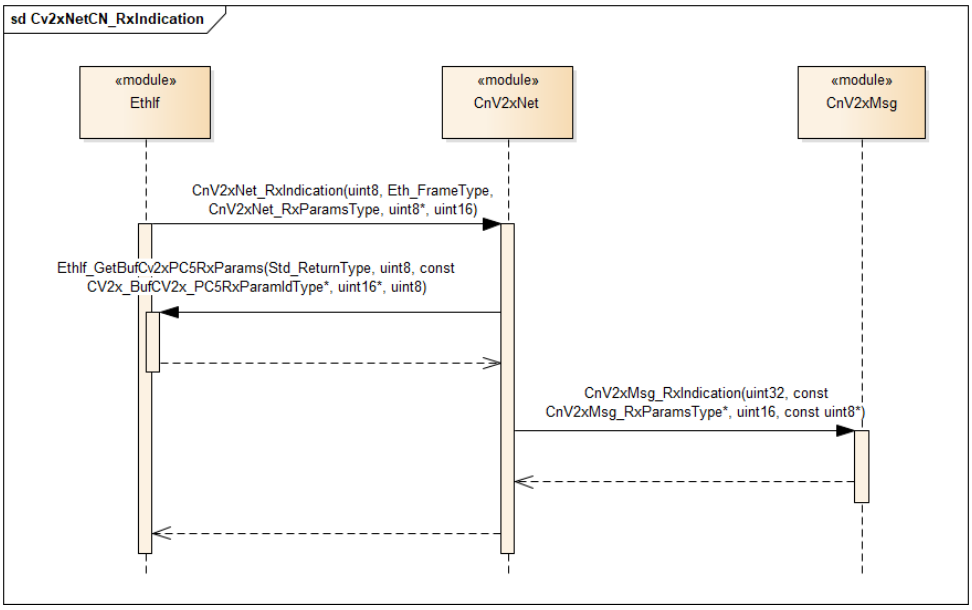

Figure 9.1: RxIndication

9.2 Transmission

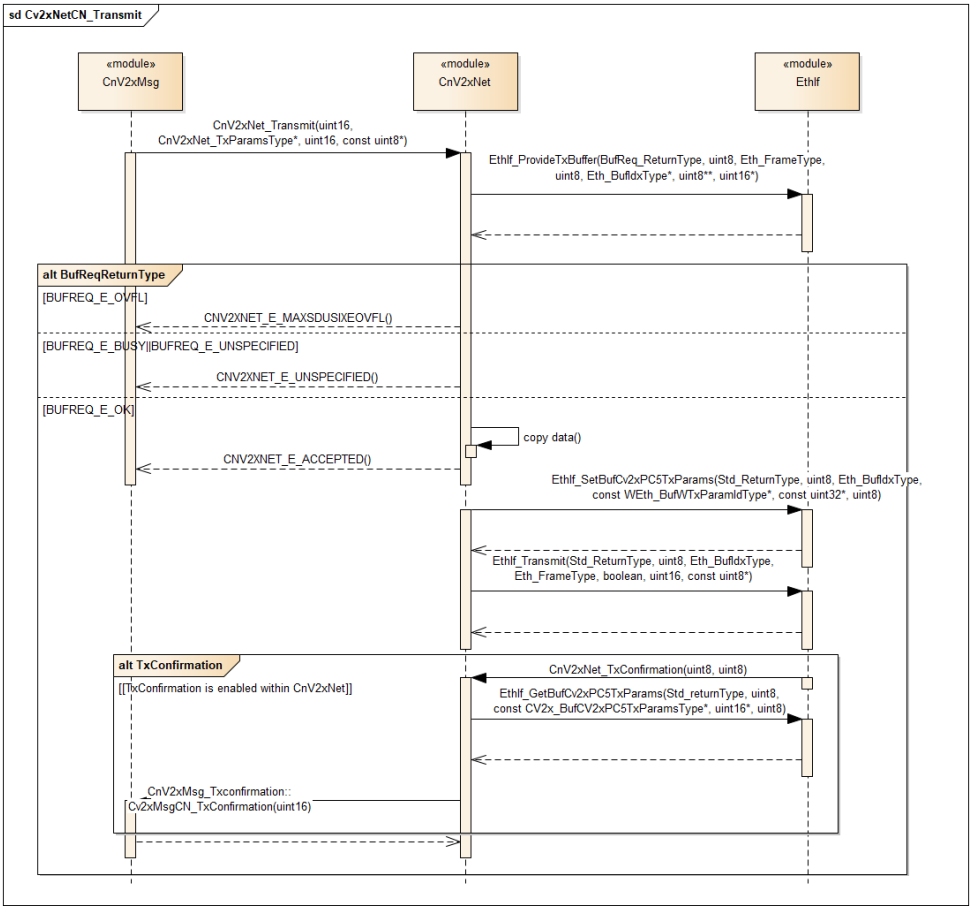

Figure 9.2: Transmission

10 Configuration specification

10.1 Containers and configuration parameters

The following chapters summarize all configuration parameters. The detailed meanings of the parameters are described in Chapter 7 and Chapter 8.

10.1.1 Variant

[SWS_CnV2xNet_03001]{DRAFT} ⌈The CnV2xNet module only supports VARIANT-PRECOMPILE⌋ *(SRS_BSW_00345)*

10.1.2 CnV2xNet

SWS Item	[ECUC_CnV2xNet_00001]
Module Name	CnV2xNet
Description	Configuration of the CnV2xMsg module.
Post-Build Variant Support	false
Supported Config Variants	VARIANT-PRE-COMPILE

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CnV2xNetGeneral	1	This container contains the configuration parameters of the BSW module CnV2xNet. **Tags:** atp.Status=draft

10.1.3 CnV2xNetGeneral

SWS Item	[ECUC_CnV2xNet_00002]
Container Name	CnV2xNetGeneral
Parent Container	CnV2xNet
Description	This container contains the configuration parameters of the BSW module CnV2xNet. **Tags:** atp.Status=draft
Configuration Parameters	

SWS Item	[ECUC_CnV2xNet_00004]
Parameter Name	CnV2xNetDevErrorDetect
Parent Container	CnV2xNetGeneral

▽

△

Description	Switches the Default Error Tracer (Det) detection and notification ON or OFF. - true: enabled (ON) - false: disabled (OFF) **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: local		

SWS Item	**[ECUC_CnV2xNet_00008]**		
Parameter Name	CnV2xNetDSMPprotocolVersion		
Parent Container	CnV2xNetGeneral		
Description	DSMP protocol version as defined in chapter 4.3.3.1 [13] **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 7		
Default value	0		
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_CnV2xNet_00009]**		
Parameter Name	CnV2xNetDSMPSduSize		
Parent Container	CnV2xNetGeneral		
Description	Maximum size of DSMP-SDU in [Byte] **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	65535		
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_CnV2xNet_00003]**
Parameter Name	CnV2xNetMainFunctionPeriod
Parent Container	CnV2xNetGeneral

▽

△

Description	This parameter defines the schedule period of CnV2xNet_BsmBs_Main Function.Unit:[s] **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	]0 .. 1[		
Default value	0.1		
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_CnV2xNet_00006]**		
Parameter Name	CnV2xNetTxConfirmation		
Parent Container	CnV2xNetGeneral		
Description	When enabled, transmission status information will be forwarded to the upper layer. - true: enabled (ON) - false: disabled (OFF) **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: local		

SWS Item	**[ECUC_CnV2xNet_00005]**		
Parameter Name	CnV2xNetVersionInfoApi		
Parent Container	CnV2xNetGeneral		
Description	Enable/disables the API for reading the version information of the CnV2xNet Module. - true: enabled (ON) - false: disabled (OFF) **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: local		

SWS Item	**[ECUC_CnV2xNet_00007]**		
Parameter Name	CnV2xNetEthIfRef		
Parent Container	CnV2xNetGeneral		

▽

△

Description	This is represents the reference to the Ethernet interface taken to transmit the C-V2X packets to. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	Symbolic name reference to EthIfController		
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

A Not applicable requirements