Document Title	Specification of Vehicle-2-X Basic Transport
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	794

Document Status	Final
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	4.3.1

Document Change History

Date	Release	Changed by	Change Description
2017-12-08	4.3.1	AUTOSAR Release Management	• Editorial changes
2016-11-30	4.3.0	AUTOSAR Release Management	• Initial Release

Disclaimer

Table of Contents

1 Introduction and functional overview

This specification specifies the functionality, API and the configuration of the AUTOSAR Basic Software module Vehicle-2-X Basic Transport (called “V2xBtp module” in this document).

V2xBtp module together with Vehicle-2-X Facilities [8] (V2xFac), Vehicle-2-X GeoNetworking [9] (V2xGn), Vehicle-2-X Management (V2xM) and AUTOSAR BSW module Ethernet Interface (EthIf) [5] forms the V2X stack within the AUTOSAR architecture.

The base for this document is the BTP specification [12]. It is assumed that the reader is familiar with this specification. This document will not redefine BTP functionality, but it will try to follow the same order as the BTP specification.

1.1 Architectural overview

V2xBtp module provides services to and is dependent on the upper V2xFac module and uses the services of and gets services from the lower V2xGn module to realize its functionality explained in sections 1.2, 7.1, 7.2, 7.3 of this document.

Positioning of the V2xBtp module within the AUTOSAR BSW and the Layered Software architecture [1] is shown in Figure 1 below.

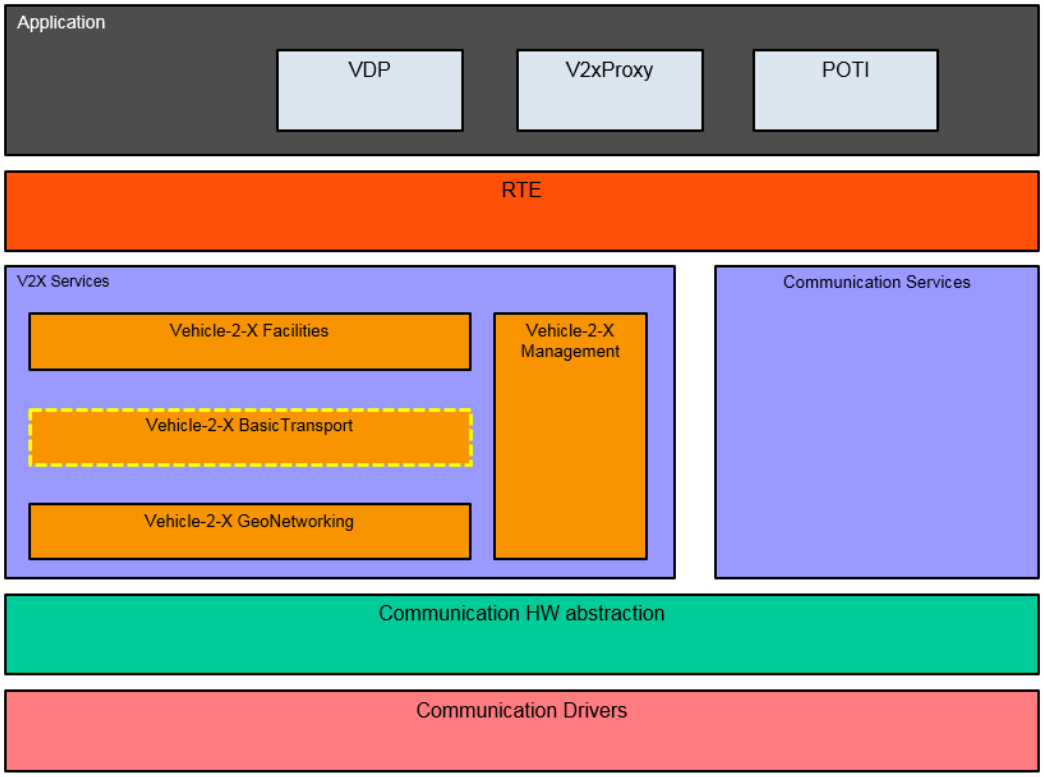

Figure 1 – AUTOSAR BSW software architecture – V2xBtp module scope

1.2 Functional overview

V2xBtp module implements the BTP protocol specified in [12], which is a lightweight protocol that requires minimal processing.

V2xBtp module resides on top of the V2xGn module [9], which implements the GeoNetworking protocol specified in [13] and [14] and below the V2xFac module specified in [8]. It multiplexes/demultiplexes messages from different processes at the V2xFac module and provides a connectionless, unreliable (i.e. packets can arrive out-of-order, appear duplicated or can be lost), end-to-end packet transport service similar to UDP [19] in the ITS ad hoc network [11]. The design of BTP assumes that entities using the protocol are either tolerant against the unreliable packet transport or provide appropriate mechanisms for reliable communication themselves. It adopts the concept of ports from the IP suite and assigns well-known ports to specific ITS facilities layer protocols. The usage of ports is similar to the two-stage packet transport in the IP protocol suite but in the case of BTP, the GeoNetworking protocol transports the packets among the ITS stations and the BTP protocol delivers the packets to the entities at the ITS facilities layer.

Upper layers can access the transport services provided by the V2xBtp module via the API primitive `V2xBtp_Transmit` by supplying the necessary data. In order to provide its packet transport services, V2xBtp module relies on the services of the V2xGn module (i.e. calls `V2XGn_Transmit`). V2xBtp module also provides services for the V2xFac module by providing message receive notification, payload and Transaction ID (generated by the V2xGn module and used for verification on demand) of the BTP packet that is received from the peer BTP entity by calling `V2xFac_RxIndication`. It also provides the callback `V2xBtp_RxIndication` to the V2xGn module so that it can get message receive notification, payload and packet ID (generated by the V2xGn module and used for verification on demand) of the GN packet that is received from the peer GeoNetworking entity. V2xBtp module can notify V2xFac module about the transmission status via the callback `V2xFac_TxConfirmation` and can get notified by the V2xGn module about the transmission status via the callback `V2xBtp_TxConfirmation`.

According to the explanations above and the specification in [12], responsibilities of the V2xBtp module can be summarized as follows:

Multiplexes messages from different processes at the V2xFac Module, e.g. CAM and DENM from the cooperative awareness basic service [17] and the decentralized environmental notification basic service [18] for the transmission of packets via the V2xGn module as well as de-multiplexing at the destination BTP protocol entity.

Accepting message transmit requests along with protocol control information from the upper layer (V2xFac module) and transmitting the data using the services of the V2xGn module.
Providing message receive notification, payload and Transaction ID of the received packet (generated by the V2xGn module and used for verification on demand) to the upper layer (i.e. V2xFac module) when a BTP packet is received.

V2xBtp can also notify the upper layer about the transmission status of sent packets.

2 Acronyms and abbreviations

The following acronyms and abbreviations have a local scope and are therefore not contained in the AUTOSAR glossary [4].

Abbreviation / Acronym:	***Description:***
BTP	Basic Transport Protocol
BTP-A	BTP header type denoting interactive packet transport
BTP-B	BTP header type denoting non-interactive packet transport
CAM	Cooperative Awareness Message
DENM	Decentralized Environmental Notification Message
DET	Default Error Tracer
GN	GeoNetworking
ITS	Intelligent Transport System
IP	Internet Protocol
PDU	Protocol Data Unit
UDP	User Datagram Protocol

3 Related documentation

3.1 Input documents

[1] AUTOSAR Layered Software Architecture
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

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

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

[4] Glossary
AUTOSAR_TR_Glossary

[5] Specification of Ethernet Interface
AUTOSAR_SWS_EthernetInterface.pdf

[6] Specification of ECU State Manager
AUTOSAR_SWS_ECUStateManager.pdf

[7] Specification of Default Error Tracer
AUTOSAR_SWS_DefaultErrorTracer.pdf

[8] Specification of Vehicle-2-X Facilities
AUTOSAR_SWS_V2XFacilities.pdf

[9] Specification of Vehicle-2-X Geo Networking
AUTOSAR_SWS_V2XGeoNetworking.pdf

[10] Specification of Module V2X Management
AUTOSAR_SWS_V2XManagement.pdf

3.2 Related standards and norms

[11] Intelligent Transport Systems (ITS); Communications Architecture
ETSI EN 302 665 V1.1.1 (2010-09)

[12] Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 5: Transport Protocols; Sub-part 1: Basic Transport Protocol
ETSI EN 302 636-5-1 V1.2.1 (2014-08)

[13] Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality
ETSI EN 302 636-4-1 V1.2.1 (2014-07)

[14] Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 2: Media-dependent functionalities for ITS-G5
ETSI TS 102 636-4-2 V1.1.1 (2013-10)

[15] Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 1: Requirements
ETSI EN 302 636-1 (2014-04)

[16] Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 2: Scenarios
ETSI EN 302 636-2 (2013-11)

[17] Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications;
Part 2: Specification of Cooperative Awareness Basic Service
ETSI EN 302 637-2 V1.3.2 (2014-11)

[18] Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications;
Part 3: Specifications of Decentralized Environmental Notification Basic Service
ETSI EN 302 637-3 V1.2.2 (2014-11)

[19] IETF RFC 768
http://tools.ietf.org/html/rfc768

3.3 Related specification

AUTOSAR provides a General Specification on Basic Software (SWS BSW General) [3] which is also valid for V2xBtp.

Thus, the specification SWS BSW General [3] shall be considered as additional and required specification for V2xBtp.

4 Constraints and assumptions

4.1 Limitations

- Wireless Communication supports IEEE 802.11p only. Other 802.11 standards (e.g. for infrastructure networks and integration with TCP/IP) can be extended in future releases of the AUTOSAR standard.
- The V2X modules follow the guidance regarding the Day-1 scenarios defined by the Basic System Standards Profile from Car-2-Car-Consortium.
- AUTOSAR R4.3.0 only focuses on the European version of car-to-car communication as defined by ETSI. Extension to other regions are planned for future releases of the AUTOSAR standard.

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 the V2xBtp module to other modules within the AUTOSAR basic software architecture. It outlines the modules that are required or optional for the realization of the V2xBtp module and the V2xBtp services that these modules use.

5.1 AUTOSAR DET (Default Error Tracer)

In development mode, the V2xBtp module reports errors through the Det_ReportError function of the DET Module [7].

5.2 AUTOSAR EcuM (Ecu State Manager)

The EcuM [6] initializes the V2xBtp module by calling `V2xBtp_Init` specified in 8.3.1.

5.3 AUTOSAR V2xFac (Vehicle-2-X Facilities)

The API used by the V2xBtp module consists of notification services as basic agents for the transfer of BTP related data to the target upper layer namely the V2xFac Module. The callback services called by the V2xBtp module are declared and placed inside the V2xFac module. These callbacks provide receive indication (`V2xFac_RxIndication`, see **[SWS_V2xBtp_00015]**) and transmit confirmation (`V2xFac_TxConfirmation`, see **[SWS_V2xBtp_00020]**) services for the V2xFac Module.

5.4 AUTOSAR V2xGn (Vehicle-2-X GeoNetworking)

The V2xBtp module assumes a transmit request primitive (`V2xGn_Transmit` [9], see **[SWS_V2xBtp_00017]**) to be provided by the V2xGn module.

The V2xBtp module provides notification services used by the V2xGn module. These callbacks called by the V2xGn module are declared and placed inside the V2xBtp module. These callbacks provide receive indication (`V2xBtp_RxIndication` specified in 8.4.1, see **[SWS_V2xBtp_00016]**) and transmit confirmation (`V2xBtp_TxConfirmation` specified in 8.4.2, see **[SWS_V2xBtp_00021]**) services for the V2xBtp module.

5.5 File structure

5.5.1 Header file structure

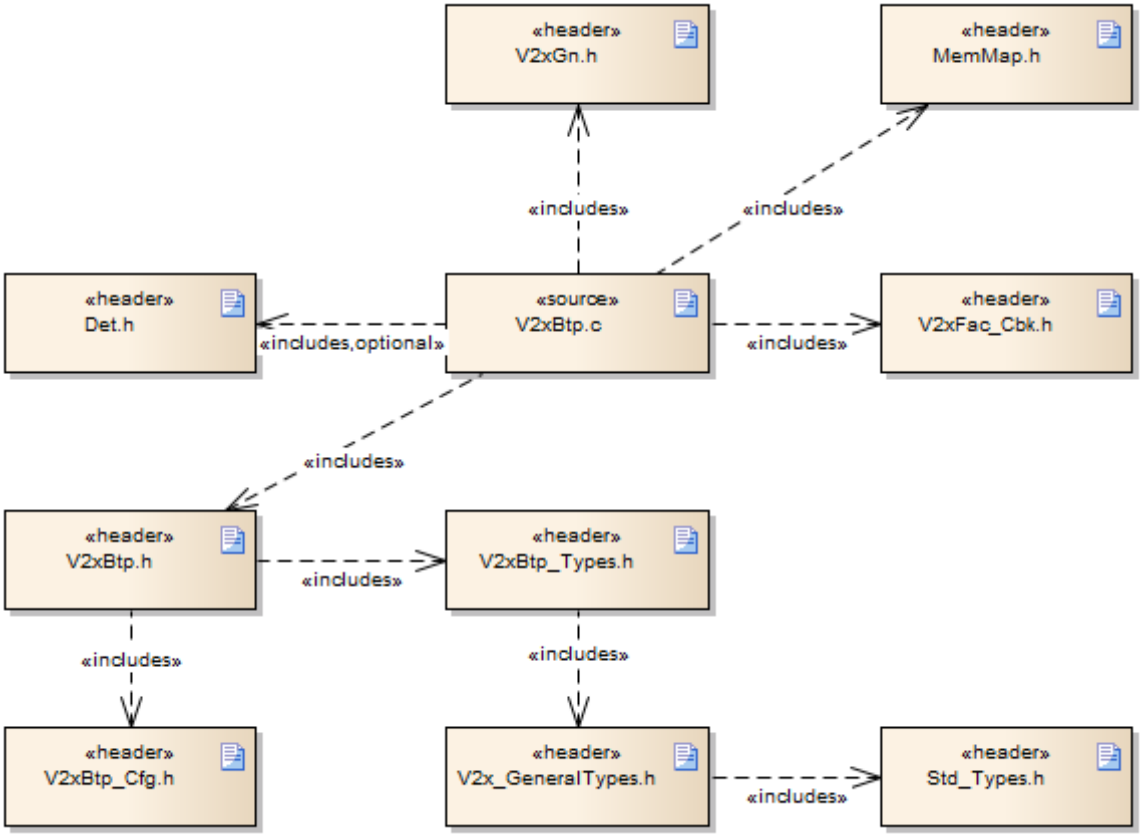

Figure 2 –V2xBtp header file structure

[SWS_V2xBtp_00001] ⌈ General V2xBtp module definitions shall be defined in V2xBtp.h.
⌋ (SRS_BSW_00348, SRS_BSW_00353, SRS_BSW_00381, SRS_BSW_00415)

[SWS_V2xBtp_00002] ⌈. Type definitions of the V2xBtp module shall be defined in V2xBtp_Types.h. ⌋ ( )

6 Requirements traceability

Requirement	Description	Satisfied by
SRS_BSW_00345	BSW Modules shall support pre-compile configuration	SWS_V2xBtp_00036
SRS_BSW_00348	All AUTOSAR standard types and constants shall be placed and organized in a standard type header file	SWS_V2xBtp_00001
SRS_BSW_00353	All integer type definitions of target and compiler specific scope shall be placed and organized in a single type header	SWS_V2xBtp_00001
SRS_BSW_00381	The pre-compile time parameters shall be placed into a separate configuration header file	SWS_V2xBtp_00001
SRS_BSW_00415	Interfaces which are provided exclusively for one module shall be separated into a dedicated header file	SWS_V2xBtp_00001
SRS_V2X_00631	The V2X system shall support an ETSI compliant Basic Transport Protocol	SWS_V2xBtp_00011, SWS_V2xBtp_20273, SWS_V2xBtp_20274

Note:
Requirement IDs within this document have an encoding to state where each requirement has its origin:

- SWS items starting with a leading 0 (SWS_V2xM_0xxxx) are module specific and not inherited.
- SWS items starting with a leading 2 (SWS_V2xM_2xxxx) are inherited from C2C-CC Basic System Profile

7 Functional specification

7.1 General Functionality

[SWS_V2xBtp_00011] ⌈ The V2xBtp module shall implement the Basic Transport Protocol (BTP) as specified in [12], unless specified otherwise in this document ⌋ (SRS_V2X_00631)

[SWS_V2xBtp_00012] ⌈ The BTP protocol shall meet the BTP related requirements specified in [15] and support the scenarios specified in [16]. ⌋ ( )

[SWS_V2xBtp_00013] ⌈ The V2xBtp Module shall encapsulate the payload from the V2xFac module with a BTP B header only as specified in [12] chapter 6 with the header format and structure as specified in chapter 7. ⌋ ( )

7.2 Message Reception

[SWS_V2xBtp_00015] ⌈ The V2xBtp module shall call `V2XFac_RxIndication()` callback of the V2xFac module to pass the payload, the GeoNetworking parameters, Transaction ID (generated by the V2xGn Module and used for verification on demand) of a received packet to the V2xFac module. This callback shall be triggered by the V2xBtp module once a BTP packet is received from the peer BTP entity. ⌋ ( )

[SWS_V2xBtp_00016] ⌈ The V2xBtp module shall get the data (BTP packet), the GeoNetworking parameters and the Transaction ID (generated by the V2xGn module and used for verification on demand) of a received GeoNetworking packet via the `V2xBtp_RxIndication()` callback. This callback shall be triggered by the V2xGn module once a GN packet is received from the peer GN entity. ⌋ ( )

7.3 Message Transmission

[SWS_V2xBtp_00017] ⌈ The V2xBtp module shall provide the API `V2xBtp_Transmit()` to service transmit requests from the V2xFac module. ⌋ ( )

[SWS_V2xBtp_00018] ⌈ Source protocol operations when `V2xBtp_Transmit()` is triggered shall be as specified in [12] chapter 8.2. ⌋ ( )

[SWS_V2xBtp_00019] ⌈ The V2xBtp module transmits BTP packets to a peer BTP entity using the `V2xGn_Transmit()` API provided by the V2xGn module. The parameters for `V2xGn_Transmit()` shall be set according to [12] chapter 8.2. The parameters are provided to the V2xBtp module by the V2xFac module with a call to `V2xBtp_Transmit()`.⌋ ( )

[SWS_V2xBtp_00020] ⌈ The V2xBtp module shall provide information about the status of the transmission of the data with the associated Transaction ID (generated by the V2xFac module and handed down to track the status of the packet) to the V2xFac Module via the `V2xFac_TxConfirmation()` callback.⌋ ( )

[SWS_V2xBtp_20273] ⌈ The V2xBtp module shall employ BTP-B headers. Consequently, the GeoNetworking common header shall use a value of 2 for the NH field. ⌋ (SRS_V2X_00631)

[SWS_V2xBtp_20274] ⌈
The V2xBtp module shall set the destination port info field to the value 0. ⌋ (SRS_V2X_00631)

7.4 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.

7.4.1 Development Errors

[SWS_V2xBtp_00023] The following table lists development errors that shall be distinguished by the V2xBtp module. V2xBtp shall report them to the DET, if development error detection is enabled.
⌈

Type of error	*Related error code*	*Value [hex]*
API service called with invalid parameter	V2XBTP_E_PARAM	0x01
API service called with invalid pointer	V2XBTP_E_PARAM_POINTER	0x02
API service used without module initialization	V2XBTP_E_UNINIT	0x03
V2xBtp initialization failed	V2XBTP_E_INIT_FAILED	0x04

⌋ ( )

7.4.2 Runtime Errors

There are no runtime errors.

7.4.3 Transient Faults

There are no transient faults.

7.4.4 Production Errors

There are no production errors.

7.4.5 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:

[SWS_V2xBtp_00024] ⌈

Module	*Imported Type*
Std_Types	Std_ReturnType
	Std_VersionInfoType
V2xFac	V2xFac_RxParamsType
V2xGn	V2xGn_TxParamsType
V2x_GeneralTypes	V2x_GnAddressType
	V2x_GnDestinationAreaType
	V2x_GnDestinationType
	V2x_GnLongPositionVectorType
	V2x_GnPacketTransportType
	V2x_GnTxResultType
	V2x_GnUpperProtocolType
	V2x_SecProfileType
	V2x_SecReportType
	V2x_TrafficClassIdType

⌋ ()

8.2 Type definitions

8.2.1 V2xBtp_ConfigType

[SWS_V2xBtp_00025] ⌈

Name:	`V2xBtp_ConfigType`	
Type:	`Structure`	
Range:	`implementation specific`	The content of the configuration data structure is implementation specific.
Description:	Configuration data structure of the V2xBtp module.	

⌋ ()

8.2.2 V2xBtp_TxParamsType

[SWS_V2xBtp_00027] ⌈

Name:	`V2xBtp_TxParamsType`		
Type:	`Structure`		
Element:	`V2x_GnUpperProtocolType`	`upperProtocol`	Specifies whether the BTP is interactive (BTP-A) or non-interactive (BTP-B).
	`uint16`	`destinationPort`	Identifies the protocol entity at the ITS facilities layer at the destination of a BTP packet.

	V2x_GnPacketTransportType	transportType	Packet transport type (GeoUnicast, SingleHopBroadcast, TopologyScopedBroadcast, GeoBroadcast, GeoAnycast).
	V2x_GnAddressType	destinationAddress	Destination address for GeoUnicast.
	V2x_GnDestinationAreaType	destinationArea	Destination area for GeoBroadcast/GeoAnycast
	V2x_GnDestinationType	destinationType	Select which destination type (destinationAddress or destinationArea is used for this packet).
	V2x_SecProfileType	secProfile	Security Profile Information (Determines the security service to invoke).
	uint16	maxPacketLifetime	Maximum tolerable time in [s] a GeoNetworking packet can be buffered until it reaches its destination.
	V2x_TrafficClassIdType	trafficClassId	Traffic class id for the message.
Description:	Structure containing BTP and GeoNetworking related parameters that should be passed to the transmit request function (i.e. V2xBtp_Transmit defined in 8.3.3) by the V2xFac module.		

] ()

8.2.3 V2xBtp_RxParamsType

[SWS_V2xBtp_00028] [

Name:	V2xBtp_RxParamsType		
Type:	Structure		
Element:	V2x_GnUpperProtocolType	upperProtocol	The protocol entity that processes the service primitive (BTP-A, BTP-B).
	V2x_GnPacketTransportType	packetTransportType	Packet transport type of the received packet.
	V2x_GnAddressType	destinationAddress	Destination address for GeoUnicast packet
	V2x_GnDestinationAreaType	destinationArea	Destination area for GeoBroadcast/GeoAnycast packet.
	V2x_GnDestinationType	destinationType	Select which destination type (destinationAddress or destinationArea is used for this packet).
	V2x_GnLongPositionVectorType	sourcePositionVector	Geographical position for the source of the received GeoNetworking packet.
	V2x_SecReportType	securityReport	Result information from the security operations for decryption and verification. This parameter is supplied by the V2xM module and forwarded up to the ITS

			Facilities layer passing through the GeoNetworking and BTP layers.
	uint64	certificateId	Identification of source certificate, for example the certificate hash. This parameter is supplied by the V2xM and forwarded up to the ITS Facilities layer passing through the GeoNetworking and BTP layers.
	uint8[4]	SspBits	Sender permissions.
	uint8	SspLength	Sender permissions length
	V2x_TrafficClassIdType	trafficClass	Traffic class, with which the GeoNetworking packet was generated by the source.
	uint16	remPacketLifetime	Remaining lifetime of the packet in [s].
	uint8	remHopLimit	Remaining hop limit of the packet.
Description:	Structure containing BTP and GeoNetworking parameters related to a received BTP packet. This structure should be passed to the V2xBtp module from the V2xGn Module when the V2xGn module receives a GN packet from the peer GeoNetworking entity.		

] ()

8.3 Function definitions

8.3.1 V2xBtp_Init

[SWS_V2xBtp_00029] ⌈

Service name:	V2xBtp_Init

Syntax:	`void V2xBtp_Init(`
	`  void`
	`)`

Service ID[hex]:	0x01
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	Initializes the V2xBtp module.

] ()

8.3.2 V2xBtp_GetVersionInfo

[SWS_V2xBtp_00030] ⌈

Service name:	V2Btp_GetVersionInfo	
Syntax:	`void V2Btp_GetVersionInfo( Std_VersionInfoType* VersionInfoPtr )`	
Service ID[hex]:	0x02	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	VersionInfoPtr	Pointer to where to store the version information of this module.
Return value:	None	
Description:	Returns the version information of this module.	

] ()

[SWS_V2xBtp_00041] [
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 V2XBTP_E_PARAM_POINTER.]()

8.3.3 V2xBtp_Transmit

[SWS_V2xBtp_00031] [

Service name:	V2xBtp_Transmit	
Syntax:	`Std_ReturnType V2xBtp_Transmit( uint16 TransactionId16, const V2xBtp_TxParamsType* TransmitParams, uint16 Length, const uint8* DataPtr )`	
Service ID[hex]:	0x03	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TransactionId16	ID identifying the payload to be transmitted. This ID is generated by the V2xFac module and is used later to indicate the status of the transmission of the message having this ID to the V2xFac module.
	TransmitParams	Structure containing all the BTP and GeoNetworking parameters used for the transmit request.
	Length	Length of the data pointed by DataPtr.
	DataPtr	Payload of the BTP packet to be transmitted
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Transmit request has been accepted. E_NOT_OK: Transmit request has not been accepted.
Description:	This API is called by the V2xFac module to request sending a BTP-PDU to the peer BTP entity.	

] ()

[SWS_V2xBtp_00042] [
If development error detection is enabled: the function shall check that the service V2xBtp_Init was previously called. If the check fails, the function shall raise the

development error V2XBTP_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK.](SWS_V2xBtp_00042)

[SWS_V2xBtp_00043] [
If development error detection is enabled: the function shall check the parameter TransmitParams for being valid. If the check fails, the function shall raise the development error V2XBTP_E_PARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.](SWS_V2xBtp_00043)

[SWS_V2xBtp_00044] [
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 V2XBTP_E_PARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.](SWS_V2xBtp_00044)

8.4 Call-back notifications

8.4.1 V2xBtp_RxIndication

[SWS_V2xBtp_00032] [

Service name:	V2xBtp_RxIndication	

Syntax:	`void V2xBtp_RxIndication(`
	`  uint32 TransactionId32,`
	`  const V2xBtp_RxParamsType* ReceiveParams,`
	`  uint16 Length,`
	`  const uint8* DataPtr`
	`)`	

Service ID[hex]:	0x04	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TransactionId32	ID of the received packet. This ID is created in the V2xGn module and handed up in the protocol stack to be used for verification on demand.
	ReceiveParams	Structure containing all the BTP and GeoNetworking parameters of a received BTP packet.
	Length	Length of the data pointed by dataPtr.
	DataPtr	Payload of the received GeoNetworking packet (BTP-PDU).
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Via this API, the V2xBtp module gets the data (BTP-PDU), the GeoNetworking parameters and the Transaction ID of a received GeoNetworking packet.	

] ()

[SWS_V2xBtp_00046] [
If development error detection is enabled: the function shall check that the service V2xBtp_Init was previously called. If the check fails, the function shall raise the development error V2XBTP_E_UNINIT.](SWS_V2xBtp_00046)

[SWS_V2xBtp_00047] [

If development error detection is enabled: the function shall check the parameter ReceiveParams for being valid. If the check fails, the function shall raise the development error V2XBTP_E_PARAM_POINTER.](RS_BRF_01784)

[SWS_V2xBtp_00048] [
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 V2XBTP_E_PARAM_POINTER.](RS_BRF_01784)

8.4.2 V2xBtp_TxConfirmation

[SWS_V2xBtp_00033] [

Service name:	V2xBtp_TxConfirmation	

Syntax:	`void V2xBtp_TxConfirmation(`
	`  uint16 TransactionId16`
	`)`	

Service ID[hex]:	0x05	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TransactionId16	ID identifying the payload (i.e. FAC-PDU) which is confirmed.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	By this API primitive, the V2xBtp module gets an indication from the V2xGn module about the status of the transmission of the data (FAC-PDU) with the associated ID.	

] ()

[SWS_V2xBtp_00050] [
If development error detection is enabled: the function shall check that the service V2xBtp_Init was previously called. If the check fails, the function shall raise the development error V2XBTP_E_UNINIT.](RS_BRF_01784)

8.4.3 V2xBtp_CopyTxData

[SWS_V2xBtp_00052] [

Service name:	V2xBtp_CopyTxData	

Syntax:	`Std_ReturnType V2xBtp_CopyTxData(`
	`  uint16 TransactionId16,`
	`  uint8* DestPtr,`
	`  uint16* Length`
	`)`	

Service ID[hex]:	0x06	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different TransactionId16	
Parameters (in):	TransactionId16	ID identifying the payload to be transmitted. This ID is used to get the respective transmit data (BTP header and payload from the V2xBtp module)
	DestPtr	Pointer to the buffer where the BTP module shall copy the data

		to.
Parameters (inout):	Length	In direction: Maximum length available for data to be copied. Out direction: The actual length of the copied data.
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Data has been copied E_NOT_OK: Length does not match
Description:	This API is called by the V2xGn module to request the V2xBtp module to copy the transmission data to a specific location.	

] ()

[SWS_V2xBtp_00053] [
If development error detection is enabled: the function shall check that the service V2xBtp_Init was previously called. If the check fails, the function shall raise the development error V2XBTP_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK.]()

[SWS_V2xBtp_00054] [
If development error detection is enabled: the function shall check the parameter DestPtr for being valid. If the check fails, the function shall raise the development error V2XBTP_E_PARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.]()

[SWS_V2xBtp_00057] [
If development error detection is enabled: the function shall check the parameter Length for being valid (Pointer is not NULL_PTR and value matches to data length). If the check fails, the function shall raise the development error V2XBTP_E_PARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.]()

Note: The function requires previous transmission request for the given TransactionId16 via the API V2xBtp_Transmit.

8.5 Expected Interfaces

8.5.1 Mandatory Interfaces

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

[SWS_V2xBtp_00034] [

API function	***Description***
V2xFac_RxIndication	This API primitive is called by the V2xBtp module providing the data and the GeoNetworking parameters of a received BTP packet to V2xFac module.
V2xGn_Transmit	Is called by V2x_Btp to send a message.

] ()

8.5.2 Optional interfaces

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

[SWS_V2xBtp_00035] ⌈

API function	*Description*
Det_ReportError	Service to report development errors.
V2xFac_TxConfirmation	By this API primitive the V2xFac module gets a confirmation that the V2X message with a certain ID was send successfully.

⌋ ()

8.5.3 Configurable Interfaces

8.5.3.1 <User>_RxIndication

[SWS_V2xBtp_00056] ⌈

Service name:	<User>_RxIndication	

Syntax:	`void <User>_RxIndication(`
	`  uint32 TransactionId32,`
	`  const V2xBtp_RxParamsType* ReceiveParams,`
	`  uint16 Length,`
	`  const uint8* DataPtr`
	`)`	

Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TransactionId32	ID of the received packet. This ID is created in the V2xGn module and handed up in the protocol stack to be used for verification on demand.
	ReceiveParams	Structure containing all the BTP and GeoNetworking parameters of a received BTP packet.
	Length	Length of the data pointed by dataPtr.
	DataPtr	Payload of the received GeoNetworking packet (BTP packet).
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Via this API, the BTP upper layer modules gets the data (BTP-SDU), the GeoNetworking parameters and the Transaction ID of a received BTP packet.	

⌋ ()

[SWS_V2xBtp_00058] ⌈
The callback function shall be configurable by the configuration parameter: V2xBtpRxIndicationFunction. ⌋()

9 Sequence diagrams

The following sequence diagrams show the interactions between the V2xBtp module and its lower and upper modules (V2xGn and V2xFac respectively).

Please note that the sequence diagrams are an extension for illustrational purposes to ease understanding of the specification and to support the functional specification described in chapter 7 and API specification described in chapter 8.

Note that all parameters and return types are left out to make the diagrams easier to read and understand.

9.1 Transmit Request / Transmit Confirmation

Figure 3 shows transmission of an ITS-FPDU from the V2xFac module to V2xGn module using `V2xBtp_Transmit`. When V2xBtp gets a transmission request from the upper V2xFac module, it will validate the input parameters, construct a BTP packet by adding a BTP header to the payload it received from the V2xFac module (i.e. ITS-FPDU), then calls `V2xGn_Transmit` to request sending a GeoNetworking packet setting the parameters for V2xGn_Transmit as specified in [12] clause 8.2. The transmission status of the sent packets can be indicated to the upper layers via the TxConfirmation callbacks in the respective module by specifying the transaction ID of the packet for which a confirmation is being communicated and the corresponding confirmation result.

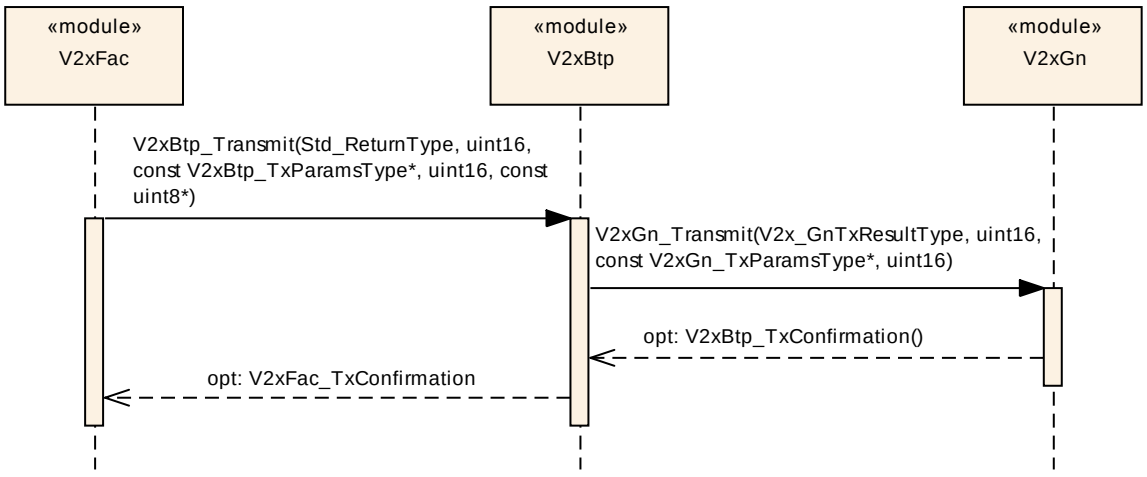

Figure 3 – Transmission request to V2xBtp

9.2 Receive Indication

As shown in

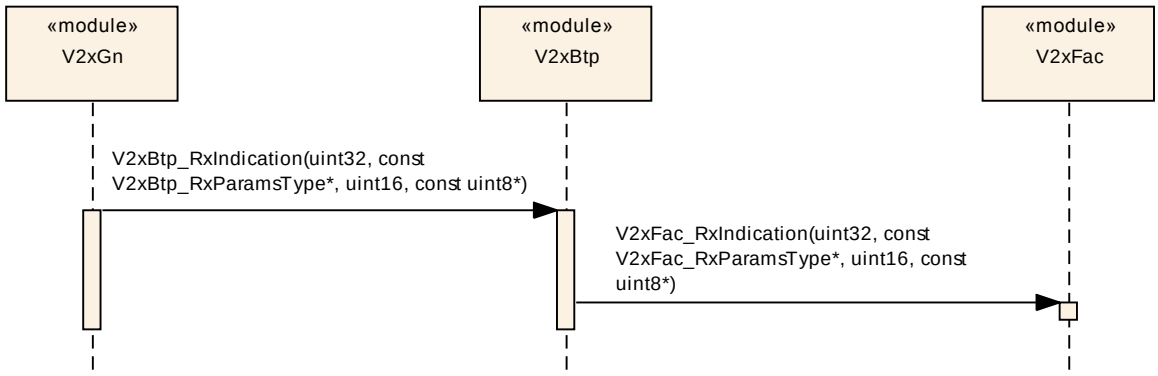

Figure 4, the reception of a BTP packet is indicated to the V2xBtp module by the V2xGn module by via a call to `V2xBtp_RxIndication`. V2xBtp module then calls the corresponding receive indication of the V2xFac module to signal a reception to the upper layer. Packet payload, GeoNetworking related parameters pertaining to the received packet and a handle to be used for verification on demand is the information that is passed to the upper layers through the use of RxIndication primitives.

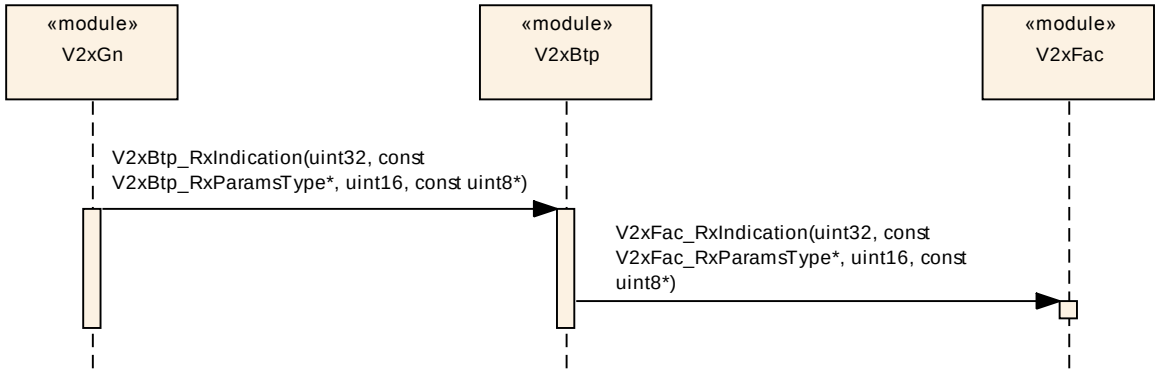

Figure 4 – Receive Indication to V2xBtp

10 Configuration specification

In general, this chapter defines configuration parameters and their clustering into containers. In order to support the specification, section 10.1 describes fundamentals. It also specifies a template (table) that should be used for the parameter specification. We intend to leave section 10.1 in the specification to guarantee comprehension

Chapter 10.2 specifies the structure (containers) and the parameters of the module V2xBtp.

Chapter 10.3 specifies additionally published information of the module V2xBtp.

10.1 How to read this chapter

For details refer to the chapter 10.1 “Introduction to configuration specification” in [3].

10.2 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.

10.2.1 Variants

Currently one configuration variant is defined for the V2xBtp module:

10.2.1.1 VARIANT-PRE-COMPILE

[SWS_V2xBtp_00036] [The V2xBtp module only supports VARIANT-PRE-COMPILE] (SRS_BSW_00345)

10.2.2 V2xBtp

SWS Item	**ECUC_V2xBtp_00001 :**
Module Name	*V2xBtp*
Module Description	Configuration of the V2xBtp (Vehicle-2-X Basic Transport) module.
Post-Build Variant Support	false
Supported Config Variants	VARIANT-PRE-COMPILE

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
V2xBtpGeneral	1	This container contains the general configuration parameters of the Vehicle-2-X Basic Transport.
V2xBtpRxIndicationFunction	1..*	This container contains the RxIndication functions for V2xBtp upper layers

10.2.3 V2xBtpGeneral

SWS Item	**ECUC_V2xBtp_00002 :**
Container Name	V2xBtpGeneral
Description	This container contains the general configuration parameters of the Vehicle-2-X Basic Transport.
Configuration Parameters	

SWS Item	**ECUC_V2xBtp_00005 :**		
Name	V2xBtpDevErrorDetect		
Parent Container	V2xBtpGeneral		
Description	Switches the Default Error Tracer (Det) detection and notification ON or OFF. ▪ true: enabled (ON) ▪ false: disabled (OFF)		
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_V2xBtp_00004 :**		
Name	V2xBtpVersionInfoApi		
Parent Container	V2xBtpGeneral		
Description	Enable/disables the API for reading the version information of the V2xBtp Module. ▪ true: enabled (ON) ▪ false: disabled (OFF)		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

10.2.4 V2xBtpRxIndicationFunction

SWS Item	**ECUC_V2xBtp_00009 :**
Container Name	V2xBtpRxIndicationFunction
Description	This container contains the RxIndication functions for V2xBtp upper layers
Configuration Parameters	

SWS Item	**ECUC_V2xBtp_00008 :**		
Name	V2xBtpCddHeaderFile		
Parent Container	V2xBtpRxIndicationFunction		
Description	This parameter specifies the name of the header file containing the definition of the V2xBtpRxIndicationFunction module functions.		
Multiplicity	1		
Type	EcucStringParamDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
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_V2xBtp_00006 :**		
Name	V2xBtpDestinationPort		
Parent Container	V2xBtpRxIndicationFunction		
Description	Destination port for the <User>_RxIndication function.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	0		
Post-Build Variant Value	false		
Scope / Dependency	scope: local		

SWS Item	**ECUC_V2xBtp_00007 :**		
Name	V2xBtpRxIndicationFunctionName		
Parent Container	V2xBtpRxIndicationFunction		
Description	Name of the <User>_RxIndication callback function.		
Multiplicity	1		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

10.3 Published Information

For details refer to the chapter 10.3 “Published Information” in [3].

11 Not applicable requirements