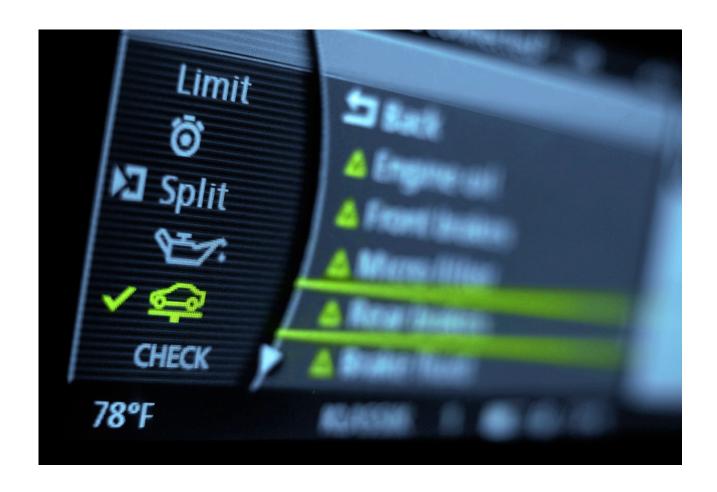# EB tresos® ECU Configuration Wizard documentation

product release 8.8.4

Elektrobit Automotive GmbH
Am Wolfsmantel 46
91058 Erlangen, Germany
Phone: +49 9131 7701 0
Fax: +49 9131 7701 6333
Email: info.automotive@elektrobit.com

Technical support

https://www.elektrobit.com/support

Legal disclaimer

Table of Contents

1. Overview of EB tresos ECU Configuration Wizard documentation

Welcome to the EB tresos ECU Configuration Wizard (ECU Configuration Wizard) product documentation.

This document provides:

► Chapter 2, "ECU Configuration Wizard release notes": release notes for the ECU Configuration Wizard module

► Chapter 3, "ECU Configuration Wizard user's guide": containing background information and instructions

2. ECU Configuration Wizard release notes

2.1. Overview

This chapter provides the ECU Configuration Wizard product specific release notes. General release notes that are applicable to all products are provided in the EB tresos AutoCore Generic documentation. Refer to the general release notes in addition to the product release notes documented here.

2.2. Scope of the release

2.2.1. Configuration tool

Your release of EB tresos AutoCore is compatible with the release of the EB tresos Studio configuration tool:

► EB tresos Studio: 28.2.0 b211016-0103

2.2.2. AUTOSAR modules

The following table lists the AUTOSAR modules that are part of this ECU Configuration Wizard release.

Module name	AUTOSAR version and revision	SWS version and revision	Module version	Supplier
No AUTOSAR modules available				

Table 2.1. Hardware-Independent Modules specified by the AUTOSAR standard

2.2.3. EB (Elektrobit) modules

The following table lists all modules which are part of this release but are not specified by the AUTOSAR standard. These modules include tooling developed by EB or they may hold files shared by all other modules.

Module name	Module version	Supplier
Configurators	2.8.38	Elektrobit Automotive GmbH

Table 2.2. Modules not specified by the AUTOSAR standard

2.2.4. MCAL modules and EB tresos AutoCore OS

For information about MCAL modules and OS, refer to the respective documentation, which is available as PDF at `$TRESOS_BASE/doc/3.0_EB_tresos_AutoCore_OS` and `$TRESOS_BASE/doc/5.0_MCAL_-modules`[1]. It is also available in the online help in EB tresos Studio. Browse to the folders `EB tresos AutoCore OS` and `MCAL modules`.

2.3. Module release notes

2.3.1. Configurators module release notes

► Module version: 2.8.38.B466224

► Supplier: Elektrobit Automotive GmbH

2.3.1.1. Change log

This chapter lists the changes between different versions.

Module version 2.8.38
2021-10-08

► Extended Transformer for Com such that it creates the `ComTimeBase` container only if one of its contained parameters can be configured

► Updated Transformer for Com to configure `ComDataInvalidAction` only for received ISignals and ISignalGroups that provide specific values in `ComDataTypePolicy`

---

[1] `$TRESOS_BASE` is the location at which you installed EB tresos Studio.

Module version 2.8.37

2021-09-17

▶ Extended Transformer for TcpIp to configure parameters in `TcpIpIpConfig`, `TcpIpUdpConfig`, and `TcpIpTcpConfig`

Module version 2.8.36

2021-08-20

▶ Extended SdTransformer to configure `SdSubscribeEventgroupRetryDelay`, `SdSubscribeEventgroupRetryMax`, `SdVersionDrivenFindBehavior`, and `SdBlacklistedVersions`

▶ Extended Transformers for CanTSyn, FrTSyn, and StbM to configure `CanTSynGlobalTimeNetworkSegmentId`, `FrTSynGlobalTimeNetworkSegmentId`, and `StbMTimeValidationRecordTableBlockCount`

Module version 2.8.35

2021-06-25

▶ Extended ComTransformers for Tls and Csm to configure the TLS cipher suites `TLS_ECDHE_ECDSA_WITH_NULL_SHA` and `TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256`

▶ Extended DcmTransformer to configure DcmDspDidRead and DcmDspDidWrite subcontainers if either `DiagnosticServiceInstance` or `DiagnosticServiceClass` references a valid `DiagnosticAccessPermission`

Module version 2.8.34

2021-05-28

▶ Extended ComTransformer for EthIf to configure `EthIfPortStartupActiveTime`

▶ ASCCONFIGURATORS-1215 Fixed known issue: J1939Rm Transformer configures identical short names for `J1939RmUpperComIPdu` and `J1939RmLowerComIPdu`

Module version 2.8.33

2021-04-30

▶ Add support for `ConsumedServiceInstances` in multicast `SocketAddress` elements

▶ Internal module performance improvement. This module version update does not affect module functionality

▶ ASCCONFIGURATORS-1208 Fixed known issue: Com Transformer calculates incorrect ComSignalType when dataTypePolicy is set to networkRepresentationFromComSpec

▶ ASCCONFIGURATORS-1205 Fixed known issue: Dcm Transformer uses potentially invalid values to configure `DcmDspReadMemoryRangeInfo` and `DcmDspWriteMemoryRangeInfo` container creation

▶ Added Support AUTOSAR SignalLength SDG for the configuration of ComBitSize

▶ ASCCONFIGURATORS-1206 Fixed known issue: Dcm Transformer uses potentially invalid values to configure `DcmDspDidFreezeCurrentState, DcmDspDidResetToDefault` and `DcmDspDidShortTermAdjustment`

▶ ASCCONFIGURATORS-1207 Fixed known issue: Dcm Transformer uses potentially invalid values to configure `DcmDspVehInfoDataOrder` and `DcmDspVehInfoDataSize`

▶ Extended DcmTransformer to configure `DcmDemClientRef` for `DcmDslProtocolRow`

Module version 2.8.32

2021-03-05

▶ Adapted ComTransformer for EthTSyn with fixed GlobalTimeDomain and Port Structure

▶ Extended ComTransformer for Eth to support the configuration according to the AUTOSAR 4.3 schema

▶ Extended ComTransformer for SecOC to support the configuration of SecOCAuthPduHeaderLength

▶ Extended DcmTransformer to configure `DcmDspDataEndianness` for `DcmDspData`

▶ ASCCONFIGURATORS-1189 Fixed known issue: Dem Transformer stops with an error when processing `DiagnosticDataElements` with identical names

Module version 2.8.31

2021-02-12

▶ ASCCONFIGURATORS-1178 Fixed known issue: Dem Transformer uses potentially invalid value to configure `DemAgingCycleCounterThreshold`

▶ ASCCONFIGURATORS-1177 Fixed known issue: Dem Transformer uses potentially invalid Id field to configure `DemDidIdentifier`

▶ Extended transformer for `SecOC` to support the authentication of `DoIP` and `SomeIpTp` Sdus

▶ ASCCONFIGURATORS-1176 Fixed known issue: Dcm Transformer uses a potentially invalid ID field for `DcmDspMemoryIdInfo` container creation

Module version 2.8.30

2021-01-22

▶ Added transformer for `Tls`

Module version 2.8.29

2020-12-18

▶ Added transformer for `EcuM` to support the configuration of synchronous PNC wake-up

▶ Extended transformer for `SoAd` to consider two remote `SocketAddress` elements as distinct if at least one of them has a dynamic port assigned

▶ Extended transformer for `TcpIp` to configure the `TcpIpIpSecConfig` container

▶ Extended transformers for `TcpIp` and `Sd` to stop accessing the `EthernetCommunicationConnector` to `NetworkEndpoint` references since they will become obsolete in the future

Module version 2.8.28

2020-10-23

▶ Extended transformers for `Dcm` to support the multiple `DiagnosticServiceTable` and to configure `DcmDspMaxDidToRead` parameter for `DiagnosticReadDataByIdentifier`

▶ Added transformers for `J1939Tp`, `J1939Nm`, `J1939Rm`, and `J1939Dcm`

▶ Extended transformers for `Dcm` to support Parallel Processing for `OBD` and `UDS`

Module version 2.8.27

2020-09-25

▶ Extended transformers for `SoAd` and to `SoAd` to support the Ethernet service and communication model of the AUTOSAR R19-11 release

▶ ASCCONFIGURATORS-1151 Fixed known issue: Transformer for `Sd` does not create all `SdServerService/SdClientService` containers

▶ Extended transformer for `EthTSyn` to create one additional `EthTSynPortConfig` container if the configured ECU instance contains a switch

▶ Extended transformers for `TcpIp` and `SoAd` to configure `TcpIpIpFragmentationRxEnabled`, `TcpIpIpFragmentationTxEnabled`, and `SoAdSocketUdpRetryEnabled` depending on the MTU that is configured at the `EthernetCommunicationConnector`

▶ Extended transformer for `Com` such that no `ComSignal` containers are created for `ISignals` that have a bit offset assigned which does not lie in the user data area of its `NmPdu`

Module version 2.8.26

2020-07-31

▶ Extended transformer for `FrTp` to support the configuration of a single `FrTpConnection` container using two `FlexrayTpConnection` elements

▶ Extended transformer for `Com` to support the configuration of `ComSignalType` for signals of type `ARRAY` that have an `SwBaseType` without valid `length` assigned

Module version 2.8.25

2020-06-19

▶ Extended Transformer for `Sd` to support the configuration of `SdMulticastEventSoConRef` even if the `SoAdSocketConnectionGroup` container of the `SdEventHandler` contains more than one multicast `SoAdSocketConnection`

▶ Extended Transformer for `PduR` to support the simultaneous routing and local reception of `DoIp` SDUs that are received via functional addressing or are related to a `DiagnosticConnection`

▶ Added Transformer for the `Csm` module

▶ Internal module improvement. This module version update does not affect module functionality

▶ Extended transformer for `DoIp` to configure `DoIpChannel` containers for `DoIpTpConnection` elements that do not contain an `ident` sub element

▶ ASCCONFIGURATORS-1125 Fixed known issue: `DcmDslProtocolID` is not configured for `DoIpTpConnections`

▶ ASCCONFIGURATORS-1124 Fixed known issue: Empty `DemExtendedDataClass` containers are configured

Module version 2.8.24

2020-05-22

▶ Extended Transformer for `SecOC` to support the configuration of `SecOCMessageLinkLen` and `SecOCMessageLinkPos`

▶ Extended Transformers for `EcuC`, `CanIf`, `CanNm`, `SoAd` and `TcpIp` to consider `IPduMapping.pduMaxLength` for the calculation of the payload length of a PDU

▶ Extended Transformer for `IpduM` to configure an Rx PDU fan-in for contained PDUs if they have the same header-id values assigned and their `ContainerIPdus` are configured as `AcceptAll`

▶ Extended Transformer for `TcpIp` to support the configuration of `TcpIpIpV4PathMtuTimeout`, `TcpIpIpV4PathMtuEnabled`, `TcpIpIpV6PathMtuTimeout` and `TcpIpIpV6PathMtuEnabled`

▶ ASCCONFIGURATORS-1114 Fixed known issue: During the import of the Diagnostic Extract, the `DiagnosticEvent` and `DiagnosticAging` are in common namespace and the duplicate names are not appended with a suffix "_1" to the short name

Module version 2.8.23

2020-04-27

▶ Extended Transformers for `FrIf` and `FrArTp` to support merging of `FrArTpConnection` elements with the same local/remote Tp address pair and to support the configuration of one and the same Tx N-PDU in different slots

▶ Extended Transformer for `Com` to support the configuration of PDUs that are received via `SecOC` and TP API

▶ Extended Transformer for `Com` to support the configuration of `ComDataInvalidAction`

Module version 2.8.21

2020-02-21

▶ Extended Transformer for `FrArTp` to add N-PDUs to `FrArTpChannel` containers only if they are actually sent or received by the Ecu Instance

▶ ASCCONFIGURATORS-1087 Fixed known issue: SoAdPduRoute containers are not created for Broadcast/Multicast SocketConnectionGroups

▶ Extended Transformer for `DoIp` to support the configuration of more than one `DoIpUdpConnection` for a given remote tester address

▶ Extended Transformer for `Com` to support boolean `SwBaseType` elements of arbitrary bit lengths

▶ Extended Transformer for `ComM` to support the configuration of `ComMUser` containers

▶ Extended Transformers for `Com`, `LdCom`, `SecOC`, and `SoAd` to configure TP API for PDU Triggerings that are referenced by `EthTpConnection` elements

Module version 2.8.20

2019-10-17

▶ Extended Transformer for `SomeIpTp` to support `SomeIpTpChannel` entities

▶ Extended Transformer for `TcpIp` to configure `TcpIpTcpKeepAliveEnabled`, extended Transformer for `SoAd` to configure `SoAdSocketTcpKeepAlive`

▶ Extended Transformer for `StbM` to support parameters `StbMOffsetCorrectionAdaptionInterval`, `StbMOffsetCorrectionJumpThreshold`, `StbMRateCorrectionMeasurementDuration` and `StbMRateCorrectionsPerMeasurementDuration`

▶ Extended Transformer for `LinIf` to prevent the configuration of `LinFrame` elements that the configured Ecu Instance sends or receives via `LinSlave` communication controllers

▶ ASCCONFIGURATORS-1074 Fixed known issue: Transformer for CanSM issues Null pointer exception for CommunicationConnectors that do not refer to valid Controllers

Module version 2.8.19

2019-09-06

▶ ASCCONFIGURATORS-1064 Fixed known issue: Transformer for SoAd does not create Pdu Routes for dynamically configured local IP addresses

Module version 2.8.18

2019-07-12

▶ Support array types for the `DcmDspRoutineSignalType`

▶ ASCCONFIGURATORS-1055 Fixed known issue: Transformers for Com, ComM issue a NullPointerException if PNC vector is missing

Module version 2.8.17

2019-06-14

▶ Remove prefix for names of DcmDspData elements

▶ Extended Transformer for `ComM` to configure `ComMNmVariant` as `NONE` if the configured Ecu Instance does not send or receive PDUs on the `ComMChannel`

▶ Extended Transformers for `EcuC`, `CanNm`, `FrNm`, `UdpNm`, `PduR` and `Com` to configure received Nm User Data PDUs even if the signals contained in the Nm PDUs have no signal port assigned

▶ Extended Transformer for `SoAd` to configure `GeneralPurposeIPdu` elements of category Dlt_TP and Dlt_IF

▶ Extended Transformer for `SoAd` to configure `SoAdTxPduCollectionSemantics`

▶ ASCCONFIGURATORS-1041 Fixed known issue: During the import of the Diagnostic Extract, the creation of a configuration container in Dem throws an exception. This exception is caused by an illegal container short name

Module version 2.8.16

2019-05-17

▶ Extended Transformer for `SoAd` to support Service Oriented Communication of System Description .arxml files

▶ Extended Transformer for `StbM` to configure `StbMIsSystemWideGlobalTimeMaster` for abstract Global Time Domains

▶ ASCCONFIGURATORS-1035 Fixed known issue: Transformer for LinIf configures `LinIfCollisionResolvingRef` incorrectly

▶ ASCCONFIGURATORS-1038 Fixed known issue: Dcm importer wrongly configures the `SessionRefs` and `SecurityLevelRefs` for the `ServiceInstance` value of `AccessPermissionValidity`

Module version 2.8.15
2019-04-18

▶ Extended Transformer for `Dem` to support the configuration of the `DemOBDSupport` and the `DemOBDSupportKind` parameters

▶ Adapted Transformer for `TcpIp` to skip the configuration of `TcpIpStaticIpAddressConfig` containers if the parent `TcpIpLocalAddr` container has been created for a multicast address that is configured at runtime by `Sd`

▶ Adapted Transformer documentation for `Dcm` to correctly reflect the information for DiagnosticComControl (0x28) service

▶ Adapted Transformer for `Dcm` to fill the Data Access Interface by the DEXT importer

▶ Adapted Transformer documentation for `Dcm` to write `VARIABLE-SIZE` instead of `VARIABLE-LENGTH`

▶ Extended Transformer for `CanNm` to resolve short name collisions in the case that `NmPdu` elements are transmitted in different CAN networks using identical CAN Ids or CAN Id ranges

▶ Adapted Transformer for `Dcm` to also take into account `MAPPED-FLAT-SWC-SERVICE-DEPENDENCY` when configuring the `DcmDspRoutineUsePort`

▶ Extended Transformer for `Com` to configure `ComSignalDirection` and `ComSignalGroupDirection`

▶ Extended Transformers to support the reception of one and the same PDU via different networks

Module version 2.8.13
2019-03-14

▶ ASCCONFIGURATORS-1006 Fixed known issue: Transformer for Xcp erroneously adds non-Xcp PDUs

Module version 2.8.12
2019-02-15

▶ Extended Transformer for `Dcm` to configure the `DcmDspRequestRoutineResultsIn` container inside the `Dsp` container, only if the `REQUESTS` tag exists inside the DiagnosticExtract

▶ Extended Transformer for `IpduM` to configure `IpduMContainerQueueSize` by using `ContainerIPdu.minimumRxContainerQueueSize` and `ContainerIPdu.minimumTxContainerQueueSize`

▶ Extended Transformer for `StbM`, `EthTSyn`, `CanTSyn`, and `FrTSyn` to support configuration via AUTOSAR 4.4 GlobalTimeDomain elements

▶ Extended Transformer for `IpduM` to configure `IpduMContainerTxTriggerMode` of `IpduMContainerTxPdu` to `IPDUM_DIRECT`, if the `IpduMContainerTxPdu` is itself contained in a `SecuredIPdu`, i.e. a PDU that is processed in `SecOC`

▶ Extended Transformer for `ComM` to take the `PncMapping` references to `PhysicalChannel` elements into account for setting up the `ComMPnc` lists

▶ Extended Transformer for `ComM` to support the configuration of managed `ComMChannel` containers

Module version 2.8.11
2019-02-05

▶ ASCCONFIGURATORS-996 Fixed known issue: DiagnosticEventHandler issues NullPointerException on Event that is not mapped onto an operation Cycle

▶ Extended Transformer for `Dem` to configure the `DiagnosticDemProvidedDataMapping.DataProvider` with the standardized values

▶ Extended Transformer for `IpduM` to configure `IpduMContainedTxPduPriority`

Module version 2.8.10
2019-01-25

▶ Extended configuration of `CanTSynGlobalTimeFollowUpTimeout` in Transformer for `CanTSyn`

▶ Extended Transformer for `ComM` to configure `ComMPncEthIfSwitchPortGroupRef`

▶ Extended Transformers to allow execution in EB tresos Studio 26 and to support transformation of ASR 4.4 system description files

▶ Extended Transformers for `CanIf` and `FrIf` to configure `PduR` as upper layer for routed N-PDUs

▶ Extended Transformer for `Dem` to configure the `DemDtcStatusAvailabilityMask` parameter inside the `DemGeneral` container

Module version 2.8.9
2018-12-13

▶ ASCCONFIGURATORS-977 Fixed known issue: Transformer for `EcuC` stops with an error if a sent PDU undergoes a tx fanout before SecOC/IpduM and before <Bus<If at the same time

▶ Extended Transformer for `Dcm` to configure the `DcmDspRoutineFixedLength` parameter of a `DcmDspRoutine`

▶ Extended Transformer for `Dem` to configure the `DemEventSignificance` parameter inside the `DemEventClass` container

▶ ASCCONFIGURATORS-980 Fixed known issue: Transformer for `SoAd` configures `SomeIpTp` N-PDUs as TP API PDUs instead of as IF API PDUs

▶ Extended Transformers for `IpduM`, `SecOC`, and `PduR` to support transmission of cryptographic PDUs in container PDUs and routing of cryptographic PDUs

▶ Extended Transformer for `Dem` to configure the `DemEventDestination` parameter inside the `DemEventClass` container

▶ Extended Transformer for `IpduM` to configure `IpduMContainerQueueSize`

Module version 2.8.8

2018-10-26

▶ ASCCONFIGURATORS-964 Fixed known issue: Transformer for `EcuC` does not configure Routing-only Applicative TP SDUs

▶ Extended Transformers for `Com` and `LdCom` to support the configuration of `Tp` API for PDUs that are sent or received via `SomeIpTp`

▶ Extended Transformer for `ComM` to configure `ComMPncComSignal` entries as ERA if they are associated with a `COMM_GATEWAY_TYPE_PASSIVE` ComMChannel

▶ ASCCONFIGURATORS-971 Fixed known issue: Transformer for `FrTSyn` does not configure FrTSynGlobalTimeSyncDataIDList correctly

Module version 2.8.7

2018-09-28

▶ Extended Transformer for `SoAd` to support use of "0.0.0.0" IPV4 adresses as dynamic remote IP addresses

▶ Extended Transformer for `IPduM` to support parameter`IpduMUnusedAreasDefault` in `IpduMContainerTxPdu`s

▶ ASCCONFIGURATORS-957 Fixed known issue: Transformer for `SecOC` does not process cryptographic PDUs

Module version 2.8.6

2018-08-24

▶ Removed configuration of `EthSwtPortSpeed` from Transformer for `EthSwt`

Module version 2.8.5

2018-07-31

▶ Extended Transformer for `StbM`: Added support for the configuration of `StbMSynchronizedTime-BaseIdentifier`

▶ Extended Transformers to ignore Frames, PDUs, and Signals that are sent or received via `LinSlave` elements

▶ Extended Transformer for `LdCom` to support the configuration of a received PDU irrespective of its `minimumDelay`, `transmissionModeTrueTiming` and `transmissionModeCondition`

▶ Extended Transformer for `Com` to implement OEM specific fallback for configuration of `ComSignalType`

▶ Extended Transformer for `EcuC` to exclude secured PDUs and Pdus contained in a secured PDU from the Meta-Data handling

▶ Extended Transformer for `Dcm` to configure `DiagnosticDataTransfer`

▶ ASCCONFIGURATORS-951 Fixed known issue: Transformer for `IPduM` creates duplicated Contained Rx PDUs

Module version 2.8.4

2018-06-22

▶ Removed creation of `IPV4` limited broadcast address entries in Transformer for `TcpIp`

▶ Extended Transformer for `PduR` to configure `PduRRoutingGroup` containers

▶ Extended Transformer for `SecOC` to support the configuration of distinct Secured and Cryptographic PDU containers

▶ Extended Transformer for `EcuC` to configure Meta-Data for Ethernet PDUs that are received or sent by server components

▶ Extended Transformer for `EthSwt` to configure modified parameter `EthSwtPortPhysicalLayerType` and new parameter `EthSwtPortMacLayerType` in `EthSwtPort`

Module version 2.8.3

2018-05-25

▶ Extended Transformer for `Com` to configure `ISignalIPduGroup` elements that refer to `NmPdu` elements

▶ Extended Transformers for `Com`, `SecOC`, `IpduM`, `PduR`, and `EcuC` to support tx fan-out in between these modules

▶ Extended Transformers for `Com` and `SomeIP` to support sending and receiving of `ComIPdu` elements via TP API

▶ Extended Transformer for `CanIf` to support `CAN_TSYN` as value for `CanIfRxPduUserRxIndicationUL` and `CanIfTxPduUserTxConfirmationUL`

▶ Extended Transformer for `Dcm` to configure custom sub-services for `DiagnosticEcuReset`

▶ ASCCONFIGURATORS-922 Fixed known issue: `EthIfCtrlMtu` is configured incorrectly

▶ Changed Transformer for `EthIf`: Length of VLAN tag is no longer subtracted from `EthIfCtrlMtu`

▶ Extended Transformer for `IPduM` to configure the `offset` and `updateIndicationBitPosition` parameters in `IpduMContained[Rx|Tx]Pdu`, added support for `ContainerIPduHeaderTypeEnum.no-Header`

▶ Extended Transformer for `LdCom` to support the configuration of a received PDU irrespective of its `transferProperty`

▶ Extended Transformer for `DoIp` to support `IPV6` during configuration of `VehicleAnnouncement` container

▶ Extended Transformer for `PduR` to configure `PduRTpThreshold` only if the `PduRRoutingPath` container holds exactly one `PduRDest` subcontainer

▶ Extended Transformer for `FrTSyn` to configure `FrTSynGlobalTimeSequenceCounterJumpWidth` in the context of a `FrTSynGlobalTimeSlave` container

▶ ASCCONFIGURATORS-921 Fixed known issue: `CanIfRxPduDlc` is configured incorrectly

▶ Extended Transformer for `SoAd` to configure `SoAdSocketTcpInitiate` only in TcpIp `SoAdSocketConnectionGroup` containers that contain exactly one `SoAdSocketConnection` sub container

Module version 2.8.2
2018-04-20

▶ ASCCONFIGURATORS-904 Fixed known issue: `MultiplexedIPdus` that are sent and received at the same time are configured incorrectly

▶ Extended Transformer for `Dcm` to configure references `DcmDspReadMemoryRangeSecurityLevelRef` in `DcmDspReadMemoryRangeInfo` and `DcmDspReadMemoryRangeByLabelInfo`, `DcmDspWriteMemoryRangeSecurityLevelRef` in `DcmDspWriteMemoryRangeInfo` and `DcmDspWriteMemoryRangeByLabelInfo`

▶ Extended Transformer for `Dcm` to configure `DiagnosticRequestOnBoardMonitoringTestResults`

▶ Extended Transformer for `Com` to configure `ComIPduGroupRef` only for `ComPduGroup` containers whose direction is not opposite to the referencing PDU

▶ Extended Transformer for `Dcm` to configure `DiagnosticRequestVehicleInfo`

Module version 2.8.1
2018-03-16

▶ ASCCONFIGURATORS-890 Adapted Transformer for `Dcm` to enable the `DcmProcessingConditions` container only in the case when a `DcmModeCondition` or a `DcmModeRule` exists

▶ Extended Transformer for `Dcm` to configure the `OBDMode_0x0A` (`DiagnosticRequestEmissionRelatedDTCPermanentStatus`) service

▶ Extended Transformer for `ComM` to configure the parameter `ComMPncPrepareSleepTimer`

▶ Extended Transformers for `Nm`, `UdpNm`, `CanNm`, and `FrNm` to configure the parameters `NmRepeatMsgIndEnabled`, `NmNodeDetectionEnabled`, `NmNodeIdEnabled` either by using the `NmCluster` data or by using the `NmEcu` data as a fallback

▶ Extended Transformer for `SecOC` to configure parameters `SecOCSecured[Rx|Tx]PduOffset` and `SecOCSecured[Rx|Tx]PduLength` in `SecOCRxPduSecuredArea`

▶ Extended Transformer for `StbM` to support the configuration of `StbMSynchronizedTimeBase` container for GlobalTimeDomains that are not linked to any network

▶ Extended Transformer for `SecOC` to configure parameter `SecOCAuthenticationBuildAttempts` in `SecOCRxPduProcessing` and `SecOCTxPduProcessing`

▶ Extended Transformer for `Dcm` to support the configuration of `DcmDspMemory` container for the `DiagnosticWriteMemoryByAddress` and `DiagnosticReadMemoryByAddress` services.

▶ Extended Transformer for `ComM` to also take routed PDUs into account for setting up the `PNC` to `ComMChannel` references

Module version 2.8.0

2018-02-16

▶ Extended Transformers for `DoIP` and `PduR` to not configure `DoIP` SDUs that are either sent and have their `DiagPduType` field set to `DiagRequest` or that are received and have their `DiagPduType` field set to `DiagResponse`

▶ Extended Transformer for `FrTSyn` to configure `FrTSynGlobalTimeOfsDataIDList` and `FrTSynGlobalTimeSyncDataIDList` configuration containers

▶ Extended Transformer for `Dem` to configure the `DemInternalDataElementClass` and `DemExternalCSDataElementClass` choices of the DemDataElementClass ChoiceContainer

▶ ASCCONFIGURATORS-878 Fix configuration of DcmDsdSubServiceId for the DiagnosticComControl

▶ ASCCONFIGURATORS-879 Fix configuration of Data Access Interface in the case of a DiagnosticIOControlService

▶ Extended Transformer for Dcm to configure `DiagnosticRequestDownload`, `DiagnosticRequestUpload`, `DiagnosticRequestTransferExit`

▶ Extended Transformer for `StbM` to configure parameters `StbMTimeLeapFutureThreshold`, `StbMTimeLeapPastThreshold`, and `StbMClearTimeleapCount`

▶ Extended Transformer for `SecOC` to configure parameters `SecOCAuthDataFreshnessLen`, `SecOCAuthDataFreshnessStartPosition`, `SecOCUseAuthDataFreshness`, and `SecOCSecuredRxPduVerification`

▶ Added Transformer for `SomeIpTp`

▶ Transformers for `EcuC`, `CanNm`, `FrNm`, `UdpNm`, `PduR`, `Com`, and `ComM` now configure PNC ERA PDUs and signals for ACTIVE as well as PASSIVE PNC gateways

▶ Adapted Transformer for `CanTSyn` to configure the `CanTSynGlobalTimeSequenceCounterJump-Width` parameter in `CanTSynGlobalTimeSlave`

Module version 2.7.11

2018-01-19

▶ Internal module improvement. This module version update does not affect module functionality

Module version 2.7.10

2017-12-15

▶ Extended Transformer for `Dcm` to support `DcmDspPid` configuration

▶ Extended Transformer for `Dcm` to configure the `OBDMode_0x02` (`DiagnosticRequestPowertrainFreezeFrameData`) service

▶ Extended Transformer for `Dcm` to configure the `OBDMode_0x01` (`DiagnosticRequestCurrentPowertrainDiagnosticData`) service

▶ Extended Transformer for `Dcm` to configure the `OBDMode_0x04` (`DiagnosticClearResetEmissionRelatedInfo`) service

▶ Extended DcmTransformer for `DcmDspComControl` to configure the `DcmDspComControlSubNode` sub-container

▶ Extended Transformer for `SecOC` to configure the `SecOCPduType` parameter for authentic PDUs

▶ ASCCONFIGURATORS-859 Fixed known issue: `ComTransferProperty` is configured incorrectly for `SignalGroups`

▶ Extended Transformer for `CanNm` to configure the `CanNmNodeIdEnabled` parameter for `NmEcu`s

▶ ASCCONFIGURATORS-864 Fixed known issue: `ComTimeout` and `ComFirstTimeout` are configured incorrectly for `SignalGroups`

Module version 2.7.9

2017-11-17

▶ Extended Transformer for `EthSwt` to also accept value of `CouplingPort.VlanMembership.defaultPriority` if all `defaultPriority` values of a `CouplingPort` are identical

▶ Modified the import of the Variable-size DID signals.

▶ Extend the DcmTransformer in order to also configure the DcmDspRequestResultsRoutineSupported and DcmDspStopRoutineSupported

▶ Adapted Transformer for IpduM: `IpduMContainedRxInContainerPduRef` is only configured if `RxAcceptContainedIPdu` of the associated `ContainerIPDU` is set to `ACCEPT-CONFIGURED`

▶ Adapted Transformer for SoAd: `SoAdSocketRemoteIpAddress` is now explicitly configured if `Ipv4Configuration/ipv4Address` or `Ipv6Configuration/ipv6Address` is set to `ANY`

▶ Extended Transformer for `Dcm` to configure the `OBDMode_0x08(RequestControlOfOnBoardDevice)` service

▶ Support fragmentation of PDU size bigger than MTU in `SoAdTransformer` and `TcpIpTransformer`

Module version 2.7.8

2017-10-20

▶ Extended Transformer for `Dcm` to configure the `OBDMode_0x03` and `OBDMode_0x07` services

▶ Improved handling of `ConsumedServiceInstance` elements linked to `ProvidedServiceInstance` elements which in turn are not connected to any `SocketConnection` or `SocketConnectionBundle`

▶ Modified the configuration of DcmDspDidInfo and of its subcontainers: DcmDspDidControl, DcmDspDidRead and DcmDspDidWrite

Module version 2.7.7

2017-09-22

▶ Removed workaround in the configuration of `IpduMContainerTxTriggerMode` that was required by a `SecOC` restriction which has been resolved in the meantime

▶ Added support for the configuration of `ContainerIPdu` elements that are sent and received by the same `EcuInstance`

▶ Extended Transformers for `CanTSyn`, `FrTSyn` and `EthTSyn` to support AUTOSAR RFC 75119

▶ Extended Transformer for `Com` to configure the `ComTransferProperty` of an `I-SIGNAL-GROUP` by taking the `ComTransferProperty` values of the contained `I-SIGNAL` elements into account

▶ Extended Transformer for `CanTSyn` to configure time domain specific data-ID lists

▶ Extended Transformer for `SecOC` to reflect the `SecOC` parameter changes in ACG 8.3 and ACG 8.4

▶ Adapted Transformers for `CanNm`, `FrNm`, `UdpNm` and `Nm` to reflect that several parameters in the modules have been moved from the general containers to the Nm channel related containers

▶ ASCCONFIGURATORS-818 Fixed known issue: Validation error for `DcmDspRoutineInfo` if no signal is configured

► Extended DcmTransformer for `DcmDspComControl` to configure the `DcmDspComControlAllChannel` and `DcmDspComControlSpecificChannel` subcontainers

Module version 2.7.6
2017-08-24

► ASCCONFIGURATORS-808 Fixed known issue: Incomplete import of Diagnostic configuration for `DcmD-spDataType`, `DcmDspDataSize`, and `DcmDspRoutineSignalType`

► ASCCONFIGURATORS-815 Fixed known issue: Received `SecuredIPdus` with `rxSecurityVerification` set to `false` are not routed via Gateway

► ASCCONFIGURATORS-814 Fixed known issue: Configuration of one and the same PDU for sending and receiving on the same VLAN does not work

Module version 2.7.4
2017-07-28

► Extended Transformer for PduR to support the fan-out of PDUs on different VLANs of the same Ethernet network

► Extended Transformer for Com to support the configuration of 64-bit signed/unsigned signal datatypes

► Modified Transformer to correctly create and configure `DcmDspDidInfo` for `DiagnosticIoControlService`

► Extended Transformer for `DcmProcessingCondition` to configure `DcmModeRule` and `DcmModeCondition`

► Extended Transformer for `Can` to make instance suffix creation robust against invalid `RxIdentifierRange`s

► ASCCONFIGURATORS-805 Fixed known issue: Received `SecuredIPdu` in `ContainerIPDU` with `rxSecurityVerification` set to `false` causes error during configuration import

► ASCCONFIGURATORS-807 Fixed known issue: Importer stops with an error if VLANs of Switch Ports are not connected to Ecu Instance

► Modified `DemTransformer` (`DiagnosticExtendedDataRecordHandler` and `DiagnosticEnableConditionGrouphandler`) to reduce the number of configured `DemExtendedDataClass` and `DemEnableConditionGroup` containers

Module version 2.7.3
2017-06-29

▶ Extended Transformer for ComM to configure `ComMPncComSignalChannelRef` in ERA `ComMPnc-ComSignal` containers

▶ Extended Transformer for EthIf to support `MaximumTransmissionUnit` from `EthernetCommunicationConnector` as well as from `EthernetCommunicationController`

▶ Extended Transformer for SecOC to support the configuration of `GeneralPurposeIPdu` elements in `SecOCTxAuthenticPduLayer` or `SecOCRxAuthenticPduLayer` containers

▶ Extended Transformer for PduR to support the fanout of sent `SecuredIPdu` elements

▶ Extended Transformer for EthIf and EthSwt to support managed switching of switch ports

Module version 2.7.2
2017-06-02

▶ Extended Transformer for Dcm to configure `DiagnosticPeriodicID`

▶ Extended Transformer for PduR to support the configuration of routing-only multiplexed PDUs and the configuration of multiplexed PDUs that contain routing-only demultiplexed PDUs

▶ Extended Transformer for PduR, CanIf, FrIf, and LinIf to support the configuration of routed Xcp PDUs

▶ Modified `DiagnosticDIDHandler`, `DiagnosticDynamicallyDefineDataIdentifierHandler`, and `DiagnosticIOControl` to reduce the number of configured `DcmDspDidInfo` containers

▶ Extended Transformer for SecOC to support `SecureCommunicationFreshnessProps` and `SecureCommunicationAuthenticationProps` for the configuration of SecOC PDUs

▶ Extended Transformer for IpduM to support the configuration of `IpduMContainerTxTriggerMode`

▶ Extended Transformers for CanTSyn and FrTSyn to support `GlobalTimeDomain.globalTimePduTriggering` as well as `GlobalTimeDomain.globalTimePdu`

Module version 2.7.1
2017-05-05

▶ Modified DemTransformer to reduce the number of configured `DemFreezeFrameClass` and `DemFreezeFrameRecNumClass` elements

▶ Configured `DemDTCClass` to suport DTC of type `DiagnosticTroubleCodeObd`

▶ Changed the way how the `DcmDspDataType` is configured based on Byte Array Signals

▶ Extended Transformer for Dcm to configure `DcmDslBuffer`

▶ Improved the configuration of `DiagnosticOperationCycles` and `DiagnosticAgingCycles`

▶ Extended DcmTransformer to set the `DcmDslProtocolSessionRef` reference to every `DcmDslProtocolRow`

Module version 2.7.0

2017-03-31

▶ Extended Transformer for Dcm to configure `ClearDiagnosticInformation`

▶ Extended Transformer for Dcm to configure `DiagnosticWriteMemoryByAddress` services

▶ Extended Transformer for Dcm to configure `DiagnosticReadMemoryByAddress` services

▶ Extended Transformers for CanNm, FrNm, and UdpNm to configure `<Bus>NmPnResetTime`

▶ Extended Transformer for Dem to configure property `<Bus>DemAgingCycleRef` from the `DemEvent-Class`

▶ Extended Transformers for UdpNm and Nm to support UdpNmChannels associated with EthernetPhysicalChannels/VLANs

▶ Extended Transformer for Com to take `ISignal.iSignalType` into account for the calculation of signal data types

▶ Extended Transformer for LinIf to support the configuration of AssignNad, AssignFrameId, and UnassignFrameId frames using AUTOSAR 4.3.0 LinSlaveConfig entities

▶ Extended Transformer for Com to support the configuration of ComSignal/ComFirstTimeout

▶ Extended Transformers for CanTSyn and FrTSyn to support the configuration of data-ID lists

▶ Extended Transformer for EthTSyn to support the configuration of data-ID lists and EthTSynFramePrio

▶ Extended Transformers for Com and SoAd to support the IPV4 and IPV6 address value `ANY`

▶ Extended Transformer for UdpNm to support the configuration of `UdpNmImmediateNmCycleTime` and `UdpNmImmediateNmTransmissions`

▶ Extended Transformers for CanNm, FrNm, UdpNm, and Nm to use `NmCluster.nmPncParticipation` for the configuration of `<Bus>NmComUserDataSupport` and `<Bus>NmPnEnabled`

▶ Extended Dcm to support `DcmDslProtocolPriority`

▶ Added Transformer for `FiM`

▶ Set the `DiagnosticPeriodicRate` category in the `DiagnosticPeriodicTransmission` field in the DcmDsp container

Module version 2.6.10

2017-03-03

▶ Extended Transformer for Dcm to configure `DiagnosticIOControl`

▶ Configured `DcmDsdSidTabSubfuncAvail` parameter for `DiagnosticEcuReset`, `DiagnosticRoutineControl` and `DiagnosticReadDTCInformation` services

▶ Added Transformer for Xcp

▶ Modified Dem and Dcm configurators to not create an additional `configSet` container if one already exists

▶ ASCCONFIGURATORS-666 Fixed known issue: The `DcmRteUsage` is configured based on other parameters `DcmDspDataUsePort` and `DcmDspRoutineUsePort`

▶ Added support for IPV6 IP addresses to Transformers for TcpIp and SoAd

▶ Moved the available subfunction for diagnostic service `DynamicallyDefineDataIdentifier` from the configuration of the DataIdentifier (DID) to the configuration of the diagnostic service

▶ ASCCONFIGURATORS-652 Fixed known issue: Dcm importer wrongly adds session and security levels for a DiagnosticService

▶ ASCCONFIGURATORS-719 Fixed known issue: Dem configuration generates an error if `Diagnostic-DataIdentifier` and `DiagnosticDataElement` have the same `ShortName`

Module version 2.6.9

2017-02-03

▶ ASCCONFIGURATORS-651 Fixed known issue: Fixed Dcm error if a DID and one of its aggregated `DiagnosticDataElement` have the same `SHORT-NAME`

▶ ASCCONFIGURATORS-662 The `DcmDsdSubServiceSecurityLevelRefs` for the SubServices of `DiagnosticSecurityAccess` will not be configured.

▶ Extended Transformer for SecOC to support the configuration of secured PDUs of type `DcmIPdu`

▶ Extended Transformer for SecOC to support the configuration of secured PDUs for which `IPdu-Port.rxSecurityVerification` is set to `false`

▶ Extended Transformer for ComM to check whether a given PNC ID actually lies within the specified PNC vector

▶ Extended Transformer for TcpIp to add limited broadcast address entries if they are needed for DHCP

▶ Created ECU parameter configuration in the Bsw module Dcm for UDS service `ReadDataByPeriodicIdentifier` from information contained in the diagnostic extract

▶ Extended Dcm to configure the `DcmDspDataUsePort` parameter concerning the cases described by `ServiceNeeds/DiagnosticValueNeeds.processingStyle`

▶ ASCCONFIGURATORS-665 Fixed known issue: Fixed Dem configurator to configure the `DemEventPriority` parameter

▶ Extended Transformer for Dcm to configure `DynamicallyDefineDataIdentifier`

Module version 2.6.8

2017-01-05

▶ ASCCONFIGURATORS-652 Fixed known issue: Fixed Dcm importer wrongly adds session and security levels for a `DiagnosticService`

► Extended Transformer for Can: Add support for the configuration of `CanObjectId` in multi-`CanController` scenarios

► Extended Transformer for EthIf to support `EthIfController`, `EthIfPhysController`, and `EthIfSwitch` configuration containers

Module version 2.6.7

2016-12-02

► ASCCONFIGURATORS-654 Fixed known issue: Transformer for DoIP does not configure all `DoIPTcpConnection` containers

► Extended Transformer for Dcm to configure `ControlDTCSetting`

► Extended Transformer for IpduM to support multiplexed PDUs and secured PDUs In container PDUs

► Added Transformer for SecOC

Module version 2.6.6

2016-11-04

► Added Transformer for FrArTp

► Extended Transformers for Com and PduR to support bidirectional routing of PDUs and signals

► Extended Transformer for SoAd: `ProvidedServiceInstance` and `ConsumedEventGroup` elements are considered for the configuration of `SoAdSocketFramePriority`

Module version 2.6.5

2016-10-21

► ASCCONFIGURATORS-606 Fixed known issue: The parameter DcmDspDataInfoRef is not set and its container DcmDspData is not referenced by DIDs, if no Base Data Type can be determined

► Extended Transformer for Dcm to configure `DiagnosticEcuReset`

► ASCCONFIGURATORS-607 Fixed known issue: The parameter DcmDspRoutineSignalLength is not configured if the DiagnosticDataElement evaluates to a non-array element

► ASCCONFIGURATORS-608 Fixed known issue: Import ECU configuration erroneously configures SubServices for the Dcm RoutineControl service

► Extended Transformer for TcpIp to configure `TcpIpAssignmentPriority`

► Adapted Transformer for Nm: NmChannelId is not configured any more

► Adapted Transformer for TcpIp to support AUTOSAR 4.2.2 parameter structure

▶ Extended Transformers for PduR and SoAd to support PDU routing between <Bus>Tp and SoAd

▶ Extended Transformer for Sd to support AUTOSAR RFC 73286

▶ Added Transformer for EthSwt

▶ Extended Transformer for Dem to configure `DiagnosticExtendedDataRecord`

Module version 2.6.4
2016-09-09

▶ ASCCONFIGURATORS-614 Fixed known issue: Transformer for Dcm configures DcmDslProtocolRx-TesterSourceAddr incorrectly

▶ Extended Transformer for DoIP to support configurations without `DiagnosticConnections` in the imported system model

▶ Added Transformers for CanTSyn and FrTSyn

▶ Added Transformer for AUTOSAR 4.0 Dem: Added support for configuring `DiagnosticTroubleCode`, `DiagnosticTroubleCodeGroup`, `DiagnosticEvent`, `DiagnosticEnableCondition`, `DiagnosticEnableConditionGroup` and `DiagnosticIndicator` into Dem

▶ Extended Transformer for Dem to configure `DiagnosticDataElement`s and `DiagnosticFreezeFrame`s

▶ Extended Transformer for Dcm to configure `DiagnosticReadDTCInformation`

▶ Extended Transformer for Dcm to configure `/DcmDspRoutine/DcmDspRoutineUsePort`

Module version 2.6.3
2016-08-05

▶ ASCCONFIGURATORS-604 Fixed known issue: Configuration of vehicle announcement in DoIP assumes incorrect remote IP address

▶ ASCCONFIGURATORS-599 Fixed known issue: Default Buffer Assignment does not support CAN 2.0 and CAN FD PDUs with identical CAN Ids

Module version 2.6.2
2016-07-01

▶ ASCCONFIGURATORS-594 Fixed known issue: Upper layer of `UserDefinedIPdus` and `GeneralPurposeIPdus` is configured incorrectly in <Bus>If modules

▶ Extended Transformer for Tcp to configure at least one `TcpIpAddrAssignment` container for each `TcpIpLocalAddr` entry

Module version 2.6.1

2016-05-25

▶ Extended Transformer for SoAd to support `SoAdSocketConnection` elements without `shortLabel` attribute

▶ Extended Transformer for Can, Eth, Fr, Lin to support the configuration parameter set defined by the AUTOSAR 4.2.2 standard

▶ Extended Transformer for Dcm to configure `DiagnosticRoutineControl`

Module version 2.6.0

2016-04-29

▶ ASCCONFIGURATORS-569 Fixed known issue: Upper Layer for UUDT DcmIPdus is configured incorrectly in CanIf, LinIf, and FrIf

Module version 2.5.5

2016-04-01

▶ ASCCONFIGURATORS-557 Fixed known issue: Creation of the Dcm configuration from system description does not calculate the value of `DcmDspDataSize` correctly for non-array data instances

▶ Extended Transformers for Can and CanIf to create dedicated HOHs for CAN-FD Frames

Module version 2.5.4

2016-03-04

▶ ASCCONFIGURATORS-558 Fixed known issue: Transformer for LinIf does not support LinSlaveConfig elements without LinSlaveConfigIdent

▶ Extended Transformer for SoAd to configure `SoAdSocketTcpKeepAliveProbesMax`, `SoAdSocketTcpKeepAliveInterval`, and `SoAdSocketTcpKeepAliveTime`

▶ Extended Transformer for ComM to configure `ComMPNCGatewayType`

Module version 2.5.3

2016-02-05

▶ Added input data validity checks to Transformer for Dcm

Module version 2.5.2

2016-01-15

▶ ASCCONFIGURATORS-533 Fixed known issue: Transformer for Com configures ComSignalDataInvalid-Value and ComSignalInitValue for UINT8_N signals incorrectly

▶ ASCCONFIGURATORS-534 Fixed known issue: Transformer for Sd creates unnecessary SdConsumed-Methods and SdProvidedMethods configuration containers

▶ ASCCONFIGURATORS-537 Fixed known issue: Transformer for ComM configures ComMChannelId incorrectly

▶ Extended Transformer for LinIf: Add support for LinSlaveConfig elements

▶ ASCCONFIGURATORS-506 Fixed known issue: Transformers for PduR and Com do not conform to TPS_SYST_01056

▶ Extended Transformers: Resolved naming conflicts for PDUs received with the same CAN ID on different busses

▶ ASCCONFIGURATORS-535 Fixed known issue: Transformer for CanIf does not support CAN FD PDU configuration according to Autosar 4.2.2

Module version 2.5.1

2015-11-06

▶ ASCCONFIGURATORS-518 Fixed known issue: Configuration import into Com incorrectly reports error on inconsistent endianess in group signals

▶ Removed Transformers for ComXf, SomeIpXf, E2EXf

▶ Adapted Transformer for PduR: Add routing paths for UserDefinedIPdus and GeneralPurposeIPdus

▶ ASCCONFIGURATORS-525 Fixed known issue: ComM channel retrieval for Multiplexer PDUs leads to internal assertion failure

▶ Adapted Transformer for DoIp: Adaptations according to DoIP related changes in AUTOSAR 4.2.2 system model

▶ Added Transformer for Dcm

Module version 2.5.0

2015-10-09

▶ Adapted Transformer for CanNm, UdpNm: Nm PDU user data byte length is now calculated by subtracting the non user data byte length from the total PDU byte length, removed support for obsolete parameter UdpNmBusLoadReductionEnabled

► Adapted Transformer for CanIf: Extended support for CAN Id range reception according to AUTOSAR RFC 66324

► Added support for Com datatype retrieval for DataTypePolicy "transformingISignal"

► Adapted Transformer for EthTSyn, StbM: Added support for the configuration of EthTSynGlobalTimeTx-Period, StbMOffsetTimeBase

► Adapted Transformer for CanNm: Implemented parameter mapping of CanNmCarWakeUpFilterEnabled, nmCarWakeUpRxEnabled according to AUTOSAR RFC 65423

Module version 2.4.4

2015-08-14

► Adapted Transformer for TcpIp: TcpIpAddressType is now also configured for TcpIpLocalAddr entries that have TcpIpStaticIpAddressConfig/TcpIpStaticIpAddress set to ANY

► Adapted Transformer for UdpNm: UdpNmChannelConfig container is only configured for VLANs on which Nm PDUs are sent

► Adapted Transformer for SoAd: SoAdSocketConnectionGroup/SoAdPduHeaderEnable is set to true if at least one transmitted PDU is associated with a valid header id

► Added proxy UdpNm Transformer to package dreisoft.tresos.comimporter.api.transformer.asr41 since the extension of the UdpNm module expects the Transformer to reside in that package

► Adapted Transformer for Sd: SdInstance containers are only created for VLANs in which the configured ECU executes at least one Client Service or Server Service

► Adapted Transformer for ComXf, SomeIPXf, E2EXf: Transformer technologies that contain either "1" or "1.0.0" as version number are processed now

Module version 2.4.3

2015-06-19

► Adapted Transformers for SoAd and Sd to support seamless service migration/AUTOSAR RFC 61738

Module version 2.4.2

2015-05-22

► Update Transformer for Nm according to changes in AUTOSAR RFC 61777

► ASCCONFIGURATORS-467 Fixed known issue: Transformer for Com issues "zero length BigInteger" error message

► Added Transformers for EthTSyn, StbM

► Implemented ComDataType retrieval in Transformer for Com according to AUTOSAR RFC 65384

► ASCCONFIGURATORS-469 Fixed known issue: Transformer for IpduM configures IpduMSelectorField-Position in MostSignificantByteFirst selector fields incorrectly

► Added Transformer for E2EXf

► Added Transformer for ComXf

► Extended Transformers for Com, IPduM, PduR, CanIf, FrIf, SoAd, EcuC to support Container/Contained PDUs

Module version 2.4.1

2015-02-20

► Transformer for LinIf now configures LinIfComMNetworkHandleRef

► Transformer for FrIf now configures FrNm fan-in PDU according to FlexRay slot number, base cycle and cycle repetition parameters

► Transformers for Com and ComM now configure sent PNC IRA signals

► Transformers for EcuC, CanNm, FrNm, UdpNm, PduR, Com, ComM now configure PNC ERA PDUs and signals only for ACTIVE PNC gateways

► Added Transformer for SomeIpXf

Module version 2.4.0

2015-01-07

► ASCCONFIGURATORS-430 Fixed known issue: Transformer for Com configures initial values and invalid values of UINT8_N signals incorrectly

► ASCCONFIGURATORS-440 Fixed known issue: Transformers stop with an error if direct Tp SDU of FrTpConnection is not available

► Added support for 1..* cardinality of CanNmRxPdu containers in Transformer for CanNm

► Added support for PNC ERA PDU configuration in Transformers for EcuC, CanNm, FrNm, UdpNm, PduR, Com, ComM

► Added support for PNC Identifier configuration according to AUTOSAR RFC 52483 in Transformer for ComM

Module version 2.3.0

2014-10-03

▶ Made Sd Transformer robust against duplicated routing groups

▶ ASCCONFIGURATORS-423 Fixed known issue: Transformer for Com generates wrong ComSignalType for signals assigned to "Array" SwBaseType

▶ Extended Transformers to support AUTOSAR 4.2.1 system model files as input

▶ Added support for dynamic length signals in Com

Module version 2.2.2

2014-09-05

▶ Added Transformers for CanSM, FrSM, LinSM and EthSM

▶ Added Transformer for DoIP

▶ Added Transformer for LdCom

Module version 2.2.1

2014-08-07

▶ Adapt configuration of SdClientServiceActivationRef, SdServerServiceActivationRef according to AUTOSAR 4.1.3 Upstream Mapping

▶ Adapt Transformer for Eth so that the MAC address is configured according to AUTOSAR

▶ Added Transformer for UdpNm

▶ Transformer for Sd now configures Sd Control PDUs

▶ Transformer for PduR handles routing of Diagnosis PDUs according to AUTOSAR RFC 63555

▶ Transformer for Com now supports dataTypePolicy values other than "legacy"

▶ ASCCONFIGURATORS-387 Fixed known issue: FrTpTransformer configures FrTpRxPduPoolRef and FrTpTxPduPoolRef incorrectly

▶ ASCCONFIGURATORS-393 Fixed known issue: Transformer for CanTp configures identical symbolic names for received N-PDUs used for Data and FC at the same time

▶ ASCCONFIGURATORS-394 Fixed known issue: Transformer for FrIf configures incorrect PDU references to EcuC

▶ ASCCONFIGURATORS-391 Fixed known issue: PNC ISignalIPduGroups are only considered if the configured ECU instance directly references them

▶ Transformer for Sd now supports the configuration of combined Tcp/Udp services

▶ ASCCONFIGURATORS-399 Fixed known issue: Transformer for ComM sets ComMNmVariant to "NONE" for LIN Clusters

▶ ASCCONFIGURATORS-395 Fixed known issue: Transformers create User Data PDU containers for Nm PDUs containing signals that are not processed by the ECU

▶ Transformers for CanNm, FrNm, and UdpNm now configure the links to the associated ComM channels

▶ Transformers for ComM now configures ComMChannelId

▶ ASCCONFIGURATORS-396 Fixed known issue: Transformer for Nm does not configure NmComUserDataSupport

▶ Transformer for Com now configures ComGWMapping entries for group signals

Module version 2.2.0

2014-04-25

▶ ASCCONFIGURATORS-349 Fixed known issue: Transformer for PduR sets up incorrect routing paths for reversed FlexRay Tp SDUs

▶ ASCCONFIGURATORS-355 Fixed known issue: Transformer for Com does not create gateway mapping entries for signal groups

▶ ASCCONFIGURATORS-346 Fixed known issue: Transformer for LinIf configures `AssignNad` frame incorrectly

▶ ASCCONFIGURATORS-354 Fixed known issue: Transformer for LinIf reports an error if it encounters `AssignFrameIdRange` schedule table entries without `PID`

▶ Extended Transformers to support AUTOSAR 4.1.2 system model files as input; added Transformer for Sd

Module version 2.1.10

2014-01-17

▶ Integrated handling for CanTp N-PDUs assigned to multiple CanTp connections

▶ Integrated AUTOSAR 4.0 ComM Transformer

Module version 2.1.9

2013-10-11

▶ Internal module improvement. This module version update does not affect module functionality

Module version 2.1.8

2013-09-13

▸ Removed configuration of PN ERA PDUs which are not yet supported by CanNm and FrNm

Module version 2.1.7

2013-06-14

▸ ASCCONFIGURATORS-296 Fixed known issue: The configuration of one and the same CanNm PDU for sending and receiving results in a naming conflict

▸ Added support for `CONTAINED-I-SIGNAL-I-PDU-GROUP-REF` to detect PNC-enabled `NmCluster`s

▸ Added support for routed-only Tp-SDUs in PduR

▸ Added configuration of `CanNmPnFilterMaskByte` in CanNm

▸ Added support for configuration of `ComTxModeTrue` for sent PDUs without PDU Timing in Com

Module version 2.1.6

2013-05-10

▸ ASCCONFIGURATORS-277 Fixed known issue: Com Transformer restricts signals to be contained at most once per FlexRay frame

▸ ASCCONFIGURATORS-279 Fixed known issue: CanNm Transformer issues exception if handed over Can Network contains Rx Nm PDUs but no CAN Frame Triggerings

▸ ASCCONFIGURATORS-276 Fixed known issue: If a PDU is sent and received at the same time and fan-out takes place for the sent PDU, the received PDU instance is configured incorrectly

▸ ASCCONFIGURATORS-283 Fixed known issue: LinTp Transformer produces name clashes when importing from AUTOSAR 3.x LinTp configurations

▸ ASCCONFIGURATORS-286 Fixed known issue: CanTp Transformer cannot handle `CAN-TP-CONNECTION`s without `TP-SDU-REF`

Module version 2.1.5

2013-02-08

▸ ASCCONFIGURATORS-230 Fixed known issue: Transformer for AUTOSAR 4.0 Com configures signal invalid values and signal init values > 2^53-1 incorrectly

▸ ASCCONFIGURATORS-250 Fixed known issue: `ComTxModeNumberOfRepetition` is not configured correctly for AUTOSAR 4.0.3 Com configurations

▸ ASCCONFIGURATORS-254 Fixed known issue: LinIf Transformer uses wrong set of PDUs for retrieval of TxPdu fan-out information

▶ ASCCONFIGURATORS-256 Fixed known issue: PduR Transformer cannot handle <any bus>-PDU to Ethernet-PDU gateway mappings

▶ ASCCONFIGURATORS-258 Fixed known issue: FrIf Transformer does not configure FrIfluster.gdBit according to upstream mapping

▶ Implemented support for TP-SDUs referenced by multiple PDU triggerings

▶ Removed configuration of `<Net>NmPnFilterMaskByte` due to unclear upstream mapping rule

▶ Added pure gateway routing support

Module version 2.1.4

2012-10-12

▶ AUTOSAR 4.0 FrIf/FrNm: Added support for FrNm Rx PDU fan-in

▶ ASCCONFIGURATORS-230 Fixed known issue: Transformer for Com does not create `GroupSignal` if `SignalTriggering` is missing

Module version 2.1.3

2012-09-14

▶ AUTOSAR 4.0 Com: Added support for Com Rx Signal DataFilters

▶ AUTOSAR 4.0 Com: Adapted `ComSignalType` configuration: Special handling of type `BOOLEAN`

Module version 2.1.2

2012-08-17

▶ AUTOSAR 4.0 Com: User-defined prefix is added to group signal containers

▶ AUTOSAR 4.0 SoAd: Adapted according to configuration changes

▶ AUTOSAR 4.0 Nm: Adapted according to configuration changes

▶ AUTOSAR 4.0 Com: Adapted configuration of parameter `ComSignalType` according to AUTOSAR 4.-0 Rev 3 System Template

Module version 2.1.1

2012-06-15

▶ AUTOSAR 4.0 SoAd: Implemented configuration of `SoAdSocketConnectionGroup` containers

► AUTOSAR 4.0 Transformers: PDU Router fan-out support has been added

Module version 2.1.0

2012-05-16

► AUTOSAR 4.0 CanIf, CanTp: Adapted parameter configuration according to configuration changes in AU-
   TOSAR 4.0 Rev 3

Module version 2.0.6

2012-04-13

► AUTOSAR 4.0 SoAd: Adaptation due to removed/obsolete parameter `SoAdPduHeaderEnable`

Module version 2.0.5

2012-03-27

► Added support for the configuration of partial networks for AUTOSAR 4.0 FrNm

► Integrated AUTOSAR 4.0 Transformers for modules SoAd, EthIf, Eth, TcpIp

► ASCCONFIGURATORS-187 Fixed known issue: AUTOSAR 4.0 Com: Fixed signal data type calculation
   algorithm

Module version 2.0.4

2012-02-17

► Removed obsolete AUTOSAR 3.x Transformers

► Added support for the configuration of partial networks for AUTOSAR 4.0 CanNm

Module version 2.0.3

2012-01-20

► Integrated AUTOSAR 4.0 Transformers for modules Lin, LinIf, and LinTp

► Added User Data Nm PDU support to the involved AUTOSAR 4.0 Transformers EcuC, PduR, Com, and
   CanNm, FrNm

Module version 2.0.2

2011-10-12

▻ Integrated AUTOSAR 4.0 Transformers for modules Fr, FrIf, CanTp, CanNm, FrTp, FrNm, Nm

▻ ASCCONFIGURATORS-138 Fixed known issue: FrNm: Fixed Rx/Tx Nm PDU handling

Module version 2.0.0

2011-09-02

▻ ASCCONFIGURATORS-86 Fixed known issue: IpduM: Transformer for AUTOSAR 3.x IpduM configures `IPduMTxSelectorValue`

▻ ASCCONFIGURATORS-85 Fixed known issue: IpduM, EcuC: Containers for Demultiplexed PDUs are only created if the Com module actually processes them

▻ Integrated AUTOSAR 4.0 Transformers for modules EcuC, Can, CanIf, IpduM, PduR, and Com

Module version 1.1.2

2011-04-08

▻ ASCCONFIGURATORS-51 Fixed known issue: LinIf: Fixed references to EcuC PDU collection

▻ Improvement CanTp: Support configuration of multicast CanTp connection channels

Module version 1.1.1

2011-03-11

▻ Improvement CanTp, CanIf, EcuC: Duplication of CanTp Tx N-PDUs has been introduced

▻ Improvement LinIf: Transformer configures `LinIfFramePriority` if provided via LDF importer

▻ Transformer for generic Nm was added

▻ Improvement Com: Support for zero bitsize signals has been added

Module version 1.1.0

2011-02-03

▻ Requirements tracing: Resolve unmapped tests, unmapped requirements

▻ Improvement CanNm, FrNm, CanIf, LinIf, Com: Transformers use new Meta-Model 6/AUTOSAR 3.1.4 System parameters for module configuration

▶ Improvement CanIf, Can: Transformers use better naming schema for configuring HOH containers

Module version 1.0.2

2010-11-18

▶ ASCCONFIGURATORS-28 Fixed known issue: AUTOSAR 2.1/3.x Com: Com data types calculated correctly from integral system model data types with open ranges

Module version 1.0.1

2010-10-08

▶ Improvement AUTOSAR 3.x LinIf: Conditional frames obtain `LinIfInternalPdu` as `LinIfPduDirection` during com imports

▶ ASCCONFIGURATORS-22 Fixed known issue: AUTOSAR 2.1/3.x Com: Bit offsets and length parameters of signal groups are correctly exported

Module version 1.0.0

2010-09-10

▶ First implementation of Configurators

2.3.1.2. New features

▶ Configuration support for Csm has been added.

2.3.1.3. EB-specific enhancements

This module is not part of the AUTOSAR specification.

2.3.1.4. Deviations

This module is not part of the AUTOSAR specification.

2.3.1.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

▶ `ComSignalInitValue/ComSignalDataInvalidValue` when importing from FIBEX

Description:

When importing from FIBEX, initial and invalid values that have been defined as bit pattern that represent `FLOAT` values are interpreted as integral values and written to `ComSignalInitValue/ComSignalDataInvalidValue` as such.

Rationale:

Initial and invalid values are defined as `INTERNAL-CONSTRS` in FIBEX, which requires these values to be integral values.

▶ `CanTpChannel/CanTpChannelMode` when importing from FIBEX

Description:

When importing from FIBEX, only elements of `CanTpChannel` that contain a single `CanTpRxNSdu/CanTpTxNSdu` are created. Also, the `CanTpChannelMode` of these elements `CanTpChannel` is always set to `CANTP_MODE_HALF_DUPLEX`.

Rationale:

FIBEX 3.x only supports a single `TP-CONNECTION` per `TP-CHANNEL`.

▶ `ISignal.dataTypePolicy`:`networkRepresentationFromComSpec` requires either a SenderReceiverToSignalMapping or a SenderReceiverToSignalGroupMapping

Description:

An `ISignal` that has `dataTypePolicy` set to `networkRepresentationFromComSpec` requires a `SenderReceiverToSignalMapping` or a `SenderReceiverToSignalGroupMapping`. These mapping elements reference a `PortPrototype` which contains a `PortComSpec` that defines the `NetworkRepresentation`. A `DataMapping` other than `SenderReceiverToSignalMapping` or `SenderReceiverToSignalGroupMapping` is not supported.

▶ `ISignal.dataTypePolicy`: Data type retrieval via `networkRepresentationFromComSpec` and `ImplementationDataType` is only supported for plain `ISignal` elements

Description:

In the AUTOSAR System Template 4.2 Rev 2, [TPS_SYST_02006] and [TPS_SYST_02079] describe the data type retrieval via `ImplementationDataType` in the case that no `SenderComSpec` or `Receiver-`

`ComSpec` is available. This is currently only implemented for plain signals, not for `ISignals` that represent group signals.

▶ `ComIPdu/ComIPduCounter` is not configured

Description:

`ComIPdu/ComIPduCounter` configuration containers are not configured even if the imported AUTOSAR System Description file contains `SIGNAL-I-PDU-COUNTER` elements.

Rationale:

The ACG Com module does not support `ComIPdu/ComIPduCounter` configuration containers.

▶ `ComIPdu/ComIPduReplication` is not configured

Description:

`ComIPdu/ComIPduReplication` configuration containers are not configured even if the imported AUTOSAR System Description file contains `SIGNAL-I-PDU-REPLICATION` elements.

Rationale:

The ACG Com module does not support `ComIPdu/ComIPduReplication` configuration containers.

▶ `ComSignal/ComRxDataTimeoutAction` and `ComSignalGroup/ComRxDataTimeoutAction` are not configured

Description:

`ComSignal/ComRxDataTimeoutAction` and `ComSignalGroup/ComRxDataTimeoutAction` are not configured even if the imported system model contains `HANDLE-TIMEOUT-TYPE` elements.

Rationale:

The imported system model can contain two or more `HANDLE-TIMEOUT-TYPE` elements that are related to one and the same `ComSignal/ComRxDataTimeoutAction` or `ComSignalGroup/ComRxData-TimeoutAction` parameter. Since these `HANDLE-TIMEOUT-TYPE` elements may contain different values, a configuration is not possible.

▶ `FrController/FrFiFo` is not configured

Description:

`FrController/FrFiFo` configuration containers are not configured even if the imported AUTOSAR System Description file contains `FLEXRAY-FIFO-CONFIGURATION` elements.

Rationale:

Not all types of FlexRay communication controllers can be configured using the parameters provided in `FrController/FrFiFo`. It is therefore required to configure `FrController/FrFiFo` by hand, while taking into account the type of FlexRay communication controller that is in use.

► No support for tx fan-out of PDUs that are contained in `CONTAINER-I-PDU` elements

Description:

While the AUTOSAR System Template 4.2 Rev 1 indicates that PDUs contained in `CONTAINER-I-PDU` elements can be subject to a tx fan-out in the PduR module, the Transformer for PduR currently does not support this feature.

Rationale:

The tx fan-out support for contained PDUs requires a dedicated extension of the Transformer for PduR which has not yet been implemented.

► `AssignNAD` frames referencing Lin slave nodes which are not declared in the imported LDF file lead to import errors

Description:

During LDF imports, it is required that all `AssignNAD` frames in the LDF file reference Lin slave nodes which are also present in the file. If this is not the case, an error is issued.

Rationale:

The AUTOSAR System Model, which serves as internal data storage during the import, does not provide the possibility to store an `AssignNAD` frame without referencing the related slave node at the same time. Since the EB tresos Studio importer framework is built on top of the AUTOSAR System Model, this parameter cannot be configured.

2.3.1.6. Open-source software

Configurators does not use open-source software.

3. ECU Configuration Wizard user's guide

3.1. Overview

A configuration import consists of two steps. In the first step, one or more configuration files representing a system model - or an ECU extract thereof - are imported into the EB tresos Studio system model representation. In the second step the user selects one of the ECUs of the imported model for configuring a number of AUTOSAR modules in the EB tresos Studio project. Subsequently, the ECU relevant aspects of the imported system model are translated into AUTOSAR module configurations.

This part of the documentation describes the second importer step in more detail. For a description of the first importer step, see the EB tresos Studio user's guide.

► Section 3.3, "Common mapping aspects" describes general information that applies to the mapping between system model and AUTOSAR module configurations.

► Section 3.4, "Importer parameter mappings for AUTOSAR 4.0 modules" describes the mappings that are made between the system model and the respective AUTOSAR module configurations.

3.2. Background information

The chapter *Introduction* and the subchapter *Scope* in [3] provide the background information on the configuration of AUTOSAR modules out of a system model.

3.3. Common mapping aspects

This section contains some general information that applies to the mapping between system model and AUTOSAR module configurations.

3.3.1. Naming rules

3.3.1.1. Name mangling algorithm for importers

In AUTOSAR configuration files container names must be valid C-Identifiers and must have a length of at most 32 characters. The names of the containers must be unique within the context of the parent containers. In

order to ensure that all containers created during an import comply to the conditions above, object names are mangled.

Any characters violating the rules for valid C-Identifiers are replaced by underscores (_). In case a name exceeds 30 characters it is cut off and an underscore followed by a unique index is added so that the resulting name has a length of at most 30 characters and is unique. Since AUTOSAR allows a length of at most 32 characters, the importers can add prefixes which are at most two characters long without violating AUTOSAR naming rules, e.g. `SG` is added to normal signals, `NW` is added to network signals.

Examples:

▶   `TestName9012345678901234567890Foo` becomes `TestName90123456789012345678_1`

▶   `Test-Name9012345678901234567890Bar` becomes `TestName90123456789012345678_2`

NOTE	**Automatic modification of the container names**
(i)	If the check for the maximum container name length is disabled, the container names are mangled to 110 characters instead of 30. For further information on how to check the maximum container name length, see the EB tresos Studio documentation.
	To avoid that your container name is modified automatically, enable the check for the maximum container name length. For further information on how to check the maximum container name length, see the EB tresos Studio documentation.

3.3.1.2. Name prefixing

Prior to import a prefix can be defined which is added to the ShortName of every imported container. The prefix menu line may be left empty if you do not wish a prefix.

The entered prefix is referred to as `<PREFIX>` in this document.

3.3.1.3. Instance handling

In AUTOSAR module configurations, instances of frames, PDUs, and signals are handled in different ways, depending on the network. Whereas for CAN and LIN configurations there is usually a one-to-one relationship between prototype and instance, this is usually not the case for FlexRay configurations:

FlexRay frames can be transmitted in multiple communication slots over the bus. One PDU may reside in more than one FlexRay frame. The `FlexRay Interface` (`FrIf`) handles this one-to-many relationship by always passing the same PDU to the upper layer (e.g. to the `PDU Router`, `PduR`) even if the PDU is received in different FlexRay communication slots and/or in different frames.

If an upper layer issues a transmit request for a PDU, the `FrIf` sends it in the next FlexRay communication slot that is assigned to a FlexRay frame, which contains the requested PDU.

FlexRay N-PDUs, i.e. PDUs that are processed by the `FlexRay Transport Layer` (`FrTp`), must be handled differently by the `FrIf`. If a FlexRay N-PDU is transmitted in multiple frames and/or in multiple FlexRay communication slots, the `FrIf` processes each of the resulting PDU instances as an individual PDU.

If one and the same PDU is received on a given CAN network via two or more different CAN-IDs, the `CanIf` is set up to receive the PDU via a CAN-ID range that contains all the CAN-IDs via which the PDU is received. Since the `CanIf` does not support *standard* and *extended* CAN-IDs in one and the same CAN-ID range, an error is issued if such a configuration is imported. However, the range reception described here does not apply to N-PDUs or NM-PDUs.

Since every instance container of a configuration needs its unique name, and unique names are usually only provided for prototypes, the problem arises how to obtain unique container names for instances of the same prototype. EB tresos Studio provides an option by which the instance names are generated by appending a suffix to the name of the prototype. This guarantees that the instance name is unique.

FlexRay frames obtain different instance suffixes depending on whether they contain FlexRay N-PDUs or not. FlexRay N-PDUs obtain different suffixes than other FlexRay PDUs do. In the following text this is described in detail:

For FlexRay frames that contain N-PDUs, the following information is added to instances of frames:

▶   FlexRay identifier (1..2047)

▶   Channel information (A|B)

▶   Base cycle (0..63)

▶   Cycle repetition (1|2|4|8|16|32|64)

▶   Transmission direction information (T|R)

An instance of a frame *TestFrame*, received in slot 123, base cycle 10, cycle repetition 32 on channel A would therefore be called `TestFrame_123A1032R`.

If a FlexRay frame does not contain any N-PDU, only the transmission direction information is appended as suffix, e.g. The *TestFrame* above would be called `TestFrame_R`.

For CAN frames, the following information is added to instances of frames:

▶   CAN identifier (0..2^29-1)

▶   Transmission direction information (T|R)

An instance of a frame *TestFrame*, received with CAN-ID 123 would therefore be called `TestFrame_123R`.

For LIN frames, the following information is added to instances of frames:

▶   LIN identifier (0..63)

▶   Transmission direction information (T|R)

Although a LIN frame/PDU can be sent or received in multiple schedule tables or even multiple times in the same schedule, the layer above only sees one instance of this frame/PDU. Therefore the schedule table and the entry position information has been left out in the instance suffix. An instance of a frame `TestFrame`, received with LIN-ID `23` would therefore be called `TestFrame_23R`.

The naming scheme for frame instances can also be applied to PDU instances. The instance of PDU *TestPdu*, which is sent in the frame instance `TestFrame_123R`, is named `TestPdu_123R`.

The naming scheme for signal instances adopts the naming scheme from the I-PDU instance in which it is contained. A signal instance of `TestSignal` transmitted in the PDU instance `TestPdu_123R` is called `TestSignal_123R`.

For an import run, add instance suffixes by checking the **Instance suffix for Frames, PDUs and Signals** checkbox.

If you import FlexRay networks, check the option checkbox to ensure that configuration containers obtain unique names. Since there is usually a one-to-one relationship between prototype and instance in CAN and LIN networks, an instance suffix is not required to obtain unique configuration container names. Uncheck the option checkbox if you import CAN and/or LIN networks to obtain short configuration container names.

Also note that the length of the container names grows considerably if the option is turned on. This usually makes it necessary to turn off the check for the maximum container name length. For instructions on how this check is turned off, see the EB tresos Studio user's guide.

The suffix is referred to by `<INSTSUFFIX>` in this document.

3.3.2. Signal offsets

Signals can be packed into PDUs either in big-endian or little-endian format. The endianness information, together with the signal offset parameter defines the bitset within the PDU which is occupied by a signal. Whereas the endianness of a signal within its enclosing PDU is an agreed upon notion, virtually every document format developed its own peculiar definition of the notion signal offset making an implicit offset conversion during import runs necessary.

For information on how the signal offset is defined, see the external document format specification and the AUTOSAR System Template specification.

3.3.3. Rx NM PDUs

If Rx NM PDUs are used, only the PDU which has the lowest CAN-ID is considered (per cluster) by the AUTOSAR 3.0/3.1 modules `CanIf`, `EcuC`, and `CanNm`.

3.3.4. Duplication of Tx N-PDU configuration containers

If the same outgoing N-PDU is used simultaneously by two `CanTp` connections, i.e. as data N-PDU by an outgoing connection and as flow-control N-PDU by an incoming connection, two configuration containers are created for this N-PDU in the AUTOSAR 3.0/3.1 modules `CanIf`, `EcuC`, and `CanTp`.

The name of the second N-PDU container is suffixed with `_D` and corresponds to the data N-PDU which is used by the outgoing `CanTp` connection.

3.3.5. Collection of N-PDU elements

For `CanTp`, `LinTp`, and `FrTp` connections, there is a one-to-one relationship between an N-PDU and the frame in which the N-PDU is sent or received. For that reason an N-PDU is considered sent or received by an ECU if its frame is sent or received. This is determined by the presence of a Tx or Rx frame port that links the frame to the ECU.

An exception to that are `FrArTp` connections. In the context of `FrArTp`, an N-PDU can be contained in a frame that contains other PDUs as well. As a consequence, the presence of an Tx or Rx frame port is not sufficient to determine whether an N-PDU is sent or received by an ECU. In the context of `FrArTp`, Tx or Rx PDU ports linking the N-PDUs to the ECU are required as well.

3.3.6. PDU routing

An ECU routes a source PDU to one or more target PDUs if there is a gateway mapping which defines the routing paths from the source PDU to the target PDUs.

In this case, the source PDU is configured as an Rx PDU, and the target PDUs are configured as Tx PDUs in the modules `CanIf`, `FrIf`, `LinIf`, `SoAd`, `CanTp`, `FrTp`, `LinTp`, `DoIP`, `SecOC`, `IpduM`, and `EcuC`. The actual routing from source to destination PDUs is configured in the `PduR` module.

A PDU that is configured as a target PDU in a gateway mapping is not configured in any upper layer BSW module of the `PduR`. A sent `ISignalIPdu`, for instance, is not configured in `Com` if it is configured as a target PDU in a gateway mapping. The reason for that is that the `PduR` only allows one single source for the transmission of a PDU. The source can either be a local transmission request of any upper layer BSW module, e.-g. `Com`, or it can be a received source PDU, the content of which is forwarded using the target PDU.

If there is a gateway mapping for a PDU that is related to a *cryptographic* PDU as described in [Section 3.4.42, "SecOC"](), this gateway mapping either applies to the payload PDU or to the authenticated PDU, i.e. the PDU which `SecOC` exchanges with its upper layer module. If there is a gateway mapping for the *cryptographic* PDU as well, the payload PDU is configured to be routed. If there is no gateway mapping for the *cryptographic* PDU, the authenticated PDU is routed.

3.3.7. PDU length calculation

For the calculation of the payload length of a PDU in bytes, two fields are taken into account. The `LENGTH` field of the PDU itself is always used. The second field, `I-PDU-MAPPING/PDU-MAX-LENGTH`, is only taken into account if the `I-PDU-MAPPING` references via `SOURCE-I-PDU-REF` or via `TARGET-I-PDU-REF` a `PDU-TRIGGERING` which in turn references the PDU.

If both `LENGTH` and `I-PDU-MAPPING/PDU-MAX-LENGTH` need to be taken into account for the calculation of the payload length of a PDU, the payload length is considered to be the maximum of both values.

3.4. Importer parameter mappings for AUTOSAR 4.0 modules

3.4.1. Overview

The following chapters explain how system model parameters are mapped into AUTOSAR modules. For more in-depth information on how to invoke the importers for the different network communication formats, see the EB tresos Studio user's guide.

ECU Extract to AUTOSAR parameter mapping

The **System Description Importer** of EB tresos Studio creates configurations for each of the following AUTOSAR modules.

For detailed configuration parameter descriptions, see the EB tresos AutoCore module references.

3.4.2. Can

Configuration parameters	Mapping description
`CanConfigSet/Can-Controller`	For every `CAN-COMMUNICATION-CONTROLLER` that is connected to a `CAN-CLUSTER` or `J-1939-CLUSTER` and to the imported `ECU-INSTANCE`, a `Can-Controller` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `CAN-COMMUNICATION-CONTROLLER`. `CanControllerDefaultBaudrate` references the `CanControllerBaudrateConfig` container created for the `CAN-COMMUNICATION-CONTROLLER`. `CanControllerId` is configured in such a way that each `CanController`

Configuration parameters	Mapping description
	container obtains a unique `CanControllerId` value and the set of all `CanControllerId` values is zero-based and dense.
`CanConfigSet/CanController/CanControllerBaudrateConfig`	`CanControllerBaudRate` is set to `BAUDRATE` / 1000, as defined in the `CAN-CLUSTER` or `J-1939-CLUSTER` of the `CAN-COMMUNICATION-CONTROLLER`. If `BAUDRATE` is not available, `CanControllerBaudRate` is set to `SPEED`. The following parameters are set as defined in the `CAN-CONTROLLER-CONFIGURATION` of the `CAN-COMMUNICATION-CONTROLLER`: `CanControllerPropSeg` is set to `PROP-SEG`. `CanControllerSeg1` is set to `TIME-SEG-1`. `CanControllerSeg2` is set to `TIME-SEG-2`. `CanControllerSyncJumpWidth` is set to `SYNC-JUMP-WIDTH`.
`CanConfigSet/CanController/CanControllerFdBaudrateConfig`	The container `CanControllerFdBaudrateConfig` is only created if `CAN-FD-BAUDRATE` contains a valid value. `CanControllerFdBaudRate` is set to `CAN-FD-BAUDRATE` / 1000, as defined in the `CAN-CLUSTER` or `J-1939-CLUSTER` of the `CAN-COMMUNICATION-CONTROLLER`. The following parameters are set as defined in the `CAN-CONTROLLER-FD-ATTRIBUTES` of the `CAN-COMMUNICATION-CONTROLLER`: `CanControllerPropSeg` is set to `PROP-SEG`. `CanControllerSeg1` is set to `TIME-SEG-1`. `CanControllerSeg2` is set to `TIME-SEG-2`. `CanControllerSyncJumpWidth` is set to `SYNC-JUMP-WIDTH`. `CanControllerTrcvDelayCompensationOffset` [ns] is set to `TRCV-DELAY-COMPENSATION-OFFSET` [s]. `CanControllerTxBitRateSwitch` is set to `TX-BIT-RATE-SWITCH`.
`CanConfigSet/CanController/CanFilterMask`	`CanFilterMask` containers are only created for `Can` modules that conform to the AUTOSAR 4.0.3 standard. Two `CanFilterMask` containers are created for each `CanController`, named `AcceptAllStd` and `AcceptAllExt`. `CanFilterMaskValue` of theses containers is set to zero per default.

Configuration parameters	Mapping description
	If a `parameter` tag named `canfiltermask.dontcarebit` exists in the module's comtransformer extension point, which has its `value` attribute set to `1`, `CanFilterMaskValue` is set to `0x7ff` (`AcceptAllStd`) and `0x1fffffff` (`AcceptAllExt`) instead of zero. If a `parameter` tag named `canfiltermask.std.shiftleft` is defined in the module's comtransformer extension point, `CanFilterMaskValue` of `AcceptAllStd` is shifted left by the value defined in the `value` attribute. If a `parameter` tag named `canfiltermask.ext.shiftleft` is defined in the module's comtransformer extension point, `CanFilterMaskValue` of `AcceptAllExt` is shifted left by the value defined in the `value` attribute.
`CanConfigSet/CanHardwareObject`	If **Buffer Assignment in Can/CanIf** was set to **Create default buffer assignment** during the import, `CanHardwareObject` containers are created for each `CAN-COMMUNICATION-CONTROLLER` connected to a `CAN-CLUSTER` or `J-1939-CLUSTER` of the imported `ECU-INSTANCE`. The container name depends on `CAN-FRAME-RX-BEHAVIOR`/`CAN-FRAME-TX-BEHAVIOR` and on `CAN-ADDRESSING-MODE` of the `FRAME-TRIGGERING` associated with the `PDU` which the `ECU-INSTANCE` sends or receives. The naming schema for every combination is displayed in Table 3.1, "Rx HOH container names depending on `CAN-FRAME-RX-BEHAVIOR` and `CAN-ADDRESSING-MODE`" and Table 3.2, "Tx HOH container names depending on `CAN-FRAME-TX-BEHAVIOR` and `CAN-ADDRESSING-MODE`". `CanIdType` is set depending on the `CAN-ADDRESSING-MODE` of the `FRAME-TRIGGERING` elements the `HOH` containers have been created for. If `CAN-ADDRESSING-MODE` is set to `STANDARD` or `<undefined>`, `CanIdType` is set to `STANDARD`. If `CAN-ADDRESSING-MODE` is set to `EXTENDED`, `CanIdType` is set to `EXTENDED`. `CanObjectType` is set depending on whether the `ECU-INSTANCE` sends or receives the `FRAME-TRIGGERING` elements for which the `HOH` containers were created. If `FRAME-TRIGGERING` elements are received, `CanObjectType` is set to `RECEIVE`. If `FRAME-TRIGGERING` elements are sent, `CanObjectType` is set to `TRANSMIT`. `CanControllerRef` references the `CanController` container created for the `CAN-COMMUNICATION-CONTROLLER` of the HOH. The `CanFilterMaskRef` reference parameter is only configured for `Can` modules that conform to the AUTOSAR 4.0.3 standard. It references a `CanFilterMask` container depending on the `CAN-ADDRESSING-MODE` of the `FRAME-`

Configuration parameters	Mapping description
	`TRIGGERING` for which the `HOH` containers have been created. `HOH` containers created for `CAN-ADDRESSING-MODE STANDARD/<undefined>` reference `AcceptAllStd`, `HOH` containers created for `CAN-ADDRESSING-MODE EXTENDED` reference `AcceptAllExt`. `CanObjectId` is configured according to [Section 3.4.2.1, "CanObjectId configuration"](#). `CanHandleType` is set to `BASIC` for all HOHs. `CanFdPaddingValue` is only configured for `Can` modules that conform to the AUTOSAR 4.2.2 standard. The value is set to `CAN-COMMUNICATION-CONTROLLER/CAN-COMMUNICATION-CONTROLLER-VARIANTS/CAN-COMMUNICATION-CONTROLLER-CONDITIONAL/CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/CAN-CONTROLLER-FD-ATTRIBUTES/PADDING-VALUE` or, if this parameter is not present, to `CAN-COMMUNICATION-CONTROLLER/CAN-COMMUNICATION-CONTROLLER-VARIANTS/CAN-COMMUNICATION-CONTROLLER-CONDITIONAL/CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/CAN-CONTROLLER-FD-REQUIREMENTS/PADDING-VALUE`.
`CanConfigSet/CanHardwareObject/CanHwFilter`	The `CanHwFilter` container is only created for `Can` modules that conform to the AUTOSAR 4.2.2 standard. `CanHwFilterCode` is configured to `0`. `CanHwFilterMask` is configured to the filter mask value that allows that all CAN-IDs are received. The description of `CanConfigSet/CanController/CanFilterMask` in [Section 3.4.2, "Can"](#) provides the calculation formulas of these filter mask values.

CAN-ADDRESSING-MODE CAN-FRAME-RX-BEHAVIOR	STANDARD or <undefined>	EXTENDED
CAN-20 or <undefined>	`HOH_0_<controller name>`	`HOH_1_<controller name>`
CAN-FD	`HOH_0_<controller name>`	`HOH_1_<controller name>`
CAN-ANY	`HOH_0_<controller name>`	`HOH_1_<controller name>`

Table 3.1. Rx HOH container names depending on `CAN-FRAME-RX-BEHAVIOR` and `CAN-ADDRESSING-MODE`

CAN-ADDRESSING-MODE CAN-FRAME-TX-BEHAVIOR	STANDARD or <undefined>	EXTENDED
CAN-20 or <undefined>	`HOH_2_<controller name>`	`HOH_3_<controller name>`

CAN-FD	`HOH_4_<controller name>`	`HOH_5_<controller name>`

Table 3.2. Tx HOH container names depending on `CAN-FRAME-TX-BEHAVIOR` and `CAN-ADDRESSING-MODE`

3.4.2.1. CanObjectId configuration

The `CanObjectId` handle ID parameters of all `CanHardwareObject` containers are configured in such a way that each `CanHardwareObject` obtains a unique `CanObjectId` value and the set of all values is zero-based and dense. Moreover, the values are distributed in increasing order to the following:

▶ RECEIVE `CanHardwareObject` containers that reference the `CanCommunicationController` which has its `CanControllerId` set to 0.

▶ RECEIVE `CanHardwareObject` containers that reference the `CanCommunicationController` which has its `CanControllerId` set to 1.

▶ ...

▶ RECEIVE `CanHardwareObject` containers that reference the `CanCommunicationController` which has its `CanControllerId` set to `<n - 1>`.

▶ TRANSMIT `CanHardwareObject` containers that reference the `CanCommunicationController` which has its `CanControllerId` set to 0.

▶ TRANSMIT `CanHardwareObject` containers that reference the `CanCommunicationController` which has its `CanControllerId` set to 1.

▶ ...

▶ TRANSMIT `CanHardwareObject` containers that reference the `CanCommunicationController` which has its `CanControllerId` set to `<n - 1>`.

3.4.3. CanIf

Configuration parameters	Mapping description
`CanIfCtrlDrvCfg`	If no `CanIfCtrlDrvCfg` container exists, a new one named `<PRE-FIX>_CanIfCtrlDrvCfg` is created. Otherwise the first existing container is used and the following parameters are set: `CanIfCtrlDrvNameRef` references the `CanGeneral` container in the Can driver module configuration. `CanIfCtrlDrvInitHohConfigRef` references the `CanIfInitHohCfg` container created during this import.

Configuration parameters	Mapping description
`CanIfCtrl-DrvCfg/CanIfCtrlCfg`	For every `CAN-COMMUNICATION-CONTROLLER` connected to a `CAN-CLUSTER` or `J-1939-CLUSTER` of the imported `ECU-INSTANCE`, a `CanIfCtrlCfg` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `CAN-COMMUNICATION-CONTROLLER`. `CanIfCtrlCanCtrlRef` references the corresponding `CanController` container in the Can driver module configuration.
`CanIfInitCfg`	If no `CanIfInitCfg` container exists, a new one named `CanIfInitCfg` is created. Otherwise the first existing container is used and the following sub containers are added:
`CanIfInitCfg/CanIfInitHohCfg`	`CanIfInitRefCfgSet` references the `CanConfigSet` container in the Can driver module configuration.
`CanIfInitCfg/CanIfInitHohCfg/CanIfHrhCfg`	If **Buffer Assignment in Can/CanIf** has been set to **Create default buffer assignment** during the import, `CanIfHrhCfg` containers are created for each `CAN-COMMUNICATION-CONTROLLER` connected to a `CAN-CLUSTER` or `J-1939-CLUSTER` of the imported `ECU-INSTANCE`. The container name depends on the `CAN-FRAME-RX-BEHAVIOR` and on the `CAN-ADDRESSING-MODE` of the `FRAME-TRIGGERING` associated with the `PDU` which the `ECU-INSTANCE` receives. The naming schema for every combination is displayed in [Table 3.1, "Rx HOH container names depending on `CAN-FRAME-RX-BEHAVIOR` and `CAN-ADDRESSING-MODE`"](#). `CanIfHrhCanCtrlIdRef` references the `CanIfCtrlCfg` container created for the `CAN-COMMUNICATION-CONTROLLER` of the HOH. `CanIfHrhIdSymRef` references the corresponding `CanHardwareObject` container in the Can driver module configuration.
`CanIfInitCfg/CanIfInitHohCfg/CanIfHthCfg`	If **Buffer Assignment in Can/CanIf** has been set to **Create default buffer assignment** during the import, `CanIfHthCfg` containers are created for each `CAN-COMMUNICATION-CONTROLLER` connected to a `CAN-CLUSTER` or `J-1939-CLUSTER` of the imported `ECU-INSTANCE`. The container name depends on the `CAN-FRAME-TX-BEHAVIOR` and on the `CAN-ADDRESSING-MODE` of the `FRAME-TRIGGERING` associated with the `PDU` which the `ECU-INSTANCE` sends. The naming schema for every combination is displayed in [Table 3.2, "Tx HOH container names depending on `CAN-FRAME-TX-BEHAVIOR` and `CAN-ADDRESSING-MODE`"](#). `CanIfHthCanCtrlIdRef` references the `CanIfCtrlCfg` container created for the `CAN-COMMUNICATION-CONTROLLER` of the HOH.

Configuration parameters	Mapping description
	`CanIfHthIdSymRef` references the corresponding `CanHardwareObject` container in the Can driver module configuration.

`CanIfInitCfg/`
`CanIfRxPduCfg`
	For every PDU received or routed (see Section 3.3.6, "PDU routing") by the imported `ECU-INSTANCE` a `CanIfRxPduCfg` container is created.
	
	The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU.
	
	The following kinds of PDUs are excluded from `CanIfRxPduCfg` container creation:
	
	  ▶ PDUs that are referenced in `DYNAMIC-PARTS` or `STATIC-PARTS` of `MULTIPLEXED-I-PDU` elements.
	
	  ▶ PDUs that are received within `CONTAINER-I-PDU` elements.
	
	If multiple `NM-PDU` elements are received via one `CAN-CLUSTER` or `J-1939-CLUSTER`, a `CanIfRxPduCfg` container is only created for the PDU with the lowest CAN-ID (defined either as `IDENTIFIER` or via `RX-IDENTIFIER-RANGE`). The range of CAN-IDs of all these `NM-PDU` elements is stored under `CanIfRxPduCanIdRange`.
	
	If the `FRAME-TRIGGERING` to which the PDU belongs defines an `RX-IDENTIFIER-RANGE` or the PDU is received via different `FRAME-TRIGGERING` elements yielding two or more CAN-ID values, a `CanIfRxPduCanIdRange` container is created as well.
	
	`CanIfRxPduCanId` is set to the CAN-ID of the PDU for which this `CanIfRxPduCfg` container has been created, if no `CanIfRxPduCanIdRange` container has been created in the previous step. Otherwise this parameter is not configured.
	
	`CanIfRxPduCanIdMask` is set to the `RX-MASK` value of the `CAN-FRAME-TRIGGERING` that receives the `CAN-FRAME` which contains the PDU.
	
	`CanIfRxPduCanIdType` is set depending on the values of `CAN-ADDRESSING-MODE` and `CAN-FRAME-RX-BEHAVIOR`. If `CAN-FRAME-RX-BEHAVIOR` is not available, `CAN-20` is assumed as default value:
	
	  ▶ `STANDARD_NO_FD_CAN` for `STANDARD` and `CAN-20`
	
	  ▶ `EXTENDED_NO_FD_CAN` for `EXTENDED` and `CAN-20`
	
	  ▶ `STANDARD_FD_CAN` for `STANDARD` and `CAN-FD`

Configuration parameters	Mapping description
	► EXTENDED_FD_CAN for EXTENDED and CAN-FD ► STANDARD_CAN for STANDARD and ANY ► EXTENDED_CAN for EXTENDED and ANY CanIfRxPduHrhIdRef references the CanIfHrhCfg container created for CAN-FRAME-TRIGGERING elements with these CAN-ADDRESSING-MODE and CAN-FRAME-RX-BEHAVIOR values (see Section 3.4.2, "Can", CanHardwareObject). If CAN-ADDRESSING-MODE is undefined, CanIfRxPduHrhIdRef is not configured either. CanIfRxPduRef references the corresponding container in the EcuC module configuration. CanIfRxPduUserRxIndicationUL is set depending on the type of PDU: ► CAN_NM for NM-PDU elements that are received in CAN-CLUSTER elements. ► J1939NM for NM-PDU elements that are received in J-1939-CLUSTER elements. ► CAN_TP for N-PDU elements that are received in CAN-CLUSTER elements, and not routed via gateway. ► J1939TP for N-PDU elements that are received in J-1939-CLUSTER elements, and not routed via gateway. ► CAN_TSYN for GENERAL-PURPOSE-PDU elements with its CATEGORY parameter set to GLOBAL_TIME. ► PDUR for I-SIGNAL-PDU elements, MULTIPLEXED-I-PDU elements, CONTAINER-I-PDU elements, GENERAL-PURPOSE-I-PDU elements, DCM-I-PDU elements, for USER-DEFINED-I-PDU elements, which either have a CATEGORY other than XCP or are routed, and for N-PDU elements that are routed (see Section 3.3.6, "PDU routing"). ► XCP for XCP-PDU elements. ► CDD for received USER-DEFINED-I-PDU elements that have their CATEGORY set to XCP and for all other PDU types.
CanIfInitCfg/ CanIfRxP- duCfg/CanIfRxPdu- CanIdRange	CanIfRxPduCanIdRangeLowerCanId is set to the CAN-ID of the PDU for which this CanIfRxPduCfg container has been created, or to LOWER-CAN-ID of the RX-IDENTIFIER-RANGE.

Configuration parameters	Mapping description
	`CanIfRxPduCanIdRangeUpperCanId` is set to the highest CAN-ID of all `NM-PDU` elements received via the `CAN-CLUSTER` or `J-1939-CLUSTER`, or to `UPPER-CAN-ID` of the `RX-IDENTIFIER-RANGE`. For `CanIfRxPduCanIdRange` elements created for received `NM-PDU` elements a consistency check is performed: In case a `CAN-NM-NODE` belonging to the imported `ECU-INSTANCE` exists, which defines a different range via `CAN-NM-RANGE-CONFIG`, this range overrides the range determined via the `NM-PDU` elements. If this range is narrower than the range as determined via the `NM-PDU` elements, a warning is reported.
`CanIfInitCfg/` `CanIfTxPduCfg`	For every PDU sent or routed (see Section 3.3.6, "PDU routing") by the imported `ECU-INSTANCE` a `CanIfTxPduCfg` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU. The following kinds of PDUs are excluded from `CanIfTxPduCfg` container creation: ▸ PDUs that are referenced in `DYNAMIC-PARTS` or `STATIC-PARTS` of `MULTIPLEXED-I-PDU` elements. ▸ PDUs that are sent within `CONTAINER-I-PDU` elements. If a sent `N-PDU` is referenced by two `CAN-TP-CONNECTION` elements, a second `CanIfTxPduCfg` container is created. The Tx `CAN-TP-CONNECTION` must reference this `N-PDU` via `DATA-PDU-REF`, the Rx `CAN-TP-CONNECTION` via `FLOW-CONTROL-PDU-REF`. The direction of the `CAN-TP-CONNECTION` is determined by the PDU referenced by `TP-SDU-REF`. The container name of the second `CanIfTxPduCfg` container is `<PREFIX><name>_D<INSTSUFFIX>`. `CanIfTxPduDlc` is configured as described in Section 3.3.7, "PDU length calculation". `CanIfTxPduCanId` is set to the CAN-ID of the PDU for which this `CanIfTxPduCfg` container has been created. `CanIfTxPduCanIdMask` is set to the `TX-MASK` value of the `CAN-FRAME-TRIGGERING` that sends the `CAN-FRAME` which contains the PDU. `CanIfTxPduCanIdType` is set depending on the values of `CAN-ADDRESSING-MODE` and `CAN-FRAME-TX-BEHAVIOR`. If `CAN-FRAME-TX-BEHAVIOR` is not available, `CAN-20` is assumed as default value:

Configuration parameters	Mapping description
	`CanIfTxPduBufferRef` references the `CanIfBufferCfg` container created for `CAN-FRAME-TRIGGERING` elements with these `CAN-ADDRESSING-MODE` and `CAN-FRAME-TX-BEHAVIOR` values. ▶ `STANDARD_CAN` for `STANDARD` and `CAN-20` ▶ `EXTENDED_CAN` for `EXTENDED` and `CAN-20` ▶ `STANDARD_FD_CAN` for `STANDARD` and `CAN-FD` ▶ `EXTENDED_FD_CAN` for `EXTENDED` and `CAN-FD` `CanIfTxPduHthIdRef` references the `CanIfHthCfg` container created for `CAN-FRAME-TRIGGERING` elements with these `CAN-ADDRESSING-MODE` and `CAN-FRAME-TX-BEHAVIOR` values (see Section 3.4.2, "Can", `CanHardwareObject`). If `CAN-ADDRESSING-MODE` is undefined, `CanIfTxPduHthIdRef` is not configured either. `CanIfTxPduRef` references the corresponding container in the `EcuC` module configuration. `CanIfTxPduUserTxConfirmationUL` is set depending on the type of PDU: ▶ `CAN_NM` for `NM-PDU` elements that are sent in `CAN-CLUSTER` elements. ▶ `J1939NM` for `NM-PDU` elements that are sent in `J-1939-CLUSTER` elements. ▶ `CAN_TP` for `N-PDU` elements that are sent in `CAN-CLUSTER` elements, and not routed via gateway. ▶ `J1939TP` for `N-PDU` elements that are sent in `J-1939-CLUSTER` elements, and not routed via gateway. ▶ `CAN_TSYN` for `GENERAL-PURPOSE-PDU` elements with its `CATEGORY` parameter set to `GLOBAL_TIME`. ▶ `PDUR` for `I-SIGNAL-PDU` elements, `MULTIPLEXED-I-PDU` elements, `CONTAINER-I-PDU` elements, `GENERAL-PURPOSE-I-PDU` elements, `DCM-I-PDU` elements, for `USER-DEFINED-I-PDU` elements, which either have a `CATEGORY` other than `XCP` or are routed, and for `N-PDU` elements that are routed. For more information, see Section 3.3.6, "PDU routing". ▶ `CDD` for sent `USER-DEFINED-I-PDU` elements that have their `CATEGORY` set to `XCP` and for all other PDU types.

Configuration parameters	Mapping description
`CanIfInitCfg/CanIf-BufferCfg`	If **Buffer Assignment in Can/CanIf** was set to **Create default buffer assignment** during the import, `CanIfBufferCfg` containers are created for each `CAN-COMMUNICATION-CONTROLLER` connected to a `CAN-CLUSTER` or `J-1939-CLUSTER` of the imported `ECU-INSTANCE`. The container name depends on the `CAN-FRAME-TX-BEHAVIOR` and on the `CAN-ADDRESSING-MODE` of the `FRAME-TRIGGERING` associated with the `PDU` which the `ECU-INSTANCE` sends. The naming schema for every combination is displayed in [Table 3.2, "Tx HOH container names depending on `CAN-FRAME-TX-BEHAVIOR` and `CAN-ADDRESSING-MODE"`. `CanIfBufferHthRef` references the `CanIfInitHohCfg/CanIfHthCfg` container created for the `CAN-FRAME-TRIGGERING` elements with these `CAN-ADDRESSING-MODE` and `CAN-FRAME-TX-BEHAVIOR` values.

3.4.4. CanNm

Configuration parameters	Mapping description
`CanNmGlobalConfig`	`CanNmComUserDataSupport` is set to `true` if `CanNmRxUserDataPduRef` or `CanNmTxUserDataPduRef` is set for any `CanNmChannelConfig`, or if any `CAN-NM-CLUSTER` linked to the imported `ECU-INSTANCE` has its `NM-PNC-PARTICIPATION` either not defined or set to `true`. Otherwise `CanNmComUserDataSupport` is set to `false`. `CanNmPnResetTime` is set to `PN-RESET-TIME` of the configured `ECU-INSTANCE`. The following parameters are set using the first `NM-ECU` of the imported `ECU-INSTANCE`: `CanNmUserDataEnabled` is set to `NM-USER-DATA-ENABLED`. `CanNmRemoteSleepIndEnabled` is set to `NM-REMOTE-SLEEP-IND-ENABLED`. `CanNmBusSynchronizationEnabled` is set to `NM-BUS-SYNCHRONIZATION-ENABLED`. `CanNmStateChangeIndEnabled` is set to `NM-STATE-CHANGE-IND-ENABLED`.

Configuration parameters	Mapping description
	`CanNmPassiveModeEnabled` is set to `NM-PASSIVE-MODE-ENABLED`. If this parameter is not available, `CanNmPassiveModeEnabled` is set depending on `NM-PASSIVE-MODE-ENABLED` of all `NM-NODES` belonging to the imported `ECU-INSTANCE`: ▶ `false` if at least one `NM-NODE` has `NM-PASSIVE-MODE-ENABLED` set to `false`. ▶ `true` if all `NM-NODE` elements have `NM-PASSIVE-MODE-ENABLED` set to `true`. Otherwise `CanNmPassiveModeEnabled` is not set. `CanNmPduRxIndicationEnabled` is set to `NM-PDU-RX-INDICATION-ENABLED`. `CanNmComControlEnabled` is set to `NM-COM-CONTROL-ENABLED`. `CanNmMainFunctionPeriod` is set to `NM-CYCLETIME-MAIN-FUNCTION`. If inconsistencies are detected among parameters of multiple `NM-ECU` elements, a warning is reported. The following parameters are set using the first `CAN-NM-CLUSTER-COUPLING` of all `CAN-NM-CLUSTER` elements connected to the imported `ECU-INSTANCE`: `CanNmBusLoadReductionEnabled` is set to `NM-BUS-LOAD-REDUCTION-ENABLED`. `CanNmImmediateRestartEnabled` is set to `NM-IMMEDIATE-RESTART-ENABLED`. If inconsistencies are detected among parameters of multiple `CAN-NM-CLUSTER-COUPLING` elements, a warning is reported. If a `CAN-NM-CLUSTER` configured as partial networking cluster (PNC) (see [Section 3.4.14, "EcuC"](#)) belongs to the imported `ECU-INSTANCE`, the following parameters are set: `CanNmPnEiraRxNSduRef` references the corresponding container in the `EcuC` module configuration. `CanNmPnEiraCalcEnabled` is set to `true`.

Configuration parameters	Mapping description
	`CanNmPnInfo/CanNmPnInfoOffset` is set to `PNC-VECTOR-OFFSET`. If `PNC-VECTOR-OFFSET` is not defined, a warning is issued and `CanNmPnInfoOffset` is not set. `CanNmPnInfo/CanNmPnInfoLength` is set to `PNC-VECTOR-LENGTH`. If `PNC-VECTOR-LENGTH` is not defined, a warning is issued and `CanNmPnInfoLength` is not set. The values for `CanNmPnFilterMaskByte` are calculated as follows: In a first step the `PNC-WAKEUP-DATA-MASK` values of all `CAN-COMMUNICATION-CONNECTOR` elements that belong to the imported `ECU-INSTANCE` and a `PHYSICAL-CHANNEL` of the `CAN-NM-CLUSTER`'s `CAN-CLUSTER` are retrieved. In the PNC vector which is a 64 bit entity, all bits that lie on any collected `PNC-WAKEUP-DATA-MASK` position are set to `true`, while all other bits are set to `false`. The resulting value is shifted to the right by `PNC-VECTOR-OFFSET * 8` bit positions and thus removing the leading empty bits in the PNC vector. Finally, the value is converted into an n-byte byte array, (n corresponding to `PNC-VECTOR-LENGTH`). A `CanNmPnFilterMaskByte` container is created for each byte. `CanNmPnFilterMaskByteValue` is set to the value of the byte and `CanNmPnFilterMaskByteIndex` to the position of the byte within the array. In this array `CanNmPnFilterMaskByteIndex` of the PNC vector LSB is set to 0 and the `CanNmPnFilterMaskByteIndex` of the MSB is set to `PNC-VECTOR-LENGTH - 1`.
`CanNmChannelConfig`	For every `CAN-NM-CLUSTER` which belongs to the imported `ECU-INSTANCE`, and over which at least one `NM-PDU` is sent or received, a `CanNmChannelConfig` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `CAN-NM-CLUSTER`. A `CAN-NM-CLUSTER` belongs to the imported `ECU-INSTANCE` if at least one of its `CAN-NM-NODE` elements references a `CAN-COMMUNICATION-CONTROLLER` of this `ECU-INSTANCE`. The `NM-ECU` used for configuring some of the `CanNmChannelConfig` parameters is the `NM-ECU` which the first `CAN-NM-NODE` references via `NM-IF-ECU-REF`.

Configuration parameters	Mapping description
	In case the `ECU-INSTANCE` does not send or receive any `NM-PDU` on the `CAN-NM-CLUSTER`, or it sends multiple `NM-PDU` elements on the `CAN-NM-CLUSTER`, a warning is reported. `CanNmNodeDetectionEnabled` is set to `NM-CLUSTER/NM-NODE-DETECTION-ENABLED`, or to `NM-ECU/NM-NODE-DETECTION-ENABLED` if `NM-CLUSTER/NM-NODE-DETECTION-ENABLED` is not available. `CanNmNodeIdEnabled` is set to `NM-CLUSTER/NM-NODE-ID-ENABLED`, or to `NM-ECU/NM-NODE-ID-ENABLED` if `NM-CLUSTER/NM-NODE-ID-ENABLED` is not available. `CanNmRepeatMsgIndEnabled` is set to `NM-CLUSTER/NM-REPEAT-MSG-IND-ENABLED`, or to `NM-ECU/NM-REPEAT-MSG-IND-ENABLED` if `NM-CLUSTER/NM-REPEAT-MSG-IND-ENABLED` is not available. `CanNmBusLoadReductionActive` is set to `NM-BUSLOAD-REDUCTION-ACTIVE`. `CanNmTimeoutTime` is set to `NM-NETWORK-TIMEOUT`. `CanNmWaitBusSleepTime` is set to `NM-WAIT-BUS-SLEEP-TIME`. `CanNmRepeatMessageTime` is set to `NM-REPEAT-MESSAGE-TIME`. `CanNmRemoteSleepIndTime` is set to `NM-REMOTE-SLEEP-INDICATION-TIME`. `CanNmUserDataLength` is calculated by the formula `<PayloadLength>` - `<NonUserDataLength>` where `<PayloadLength>` is calculated for the `NM-PDU` as described in [Section 3.3.7, "PDU length calculation"](#) and `<NonUserDataLength>` denotes the byte length of the non-user data part of the `NM-PDU` which is calculated by `max(CAN-NM-CLUSTER/NM-CBV-POSITION, CAN-NM-CLUSTER/NM-NID-POSITION) + 1`. If neither `CAN-NM-CLUSTER/NM-CBV-POSITION` nor `CAN-NM-CLUSTER/NM-NID-POSITION` are defined, `<NonUserDataLength>` is zero. `CanNmMsgCycleTime` is set to `NM-MSG-CYCLE-TIME`. `CanNmMsgTimeoutTime` is set to `NM-MESSAGE-TIMEOUT-TIME`. `CanNmImmediateNmTransmissions` is set to `NM-IMMEDIATE-NM-TRANSMISSIONS`.

Configuration parameters	Mapping description
	`CanNmImmediateNmCycleTime` is set to `NM-IMMEDIATE-NM-CYCLE-TIME`. `CanNmComMNetworkHandleRef` references the `ComMChannel` container that is created for the `COMMUNICATION-CLUSTER` referenced in `COMMUNI-CATION-CLUSTER-REF`. `CanNmPduNidPosition` is set depending on the value of `NM-NID-POSITION`: ▶ `CANNM_PDU_BYTE_0` for `0`. ▶ `CANNM_PDU_BYTE_1` for `1`. ▶ `CANNM_PDU_OFF` for any other value. `CanNmPduCbvPosition` is set depending on the value of `NM-CBV-POSITION`: ▶ `CANNM_PDU_BYTE_0` for `0`. ▶ `CANNM_PDU_BYTE_1` for `1`. ▶ `CANNM_PDU_OFF` for any other value. `CanNmRxPdu/<EcuC container name>/CanNmRxPduRef` references the container in the `EcuC` module which has been created for the `NM-PDU` referenced via `RX-NM-PDU-REFS/RX-NM-PDU-REF`, which has the lowest CAN-ID assigned. `<EcuC container name>` is the name of the referenced `EcuC` container. If an `NmUserDataPdu` container has been created for the received `NM-PDU` in the `EcuC` module configuration (see Section 3.4.14, "EcuC"), `CanNmUserDataRxPdu/CanNmRxUserDataPduRef` references this container. The name of `CanNmUserDataRxPdu` is set to the name of the referenced `EcuC` container. `CanNmTxPdu/CanNmTxPduRef` references the container in the `EcuC` module which has been created for the first `NM-PDU` referenced via `TX-NM-PDU-REFS/TX-NM-PDU-REF`. The name of `CanNmTxPdu` is set to the name of the referenced `EcuC` container. If an `NmUserDataPdu` container has been created for the sent `NM-PDU` in the `EcuC` module configuration, see Section 3.4.14, "EcuC", `CanNmUserDataTxPdu/CanNmTxUserDataPduRef` references this container. The name of `CanNmUserDataTxPdu` is set to the name of the referenced `EcuC` container. If no Rx or Tx `NM-PDU` elements are referenced, or if multiple Tx `NM-PDU` elements are referenced, a warning is reported.

Configuration parameters	Mapping description
	If an ERA PDU container has been created for the `CAN-NM-CLUSTER` in the `EcuC` module (see [Section 3.4.14, "EcuC"](#)), `CanNmPnEraRxNSduRef` references the `EcuC` container and `CanNmPnEraCalcEnabled` is set to `true`. The following parameters are set using the first `NM-NODE` of the `CAN-NM-CLUSTER` connected to the imported `ECU-INSTANCE`. If inconsistencies are detected among parameters of multiple `NM-NODE` elements, a warning is reported. `CanNmMsgCycleOffset` is set to `NM-MSG-CYCLE-OFFSET`. `CanNmMsgReducedTime` is set to `NM-MSG-REDUCED-TIME`. `CanNmNodeId` is set to `NM-NODE-ID`. `CanNmPnEnabled` is set to `true` if the `CAN-NM-CLUSTER` is PNC-enabled, or if it has its `NM-PNC-PARTICIPATION` either not defined or set to `true`. Otherwise `CanNmPnEnabled` is set to `false`. If the `CAN-NM-CLUSTER` is PNC-enabled, the following parameters are set: `CanNmCarWakeUpRxEnabled` is set to `NM-CAR-WAKE-UP-RX-ENABLED`. `CanNmCarWakeUpFilterEnabled` is set to `NM-CAR-WAKE-UP-FILTER-EN-ABLED`. `CanNmCarWakeUpFilterNodeId` is set to `NM-CAR-WAKE-UP-FIL-TER-NODE-ID`. `CanNmCarWakeUpBitPosition` is set to `NM-CAR-WAKE-UP-BIT-POSITION` mod 8. `CanNmCarWakeUpBytePosition` is set to `NM-CAR-WAKE-UP-BIT-POSITION` div 8.

3.4.5. CanSM

Configuration parameters	Mapping description
`CanSMConfiguration/CanSMManagerNetwork`	For every `CAN-CLUSTER` which belongs to the imported `ECU-INSTANCE`, a `CanSMManagerNetwork` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `CAN-CLUSTER`.

Configuration parameters	Mapping description
	CanSMComMNetworkHandleRef references the ComMChannel container in the ComM module configuration, which has been created for the CAN-CLUSTER.
CanSMConfiguration/CanSMManagerNetwork/CanSMController	For every CAN-COMMUNICATION-CONTROLLER that is connected to the CAN-CLUSTER and to the imported ECU-INSTANCE, a CanSMController container is created. The container name is <PREFIX><name>, where <name> is the SHORT-NAME of the CAN-COMMUNICATION-CONTROLLER. CanSMControllerId references the CanIfCtrlCfg container in the CanIf module configuration, which has been created for the CAN-COMMUNICATION-CONTROLLER.

3.4.6. CanTp

Configuration parameters	Mapping description
CanTpConfig	CanTpMainFunctionPeriod is set to CYCLE-TIME-MAIN-FUNCTION of the first CAN-TP-ECU which belongs to the imported ECU-INSTANCE. If inconsistencies are detected among the CYCLE-TIME-MAIN-FUNCTION values of multiple CAN-TP-ECU elements of the imported ECU-INSTANCE, a warning is reported.
CanTpConfig/CanTpChannel	For each CAN-TP-CHANNEL referenced by a CAN-TP-CONNECTION which belongs to the imported ECU-INSTANCE, a CanTpChannel container is created. The container name is <PREFIX><name>, where <name> is the SHORT-NAME of the CAN-TP-CHANNEL. A CAN-TP-CONNECTION belongs to the imported ECU-INSTANCE if at least one transmitting or receiving CAN-TP-NODE of this CAN-TP-CONNECTION references a COMMUNICATION-CONNECTOR which is also referenced by the imported ECU-INSTANCE. CanTpChannelMode is set depending on the value of CHANNEL-MODE: ▶ CANTP_MODE_FULL_DUPLEX for FULLDUPLEXMODE. ▶ CANTP_MODE_HALF_DUPLEX for HALFDUPLEXMODE.
CanTpConfig/CanTpChannel/CanTpRxNSdu	For every CAN-TP-CONNECTION referencing the current CAN-TP-CHANNEL, a CanTpRxNSdu container is created, if the PDU referenced via TP-SDU-REF is received or routed (see Section 3.3.6, "PDU routing") by the imported ECU-INSTANCE. The container name is <PREFIX><name>, where <name> is the SHORT-NAME of the referenced PDU.

Configuration parameters	Mapping description
	`CanTpRxNSduRef` references the container in the EcuC module configuration, created for the referenced PDU.
	`CanTpNTa/CanTpNTa` is set to `TP-ADDRESS` of the `CAN-TP-ADDRESS` referenced via `MULTICAST-REF`. If no `MULTICAST-REF` exists, the `CAN-TP-ADDRESS` of the first `CAN-TP-NODE` (if more than one exists, a warning is issued) referenced via `RECEIVER-REFS` is used. Furthermore, `CanTpNTa` is only set if `ADDRESSING-FORMAT` of the `CAN-TP-CONNECTION` is set to `EXTENDED`.
	`CanTpNSa/CanTpNSa` is set to `TP-ADDRESS` of the `CAN-TP-ADDRESS` referenced by the `CAN-TP-NODE` referenced via `TRANSMITTER-REF`. `CanTpNSa` is only set if `ADDRESSING-FORMAT` of the `CAN-TP-CONNECTION` is set to `EXTENDED`.
	`CanTpNAe/CanTpNAe` is set to `TP-ADDRESS-EXTENSION-VALUE` of the `CAN-TP-ADDRESS` referenced via `MULTICAST-REF`. If no `MULTICAST-REF` exists, the `CAN-TP-ADDRESS` of the first `CAN-TP-NODE` referenced via `RECEIVER-REFS` is used. Additionally the same `TP-ADDRESS-EXTENSION-VALUE` must exist for the `CAN-TP-NODE` referenced via `TRANSMITTER-REF`. Otherwise a warning is issued and the parameter is not set. This parameter is only configured if `ADDRESSING-FORMAT` of the `CAN-TP-CONNECTION` is set to `MIXED`.
	`CanTpRxPaddingActivation` is set to `CANTP_ON` if `PADDING-ACTIVATION` is set to `true`, or to `CANTP_OFF` if `PADDING-ACTIVATION` is set to `false`.
	`CanTpRxTaType` is set depending on the value of `TA-TYPE`:
	► `CANTP_FUNCTIONAL` for `CANTP-FUNCTIONAL`.
	► `CANTP_PHYSICAL` for `CANTP-PHYSICAL`.
	If `TA-TYPE` is not available `CanTpRxTaType` is set to `CANTP_FUNCTIONAL` if a `CAN-TP-ADDRESS` referenced via `MULTICAST-REF` exists, or if more than one `CAN-TP-CONNECTION` elements references the same PDU via `TP-SDU-REF`. In all other cases, it is set to `CANTP_PHYSICAL`.
	`CanTpRxAddressingFormat` is set depending on the value of `ADDRESSING-FORMAT`:
	► `CANTP_STANDARD` for `STANDARD`.
	► `CANTP_EXTENDED` for `EXTENDED`.
	► `CANTP_MIXED` for `MIXED`.

Configuration parameters	Mapping description
	`CanTpBs` is set to `MAX-BLOCK-SIZE`. `CanTpNbr` is set to `TIMEOUT-BR`. `CanTpNcr` is set to `TIMEOUT-CR`. If no `MULTICAST-REF` exists, and if exactly one `CAN-TP-NODE` is referenced via `RECEIVER-REFS`, the following parameters are set using this `CAN-TP-NODE`. `CanTpRxWftMax` is set to `MAX-FC-WAIT`. `CanTpNar` is set to `TIMEOUT-AR`. `CanTpSTmin` is set to `ST-MIN` .
`CanTpConfig/CanTpChannel/CanTpRxNSdu/CanTpRxNPdu`	A `CanTpRxNPdu` container is created for the `N-PDU` referenced via `DATA-PDU-REF`. If no `DATA-PDU-REF` exists, a warning is issued. The name of `CanTpRxNPdu` is set to `<SDU name>_<PREFIX><name><INSTSUFFIX>`, where `<SDU name>` is the name of the parent `CanTpRxNSdu` container and `<name>` is the `SHORT-NAME` of the `N-PDU`. `CanTpRxNPduRef` references the corresponding container in the `EcuC` module configuration.
`CanTpConfig/CanTpChannel/CanTpRxNSdu/CanTpTxFcNPdu`	A `CanTpTxFcNPdu` container is created for the `N-PDU` referenced via `FLOW-CONTROL-PDU-REF`. The name of `CanTpTxFcNPdu` is set to `<SDU name>_<PREFIX><name><INSTSUFFIX>`, where `<SDU name>` is the name of the parent `CanTpRxNSdu` container and `<name>` is the `SHORT-NAME` of the `N-PDU`. `CanTpTxFcNPduRef` references the corresponding container in the `EcuC` module configuration.
`CanTpConfig/CanTpChannel/CanTpTxNSdu`	For every `CAN-TP-CONNECTION` referencing the current `CAN-TP-CHANNEL`, a `CanTpTxNSdu` container is created, if the PDU referenced via `TP-SDU-REF` is sent or routed (see [Section 3.3.6, "PDU routing"](#)) by the imported `ECU-INSTANCE`. The container name is `<SDU name>_<PREFIX><name><INSTSUFFIX>`, where `<SDU name>` is the name of the parent `CanTpTxNSdu` container and `<name>` is the `SHORT-NAME` of the referenced PDU. In case multiple `CAN-TP-CONNECTION` elements refer to the same PDU via `TP-SDU-REF`, only one `CanTpTxNSdu` container is created and the following parameters of the `CAN-TP-CONNECTION` are checked for consistency: `DATA-PDU-REF`, `FLOW-CONTROL-PDU-REF`, `ADDRESSING-FORMAT`, `PADDING-ACTI-`

Configuration parameters	Mapping description
	VATION, TRANSMITTER-REF. If inconsistencies are detected a warning is issued and the first CAN-TP-CONNECTION is used for the subsequent operations. CanTpTxNSduRef: see CanTpRxNSdu/CanTpRxNSduRef. CanTpNTa: see CanTpRxNSdu/CanTpNTa. If multiple CAN-TP-CONNECTION elements refer to the same PDU via TP-SDU-REF, and the TP-ADDRESS values of the first CAN-TP-NODE of each CAN-TP-CONNECTION referenced via RE-CEIVER-REF differ, CanTpNTa is not set. CanTpNSa: see CanTpRxNSdu/CanTpNSa. CanTpNAe: see CanTpRxNSdu/CanTpNAe. CanTpTxPaddingActivation: see CanTpRxNSdu/CanTpRxPaddingActivation. CanTpTxTaType: see CanTpRxNSdu/CanTpRxTaType. CanTpNbs is set to TIMEOUT-BS. CanTpNcs is set to TIMEOUT-CS. CanTpTc is set to CANCELLATION. CanTpNas is set to TIMEOUT-AS of the CAN-TP-NODE referenced via TRANSMITTER-REF.
CanTpConfig/CanTpChannel/CanTpTxNSdu/CanTpTxNPdu	See CanTpRxNSdu/CanTpRxNPdu. In case a received N-PDU is referenced by two CAN-TP-CONNECTION elements of the imported ECU-INSTANCE (either via DATA-PDU-REF by the CAN-TP-CONNECTION referencing a Tx PDU via TP-SDU-REF, or via FLOW-CON-TROL-PDU-REF by the CAN-TP-CONNECTION referencing an Rx PDU via TP-SDU-REF), CanTpTxNPduRef references the duplicate container in the EcuC module configuration (container name: <PREFIX><name>_D<INSTSUFFIX>).
CanTpConfig/CanTpChannel/CanTpTxNSdu/CanTpRxFcNPdu	See CanTpRxNSdu/CanTpTxFcNPdu.

3.4.7. CanTSyn

Configuration parameters	Mapping description
`CanTSynGlobalTime-Domain`	One `CanTSynGlobalTimeDomain` container is created for each `GLOBAL-TIME-DOMAIN` that contains a `GLOBAL-TIME-CAN-MASTER` or a `GLOBAL-TIME-CAN-SLAVE` element that references a `CAN-COMMUNI-CATION-CONNECTOR` of the imported `ECU-INSTANCE`. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `GLOBAL-TIME-DOMAIN`. `CanTSynGlobalTimeDomainId` is set to `DOMAIN-ID`. `CanTSynGlobalTimeNetworkSegmentId` is set to `NETWORK-SEGMENT-ID/NETWORK-SEGMENT-ID`. `CanTSynSynchronizedTimeBaseRef` references the corresponding `StbMSynchronizedTimeBase` container in the `StbM` module. If the `GLOBAL-TIME-DOMAIN` is a subdomain of a parent `GLOBAL-TIME-DOMAIN`, `CanTSynSynchronizedTimeBaseRef` references the `StbMSynchronizedTimeBase` container that has been created for the parent `GLOBAL-TIME-DOMAIN`. For further information about the configuration of the `StbM` module, see [Section 3.4.45, "StbM"](#).
`CanTSynGlobalTime-Domain/CanTSyn-GlobalTimeFup-DataIDList, Can-TSynGlobalTimeDo-main/CanTSynGlobal-TimeOfnsDataIDList, CanTSynGlobalTime-Domain/CanTSyn-GlobalTimeOfs-DataIDList, Can-TSynGlobalTimeDo-main/CanTSynGlobal-TimeSyncDataIDList`	The `CAN-GLOBAL-TIME-DOMAIN-PROPS` element of the CAN `GLOBAL-TIME-DOMAIN` entity is used to retrieve the following subelements to configure container lists: ▶ `FUP-DATA-ID-LIST` ▶ `OFNS-DATA-ID-LIST` ▶ `OFS-DATA-ID-LIST` ▶ `SYNC-DATA-ID-LIST` The configured container lists are the following: ▶ `CanTSynGlobalTimeDomain/CanTSynGlobalTimeFup-DataIDList/CanTSynGlobalTimeFupDataIDListElement` ▶ `CanTSynGlobalTimeDomain/CanTSynGlobalTimeOfns-DataIDList/CanTSynGlobalTimeOfnsDataIDListElement` ▶ `CanTSynGlobalTimeDomain/CanTSynGlobalTimeOfs-DataIDList/CanTSynGlobalTimeOfsDataIDListElement`

Configuration parameters	Mapping description
	▶ `CanTSynGlobalTimeDomain/CanTSynGlobalTimeSync-DataIDList/CanTSynGlobalTimeSyncDataIDListElement` In each container list, one subcontainer is created per element named `Element_<idx>`, where `<idx>` is the zero-based index of the element within the list. In each of these lists, one index parameter and one value parameter are configured. The index parameter represents the zero-based index of the element in the list, the value parameter represents the actual value. {TABLE_BELOW}

ID List	Index Parameter	Value Parameter
`CanTSynGlobalTime-FupDataIDList`	`CanTSynGlobalTime-FupDataIDListIndex`	`CanTSynGlobalTime-FupDataIDListValue`
`CanTSynGlobalTime-OfnsDataIDList`	`CanTSynGlobalTime-OfnsDataIDListIndex`	`CanTSynGlobalTime-OfnsDataIDListValue`
`CanTSynGlobalTime-OfsDataIDList`	`CanTSynGlobalTime-OfsDataIDListIndex`	`CanTSynGlobalTime-OfsDataIDListValue`
`CanTSynGlobal-TimeSyncDataIDList`	`CanTSynGlob-alTimeSync-DataIDListIndex`	`CanTSynGlobal-TimeSyncDataIDList-Value`

Configuration parameters	Mapping description
`CanTSynGlobalTime-Domain/CanTSynGlob-alTimeMaster`	If the `GLOBAL-TIME-CAN-MASTER` of the `GLOBAL-TIME-DOMAIN` references a `CAN-COMMUNICATION-CONNECTOR` of the imported `ECU-INSTANCE`, a `CanTSynGlobalTimeMaster` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `GLOBAL-TIME-CAN-MASTER`. `CanTSynGlobalTimeTxCrcSecured` is set depending on the value of `GLOBAL-TIME-CAN-MASTER/CRC-SECURED`: `CRC_SUPPORTED` for `CRC-SUPPORTED` and `CRC_NOT_SUPPORTED` for `CRC-NOT-SUPPORTED`. `CanTSynGlobalTimeTxPeriod` is set to `GLOBAL-TIME-CAN-MASTER/SYNC-PERIOD`. `CanTSynMasterConfirmationTimeout` is set to `GLOBAL-TIME-CAN-MASTER/SYNC-CONFIRMATION-TIMEOUT`. `CanTSynCyclicMsgResumeTime` is set to `GLOBAL-TIME-CAN-MASTER/IMMEDIATE-RESUME-TIME`. `CanTSynGlobalTimeDebounceTime` is set to `GLOBAL-TIME-DOMAIN/DEBOUNCE-TIME`.

Configuration parameters	Mapping description
`CanTSynGlobalTime-Domain/CanTSynGlobalTimeMaster/Can-TSynGlobalTimeMasterPdu`	A `GLOBAL-TIME-DOMAIN` is associated with a PDU if that PDU is a `GENERAL-PURPOSE-PDU` that has its `CATEGORY` set to `GLOBAL_TIME` and one of the conditions hold: ▶ The `GLOBAL-TIME-DOMAIN` references the PDU in `GLOBAL-TIME-PDU-REF` ▶ The `GLOBAL-TIME-DOMAIN` references a `PDU-TRIGGERING` either via `GLOBAL-TIME-PDU-TRIGGERING-REF` or via `PDU-TRIGGERING-REF`, and that `PDU-TRIGGERING` refers to the PDU If the `GLOBAL-TIME-DOMAIN` is associated with a PDU, and if this is the PDU that the configured `ECU-INSTANCE` sends on the `CAN-CLUSTER` which the `GLOBAL-TIME-DOMAIN` references via `COMMUNICATION-CLUSTER-REF`, one `CanTSynGlobalTimeMasterPdu` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU. `CanTSynGlobalTimePduRef` references the `EcuC` container created for the PDU that is associated with the `GLOBAL-TIME-DOMAIN`.
`CanTSynGlobalTime-Domain/CanTSynGlobalTimeSlave`	If one of the `GLOBAL-TIME-CAN-SLAVE` elements of the `GLOBAL-TIME-DOMAIN` references a `CAN-COMMUNICATION-CONNECTOR` of the imported `ECU-INSTANCE`, a `CanTSynGlobalTimeSlave` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `GLOBAL-TIME-CAN-SLAVE`. `CanTSynRxCrcValidated` is set depending on the value of `GLOBAL-TIME-CAN-SLAVE/CRC-VALIDATED`: `CRC_VALIDATED` for `CRC-VALIDATED`, `CRC_-NOT_VALIDATED` for `CRC-NOT-VALIDATED`, and `CRC_IGNORED` for `CRC-IG-NORED`. `CanTSynGlobalTimeFollowUpTimeout` is set to `GLOBAL-TIME-CAN-SLAVE/FOLLOW-UP-TIMEOUT-VALUE` or to `GLOBAL-TIME-DOMAIN/FOLLOW-UP-TIMEOUT-VALUE` if `GLOBAL-TIME-CAN-SLAVE/FOLLOW-UP-TIME-OUT-VALUE` does not exist. `CanTSynGlobalTimeSequenceCounterJumpWidth` is set to `GLOBAL-TIME-CAN-SLAVE/SEQUENCE-COUNTER-JUMP-WIDTH`.
`CanTSynGlobalTime-Domain/CanTSynGlobalTimeSlave/Can-`	If the `GLOBAL-TIME-DOMAIN` is associated with a PDU, and if this is the PDU that the configured `ECU-INSTANCE` and if this is the PDU that the configured `ECU-INSTANCE` receives on the `CAN-CLUSTER` which the `GLOB-`

Configuration parameters	Mapping description
`TSynGlobal-TimeSlavePdu`	`AL-TIME-DOMAIN` references via `COMMUNICATION-CLUSTER-REF`, one `CanTSynGlobalTimeSlavePdu` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU. `CanTSynGlobalTimePduRef` references the `EcuC` container created for the PDU that is associated with the `GLOBAL-TIME-DOMAIN`.

3.4.8. Com

Configuration parameters	Mapping description
`ComGeneral`	`ComEnableMDTForCyclicTransmission` is set to `COM-ENABLE-MDT-FOR-CYCLIC-TRANSMISSION`.
`ComConfig/ComIPdu`	For every PDU instance (see Section 3.3.1.3, "Instance handling") which represents an `I-SIGNAL-I-PDU` or an `NM-PDU` for which an `NmUserDataPdu` container has been created (see Section 3.4.14, "EcuC"), a `ComIPdu` container is created, except if an `LdCom` module configuration is present in the current project and the conditions described in Section 3.4.34, "LdCom" allow to process the PDU instance in the `LdCom` module. For `I-SIGNAL-I-PDU` elements the container name is `PD<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `I-SIGNAL-I-PDU`. For `NM-PDU` elements the container name is `PD<PREFIX><name>_NmComUserData<INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `NM-PDU`. The `ComIPduSignalRef` entries reference all `ComSignal` containers representing signal instances of the PDU instance which lie directly within the PDU (in contrast to group signals embedded in PDUs indirectly via signal groups). See also Section 3.3.1.3, "Instance handling". The `ComIPduSignalGroupRef` entries reference all `ComSignalGroup` containers which have been created for the signal group instances of the PDU instance. `ComIPduDirection` is set to `SEND` if the PDU instance is sent by the imported `ECU-INSTANCE`. Otherwise it is set to `RECEIVE`. `ComIPduType` is set to `TP` if one of the following conditions holds:

Configuration parame-ters	Mapping description
	▶ The `I-SIGNAL-I-PDU` is referenced by a `CAN-TP-CONNECTION` via `TP-SDU-REF` ▶ The `I-SIGNAL-I-PDU` is referenced by a `FLEXRAY-TP-CONNECTION` via `DIRECT-TP-SDU-REF` or via `REVERSED-TP-SDU-REF` ▶ The `I-SIGNAL-I-PDU` is referenced by a `FLEXRAY-AR-TP-CONNECTION` via `DIRECT-TP-SDU-REF` or via `REVERSED-TP-SDU-REF` ▶ The `I-SIGNAL-I-PDU` is referenced by a `LIN-TP-CONNECTION` via `LIN-TP-N-SDU-REF` ▶ The `I-SIGNAL-I-PDU` is referenced by a `PDU-TRIGGERING`, which in turn is referenced by a `SOMEIP-TP-CONNECTION` via `TP-SDU-REF` ▶ An `ETH-TP-CONNECTION` refers to the `PDU-TRIGGERING` that references the transmitted `I-SIGNAL-I-PDU` and that belongs to the `PHYSICAL-CHANNEL` on which the `I-SIGNAL-I-PDU` is transmitted. ▶ One of the `TP-CONNECTION` elements listed above references a `SECURED-I-PDU`, either directly or via a `PDU-TRIGGERING`, and the `SECURED-I-PDU` in turn references the `PDU-TRIGGERING` of the `I-SIGNAL-I-PDU` via `PAYLOAD-REF`. In all other cases `ComIPduType` is set to `NORMAL`. The `ComIPduGroupRef` entries reference all `ComIPduGroup` containers which have been created for the `I-SIGNAL-I-PDU-GROUP` elements that contain the PDU instance's `I-SIGNAL-I-PDU` or `NM-PDU`. A `ComIPduGroupRef` entry is added as long as the `COMMUNICATION-DIRECTION` of the `I-SIGNAL-I-PDU-GROUP` is not opposite to the `ComIPduDirection` of the referencing `I-SIGNAL-I-PDU`. In case the `I-SIGNAL-I-PDU` is not part of any `I-SIGNAL-I-PDU-GROUP`, a reference to the default `ComIPduGroup` (see `ComConfig/ComIPduGroup`) this PDU belongs to is added. `NM-PDU` elements always obtain a reference to the default Nm `ComIPduGroup`, since `I-SIGNAL-I-PDU-GROUP` elements cannot contain `NM-PDU` elements. `ComIPduSignalProcessing` is set to `I-PDU-SIGNAL-PROCESSING` of the `I-PDU-PORT` that connects one of the `COMMUNICATION-CONNECTOR` elements of the `ECU-INSTANCE` to the `PDU-TRIGGERING` of the PDU instance. `ComPduIdRef` references the container created for the `I-SIGNAL-I-PDU` in the `EcuC` module configuration. For `NM-PDU` elements it references the `NmUserDataPdu` container.

Configuration parameters	Mapping description
`ComConfig/ComIPdu/ComTxIPdu`	If the PDU instance is sent by the imported `ECU-INSTANCE`, a `ComTxIPdu` sub container is created. `ComTxIPduUnusedAreasDefault` is set to `UNUSED-BIT-PATTERN`. If `UNUSED-BIT-PATTERN` does not lie within the interval `[0 .. 255]` a warning is reported and `ComTxIPduUnusedAreasDefault` is not configured. `ComMinimumDelayTime` is set to `MINIMUM-DELAY` of the `I-SIGNAL-I-PDU`'s `I-PDU-TIMING`.
`ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode`	For every sent PDU instance, a `ComTxModeTrue/ComTxMode` container is created. For `NM-PDU` elements and for `I-SIGNAL-I-PDU` elements without `TRANSMISSION-MODE-TRUE-TIMING`, `ComTxModeMode` is set to `NONE`. If `TRANSMISSION-MODE-TRUE-TIMING` exists for the PDU instance, `ComTxModeMode` is set depending on the existence of timings attached to the `TRANSMISSION-MODE-TRUE-TIMING`: ▸ `MIXED`, if both `CYCLIC-TIMING` and `EVENT-CONTROLLED-TIMING` exist. ▸ `PERIODIC`, if only a `CYCLIC-TIMING` exists. ▸ `DIRECT`, if only an `EVENT-CONTROLLED-TIMING` exists. ▸ `NONE`, if no timing exists. For `NM-PDU` elements `ComTxModeMode` is always set to `NONE`. `ComTxModeTimePeriod` is set to `TIME-PERIOD/VALUE` of the `CYCLIC-TIMING` if `ComTxModeMode` is set to `MIXED` or `PERIODIC`. `ComTxModeTimeOffset` is set to `TIME-OFFSET/VALUE` of the `CYCLIC-TIMING` if `ComTxModeMode` is set to `MIXED` or `PERIODIC`. `ComTxModeNumberOfRepetitions` is set to (`NUMBER-OF-REPETITIONS` + 1) of the `EVENT-CONTROLLED-TIMING` if `ComTxModeMode` is set to `MIXED` or `DIRECT`. If `NUMBER-OF-REPETITIONS` equals `0`, `ComTxModeNumberOfRepetitions` is also set to `0`. `ComTxModeRepetitionPeriod` is set to `REPETITION-PERIOD/VALUE` of the `EVENT-CONTROLLED-TIMING` if `ComTxModeMode` is set to `MIXED` or `DIRECT` and if `NUMBER-OF-REPETITIONS` of the `EVENT-CONTROLLED-TIMING` is greater than zero.

Configuration parameters	Mapping description
`ComConfig/ComIP-du/ComTxIPdu/ComTx-ModeFalse/ComTxMode`	If a `TRANSMISSION-MODE-FALSE-TIMING` exists for the `I-SIGNAL-I-PDU`, a `ComTxModeFalse/ComTxMode` container is created. For the configuration of the container parameters see `ComTxMode-True/ComTxMode`.
`ComConfig/ComSignal`	For every signal instance which represents an `I-SIGNAL` which is sent or received by the imported `ECU-INSTANCE` and which is not contained in any `I-SIGNAL-GROUP`, a `ComISignal` container is created. A signal instance is considered sent if there is an `I-SIGNAL-PORT` with `COMMUNICATION-DIRECTION` set to `OUT` that connects one of the `COMMUNICATION-CONNECTOR` elements of the imported `ECU-INSTANCE` to the `I-SIGNAL-TRIGGERING` of the signal instance. If `COMMUNICATION-DIRECTION` is set to `IN`, the signal instance is considered received. The container name is `SG<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `I-SIGNAL`. In special cases, dedicated IRA signal containers are added. The configuration of IRA signals is described in Section 3.4.8.2, "Configuration for partial networking IRA signals". `ComTimeout` is set to `TIMEOUT` of the `I-SIGNAL-PORT` that connects one of the `COMMUNICATION-CONNECTOR` elements of the `ECU-INSTANCE` to the `I-SIGNAL-TRIGGERING` of the signal instance. Please note that if an instance of `TRANSMISSION-ACKNOWLEDGEMENT-REQUEST` or `NONQUEUED-RE-CEIVER-COM-SPEC` is related to the `I-SIGNAL`, `ComTimeout` needs to be configured using `TRANSMISSION-ACKNOWLEDGEMENT-REQUEST/TIMEOUT` or `NONQUEUED-RECEIVER-COM-SPEC/ALIVE-TIMEOUT`. This configuration step is executed by the *Service Needs Calculator*, which is part of the `RTE` module. `ComFirstTimeout` is set to `FIRST-TIMEOUT` of the `I-SIGNAL-PORT` that connects one of the `COMMUNICATION-CONNECTOR` elements of the `ECU-INS-TANCE` to the `I-SIGNAL-TRIGGERING` of the signal instance. If the imported `ECU-INSTANCE` receives the `I-SIGNAL` and `I-SIGNAL/DA-TA-TYPE-POLICY` is either not set or set to one of `LEGACY` or `OVERRIDE`, `ComDataInvalidAction` is configured according to the value of `HAN-DLE-INVALID` of the `I-SIGNAL-PORT` that connects one of the `COMMUNI-CATION-CONNECTOR` elements of the `ECU-INSTANCE` to the `I-SIGNAL-TRIG-GERING` of the signal instance: ▶ `NOTIFY` for `KEEP`. ▶ `REPLACE` for `REPLACE`. Other values of `HANDLE-INVALID` are ignored.

Configuration parameters	Mapping description
	`ComUpdateBitPosition` is set to `UPDATE-INDICATION-BIT-POSITION` of the signal instance's `I-SIGNAL-TO-I-PDU-MAPPING`.

`ComTransferProperty` is set depending on the value of `I-SIGNAL-TO-I-PDU-MAPPING/TRANSFER-PROPERTY`:

▶   `PENDING` for `PENDING`.

▶   `TRIGGERED` for `TRIGGERED`.

▶   `TRIGGERED_ON_CHANGE` for `TRIGGERED-ON-CHANGE`.

▶   `TRIGGERED_ON_CHANGE_WITHOUT_REPETITION` for `TRIGGERED-ON-CHANGE-WITHOUT-REPETITION`.

▶   `TRIGGERED_WITHOUT_REPETITION` for `TRIGGERED-WITHOUT-REPETITION`.

For `I-SIGNAL` elements of `NM-PDU` elements, `ComTransferProperty` is always set to `PENDING`.

`ComSystemTemplateSystemSignalRef` is set to the AUTOSAR path of the `I-SIGNAL-TO-I-PDU-MAPPING` of the signal instance.

`ComSignalInitValue` is set to `VALUE` of the `INIT-VALUE` of the `I-SIGNAL`.

Depending on the type of `VALUE-SPECIFICATION` the `INIT-VALUE` must be converted first:

▶   `NUMERICAL-VALUE-SPECIFICATION`

If the `ComSignalType` is `UINT8_N` or `UINT8_DYN`, the value provided in `NUMERICAL-VALUE-SPECIFICATION` is converted to a string which represents a byte array separated by spaces. If the value does not represent an integral value, a warning is issued and `ComSignalInitValue` is not set. Otherwise the value provided in `NUMERICAL-VALUE-SPECIFICATION` is configured without conversion.

▶   `ARRAY-VALUE-SPECIFICATION`

If the `ComSignalType` is `UINT8_N` or `UINT8_DYN`, the `ARRAY-VALUE-SPECIFICATION` is converted to a string which represents a byte array separated by spaces.

Otherwise, the `ARRAY-VALUE-SPECIFICATION` is converted to an integral number where the number's most significant byte contains the first element of the `ARRAY-VALUE-SPECIFICATION`.

Configuration parameters	Mapping description
	▶ `TEXT-VALUE-SPECIFICATION` If the `ComSignalType` is `UINT8_N` or `UINT8_DYN`, the `TEXT-VALUE-SPECIFICATION` is converted to a string which represents a byte array separated by spaces. Characters > 255 are not supported and generate a warning. Otherwise, the characters of the string are converted to an integral number where the number's most significant byte contains the first byte of the `TEXT-VALUE-SPECIFICATION`. Characters > 127 generate a warning. If the string has a length of more than eight characters, an error is issued. ▶ `CONSTANT-REFERENCE` A `CONSTANT-REFERENCE` refers to a `CONSTANT-SPECIFICATION` which in turn contains a `VALUE-SPECIFICATION`. This `VALUE-SPECIFICATION` is taken as input for configuring `ComSignalInitValue`. Other `VALUE-SPECIFICATION` elements are not supported and result in a warning and `ComSignalInitValue` not being set. `ComSignalDataInvalidValue` is set to `VALUE` of the `INVALID-VALUE` of the `I-SIGNAL`'s `NETWORK-REPRESENTATION-PROPS`. Depending on the type of `VALUE-SPECIFICATION` the `INVALID-VALUE` is converted first, see `ComSignalInitValue` for conversion rules. `ComBitPosition` is set to `START-POSITION` of the signal instance's `I-SIGNAL-TO-I-PDU-MAPPING`. In case the `ComSignalType` determined for the `I-SIGNAL` is `UINT8_N` or `UINT8_DYN`, and the `PACKING-BYTE-ORDER` of the `I-SIGNAL-TO-I-PDU-MAPPING` is `MOST-SIGNIFICANT-BYTE-FIRST`, the `START-POSITION` needs to be converted to `MOST-SIGNIFICANT-BYTE-LAST` ("little endian"). If the converted position is not byte-aligned within the parent PDU, a warning is reported and the original value of `START-POSITION` is used. For `I-SIGNAL` elements of `NM-PDU` elements the resulting value of `START-POSITION` needs to be adjusted by the length of the non-user data (see `Pdu-Length` in <u>Section 3.4.14, "EcuC"</u>): `ComBitPosition = (converted) START-POSITION - <length of non-user data [bytes]> * 8`. If the value of `ComBitPosition` is less than zero, a warning is reported.

Configuration parameters	Mapping description
	`ComSignalEndianness` is set depending on the value of `PACKING-BYTE-ORDER` of the signal instance's `I-SIGNAL-TO-I-PDU-MAPPING`: ▶ `BIG_ENDIAN` for `MOST-SIGNIFICANT-BYTE-FIRST`. ▶ `LITTLE_ENDIAN` for `MOST-SIGNIFICANT-BYTE-LAST`. In case the `ComSignalType` determined for the `I-SIGNAL` is `UINT8_N` or `UINT8_DYN`, `ComSignalEndianess` is set to `OPAQUE`. The configuration of `ComBitSize` and `ComSignalLength` depends on what has been configured for `ComSignalType`: ▶ If `ComSignalType` was set to `UINT8_N`, the value is taken from `BASE-TYPE-SIZE` of `SW-BASE-TYPE`. If `BASE-TYPE-SIZE` is not available, the `LENGTH` value of `I-SIGNAL` is taken. If `LENGTH` is not available the `SDG` named "cme:ISignalExtension/cme:length" is used instead. ▶ In any other cases, the `LENGTH` value of `I-SIGNAL` is taken, or, if that value is not available, the `SDG` named "cme:ISignalExtension/cme:length" is used. `ComBitSize` is set to the value calculated as described above. For `UINT8_N` or `UINT8_DYN` the parameter is only configured if it is a multiple of eight. `ComSignalDirection` is set to `SEND` if the PDU instance is sent by the imported `ECU-INSTANCE`. Otherwise it is set to `RECEIVE`. `ComSignalLength` is configured for `UINT8_N` or `UINT8_DYN` signals. If the value calculated as described above is divisible by eight without remainder, `ComSignalLength` is configured with the result of the division.
`ComConfig/ComSignal/ComFilter`	If a `TRANSMISSION-MODE-CONDITION` (`I-SIGNAL-I-PDU` -> `I-PDU-TIMING` -> `TRANSMISSION-MODE-DECLARATION` -> `TRANSMISSION-MODE-CONDITION`) referring to the same `I-SIGNAL-TO-I-PDU-MAPPING` as the signal instance exists, and the signal instance is sent by the imported `ECU-INSTANCE`, a `ComFilter` container is created. If the `DATA-FILTER-TYPE` of the `TRANSMISSION-MODE-CONDITION`'s `DATA-FILTER` is set to `ALWAYS` and a `TRANSMISSION-MODE-TRUE-TIMING` exists for this `TRANSMISSION-MODE-CONDITION`, no `ComFilter` container is created. If in this case a `TRANSMISSION-MODE-FALSE-TIMING` exists for the `TRANSMISSION-MODE-CONDITION` as well, a warning is issued.

Configuration parameters	Mapping description
	If the `I-SIGNAL-TO-I-PDU-MAPPING` has already been referenced by another `TRANSMISSION-MODE-CONDITION`, a warning is issued and the `ComFilter` parameters are not exported a second time. A `ComFilter` container is also created for received signal instances, if a `DATA-FILTER` exists for the `I-SIGNAL-PORT` via which the `I-SIGNAL` is received by the imported `ECU-INSTANCE`, and if the `I-SIGNAL`'s `DATA-TYPE-POLICY` is set to `LEGACY`. `ComFilterAlgorithm` is set depending on the value of `DATA-FILTER-TYPE`: ► `ALWAYS` for `ALWAYS`. ► `NEVER` for `NEVER`. ► `MASKED_NEW_EQUALS_X` for `MASKED-NEW-EQUALS-X`. ► `MASKED_NEW_DIFFERS_MASKED_OLD` for `NEW-IS-DIFFERENT`. ► `MASKED_NEW_DIFFERS_MASKED_OLD` for `MASKED-NEW-DIFFERS-MASKED-OLD`. ► `MASKED_NEW_DIFFERS_X` for `MASKED-NEW-DIFFERS-X`. ► `NEW_IS_WITHIN` for `NEW-IS-WITHIN`. ► `NEW_IS_OUTSIDE` for `NEW-IS-OUTSIDE`. ► `ONE_EVERY_N` for `ONE-EVERY-N`. For other `DATA-FILTER-TYPE` values, `ComFilterAlgorithm` is not exported. `ComFilterMask` is set to `MASK`, if `DATA-FILTER-TYPE` is one of `MASKED-NEW-DIFFERS-MASKED-OLD`, `MASKED-NEW-DIFFERS-X`, `MASKED-NEW-EQUALS-X` or `NEW-IS-DIFFERENT`. `ComFilterX` is set to `X`, if `DATA-FILTER-TYPE` is `MASKED-NEW-DIFFERS-X` or `MASKED-NEW-EQUALS-X`. `ComFilterMax` is set to `MAX`, if `DATA-FILTER-TYPE` is `NEW-IS-OUTSIDE` or `NEW-IS-WITHIN-X`. `ComFilterMin` is set to `MIN`, if `DATA-FILTER-TYPE` is `NEW-IS-OUTSIDE` or `NEW-IS-WITHIN-X`. `ComFilterOffset` is set to `OFFSET`, if `DATA-FILTER-TYPE` is `ONE-EVERY-N`. `ComFilterPeriod` is set to `PERIOD`, if `DATA-FILTER-TYPE` is `ONE-EVERY-N`.

Configuration parameters	Mapping description
`ComConfig/ComSignal/ComSignalType`	In the first step it is determined whether the `I-SIGNAL` references any `DATA-TRANSFORMATION` elements via `DATA-TRANSFORMATIONS` or whether `DATA-TYPE-POLICY` is set to `TRANSFORMING-I-SIGNAL`. In these cases, `ComSignalType` is set to one of the following: ► `UINT8_DYN` if the `I-SIGNAL` references a `SYSTEM-SIGNAL` that has `DYNAMIC-LENGTH` set to `true`. ► `UINT8_N` if the `I-SIGNAL` references a `SYSTEM-SIGNAL` that has `DYNAMIC-LENGTH` not set or set to `false`. If the `I-SIGNAL` does not reference any `DATA-TRANSFORMATION`, the `SW-DATA-DEF-PROPS` to be used subsequently is retrieved depending on the `DATA-TYPE-POLICY` of the `I-SIGNAL`: ► `NETWORK-REPRESENTATION-FROM-COM-SPEC`: see table "DataTypePolicyEnum" in [1] and specification items [TPS_SYST_02006] and [TPS_SYST_02079] in [3]. ► `LEGACY`, `PORT-INTERFACE-DEFINITION` and `OVERRIDE`: `SW-DATA-DEF-PROPS` is retrieved via the `NETWORK-REPRESENTATION-PROPS` of the `I-SIGNAL`. In the next step the `SW-BASE-TYPE` referenced via `BASE-TYPE-REF` of the `SW-DATA-DEF-PROPS` is retrieved. `ComSignalType` is then calculated according to the table *SwBaseType to ComSignalType Mapping* in [4]. If `I-SIGNAL-TYPE` of the `I-SIGNAL` is not available, `PRIMITIVE` is assumed per default. If `BASE-TYPE-SIZE` is not available, the `I-SIGNAL`'s `LENGTH` is used as `BASE-TYPE-SIZE`. If `BASE-TYPE-ENCODING` is not available, `BOOLEAN` is assumed for 1 bit signals and `NONE` for any other signal bit length. `BASE-TYPE-SIZE` values are rounded up to 8, 16, 32, or 64 if the `BASE-TYPE-ENCODING` yields a value of `2C` or `NONE`. If no `SW-BASE-TYPE` is available directly via `BASE-TYPE-REF`, the `IMPLEMENTATION-DATA-TYPE` is retrieved by recursively searching through `SW-DATA-DEF-PROPS` -> `IMPLEMENTATION-DATA-TYPE` -> `SW-DATA-DEF-PROPS`..., until an `IMPLEMENTATION-DATA-TYPE` with a `CATEGORY` other than `TYPE_REFERENCE` is found. If an `IMPLEMENTATION-DATA-TYPE`'s `CATEGORY` is `TYPE_REFERENCE`, its `SHORT-NAME` is `boolean`, and it references an `IMPLEMENTATION-DATA-TYPE` with `SHORT-NAME` set to `uint8`, `ComSignalType` is set to `BOOLEAN`.

Configuration parameters	Mapping description
	If no `SW-DATA-DEF-PROPS` or `IMPLEMENTATION-DATA-TYPE` was found in the previous steps, `ComSignalType` is calculated from the `I-SIGNAL`'s `LENGTH`: ► > 64, or `LENGTH` not available -> `UINT8_N` ► [33 .. 64] -> `UINT64` ► [17 .. 32] -> `UINT32` ► [9 .. 16] -> `UINT16` ► [2 .. 8] -> `UINT8` ► 1 -> `BOOLEAN` If the `CATEGORY` of the found `IMPLEMENTATION-DATA-TYPE` is `ARRAY` or `STRUCTURE`, `ComSignalType` is set to `UINT8_N`. If no `LOWER-LIMIT/@INTERVAL-TYPE` or `UPPER-LIMIT/@INTERVAL-TYPE` is defined by the `DATA-CONSTR` referenced via `DATA-CONSTR-REF`, `ComSignalType` is calculated using the `SW-BASE-TYPE` referenced by the `IMPLEMENTATION-DATA-TYPE`, or the `I-SIGNAL`'s `LENGTH` if the `SW-BASE-TYPE` is not available. If either `LOWER-LIMIT/@INTERVAL-TYPE` or `UPPER-LIMIT/@INTERVAL-TYPE` is `INFINITE`, `ComSignalType` is set to `FLOAT32` or `FLOAT64` (`I-SIGNAL`'s `LENGTH` > 32). If the `BASE-TYPE-ENCODING` as defined by the `SW-BASE-TYPE` referenced by the `IMPLEMENTATION-DATA-TYPE` via `BASE-TYPE-REF` equals `IEEE754`, `ComSignalType` is set to `FLOAT32` or `FLOAT64` (`I-SIGNAL`'s `LENGTH` > 32). `LOWER-LIMIT/UPPER-LIMIT` elements having `INTERVAL-TYPE` elements which are `OPEN` are converted to `CLOSED` first, i.e.]-1 .. 8[is converted to [0 .. 7]. Then `ComSignalType` is calculated from the bit length required to cover the value range `LOWER-LIMIT/UPPER-LIMIT` according to the lists below. Note that the `BASE-TYPE-SIZE` of a referenced `SW-BASE-TYPE` overrides this calculated bit length. If `BASE-TYPE-ENCODING` is available as well, `ComSignalType` is calculated according to the table "SwBaseType to ComSignalType Mapping" in [1]. Bit length of `UPPER-LIMIT` (value of converted `LOWER-LIMIT` >= 0): ► > 64 -> `UINT8_N` ► [33 .. 64] -> `UINT64`

Configuration parameters	Mapping description
	▶ [17 .. 32] -> `UINT32` ▶ [9 .. 16] -> `UINT16` ▶ [2 .. 8] -> `UINT8` ▶ 1 -> `BOOLEAN` Bit length of `LOWER-LIMIT`/`UPPER-LIMIT` (whichever is greater, value of converted `LOWER-LIMIT` < 0): ▶ > 64 -> `UINT8_N` ▶ [33 .. 64] -> `SINT64` ▶ [17 .. 32] -> `SINT32` ▶ [9 .. 16] -> `SINT16` ▶ [2 .. 8] -> `SINT8` ▶ 1 -> `BOOLEAN`
`ComConfig/ComSignalGroup`	For every signal instance which represents an `I-SIGNAL-GROUP` and which is sent or received by the imported `ECU-INSTANCE`, a `ComSignalGroup` container is created. The container name is `GR<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `I-SIGNAL-GROUP`. For the setting of `ComTimeout`, `ComFirstTimeout` and `ComUpdateBitPosition`, see `ComConfig/ComSignal`. If the imported `ECU-INSTANCE` receives the `I-SIGNAL-GROUP` and the `I-SIGNAL/DATA-TYPE-POLICY` properties of all `I-SIGNAL` elements of the `I-SIGNAL-GROUP` are either not set or set to one of `LEGACY` or `OVERRIDE`, `ComDataInvalidAction` is configured according to the value of `HANDLE-INVALID` of the `I-SIGNAL-PORT` that connects one of the `COMMUNICATION-CONNECTOR` elements of the `ECU-INSTANCE` to the `I-SIGNAL-TRIGGERING` of the signal instance. The mapping of the parameter values is done in the same way as described for `ComConfig/ComSignal`. If the `I-SIGNAL-GROUP` is contained in an `NM-PDU`, `ComTransferProperty` is set to `PENDING`. If the `I-SIGNAL-GROUP` is contained in an `I-SIGNAL-I-PDU`, the algorithm collects all distinct and valid `I-SIGNAL-TO-I-PDU-MAPPING/TRANSFER-PROPERTY` values of all `I-SIGNAL-TO-I-PDU-MAPPING` elements that are associated with any `I-SIGNAL` element of the `I-SIGNAL-GROUP`. If the collection is empty, the `I-SIGNAL-TO-I-PDU-MAPPING/TRANSFER-PROPERTY` value

Configuration parameters	Mapping description
	of the `I-SIGNAL-GROUP` is used to configure the `ComTransferProperty` according to `ComConfig/ComSignal`. If the collection contains exactly one value, this `TRANSFER-PROPERTY` is used to configure the `ComTransferProperty`. If the collection contains two values and one of them is `PENDING`, the other value is used to configure the `ComTransferProperty`. In any other case an error is issued. `ComSystemTemplateSignalGroupRef` is set to the AUTOSAR path of the `I-SIGNAL-TO-I-PDU-MAPPING` of the signal instance. If the `I-SIGNAL-GROUP` references a `DATA-TRANSFORMATION` via `COM-BASED-SIGNAL-GROUP-TRANSFORMATIONS/DATA-TRANSFORMATION-REF-CONDITIONAL/DATA-TRANSFORMATION-REF`, `ComSignalGroupArrayAccess` is set to `true`, otherwise `ComSignalGroupArrayAccess` is set to `false`. The `ComSignalGroupDirection` is set to `SEND` if the PDU instance is sent by the imported `ECU-INSTANCE`. Otherwise it is set to `RECEIVE`.
`ComConfig/ComSignalGroup/ComGroupSignal`	For every `SYSTEM-SIGNAL` which is part of the `I-SIGNAL-GROUP` ("GroupSignal"), a `ComGroupSignal` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `SYSTEM-SIGNAL`. `PACKING-BYTE-ORDER` of the GroupSignal's `I-SIGNAL-TO-I-PDU-MAPPING` must be the same for all GroupSignals of the `I-SIGNAL-GROUP`. Different `PACKING-BYTE-ORDER` values result in an error. For the setting of `ComSignalInitValue`, `ComSignalDataInvalidValue`, `ComBitPosition`, `ComSignalLength`, `ComSignalEndianess`, `ComBitSize` and `ComSignalType` see `ComConfig/ComSignal`. `ComSystemTemplateSystemSignalRef` is set to the AUTOSAR path of the `I-SIGNAL-TO-I-PDU-MAPPING` of the signal instance of the GroupSignal. `ComFilter` is set only for GroupSignals which are sent by the imported `ECU-INSTANCE`, see `ComSignal/ComFilter`. `ComTransferProperty` is set depending on the value of the GroupSignal's `I-SIGNAL-TO-I-PDU-MAPPING/TRANSFER-PROPERTY`: ▶ `PENDING` for `PENDING`.

Configuration parameters	Mapping description
	► `TRIGGERED_ON_CHANGE` for `TRIGGERED-ON-CHANGE`. For all other values a warning is issued and `ComTransferProperty` is not set.
`ComConfig/ComIP-duGroup`	For every `I-SIGNAL-I-PDU-GROUP` associated with the imported `ECU-INSTANCE`, a `ComIPduGroup` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `I-SIGNAL-I-PDU-GROUP`. For `I-SIGNAL-I-PDU-GROUP` elements contained within other `I-SIGNAL-I-PDU-GROUP` elements references to the containers of the parent groups are set in the reference list `ComIPduGroupGroupRef`. If no `I-SIGNAL-I-PDU-GROUP` element exists that references an `I-SIG-NAL-I-PDU` which is sent or received by the imported `ECU-INSTANCE`, two default `ComIPduGroup` elements are created, named `RXPDUS_GLOBAL` and `TXPDUS_GLOBAL`. Then for each `CAN-CLUSTER`, `FLEXRAY-CLUSTER`, `LIN-CLUSTER` and `ETHERNET-PHYSICAL-CHANNEL` to which the imported `ECU-INSTANCE` is connected, two `ComIPduGroup` elements are created named `RG<SHORT-NAME of CLUSTER or PHYSICAL-CHANNEL>` and `TG<SHORT-NAME of CLUSTER or PHYSICAL-CHANNEL>`. Via `ComIPduGroup-GroupRef` a reference to the parent `ComIPduGroup` is set, `RXPDUS_GLOBAL` for all `RG` groups, `TXPDUS_GLOBAL` for all `TG` groups. If no `I-SIGNAL-I-PDU-GROUP` element exists that references an `NM-PDU` which is sent or received by the imported `ECU-INSTANCE`, two default `ComIP-duGroup` elements are created, named `RXNMPDUS_GLOBAL` and `TXNMP-DUS_GLOBAL`. Then for each `CAN-CLUSTER`, `FLEXRAY-CLUSTER` and `ETH-ERNET-PHYSICAL-CHANNEL` to which the imported `ECU-INSTANCE` is connected, two `ComIPduGroup` elements are created named `RNG<SHORT-NAME of COMMUNICATION-CLUSTER>` and `TNG<SHORT-NAME of COMMUNI-CATION-CLUSTER>`. Via `ComIPduGroupGroupRef` a reference to the parent `ComIPduGroup` is set, `RXNMPDUS_GLOBAL` for all `RG` groups, `TXNMP-DUS_GLOBAL` for all `TG` groups. If no `NM-PDU` elements exist for one direction (Rx/Tx), no dedicated `ComIPduGroup` is created for this direction.
`ComConfig/ComTime-Base`	The `ComTimeBase` container is created if at least one of the parameters `COM-CONFIGURATION-GW-TIME-BASE`, `COM-CONFIGURATION-RX-TIME-BASE`, or `COM-CONFIGURATION-TX-TIME-BASE` is available. `ComConfigurationGwTimeBase` is set to `COM-CONFIGURATION-GW-TIME-BASE`. `ComConfigurationRxTimeBase` is set to `COM-CONFIGURATION-RX-TIME-BASE`.

Configuration parameters	Mapping description
	`ComConfigurationTxTimeBase` is set to `COM-CONFIGURATION-TX-TIME-BASE`.
`ComConfig/ComGwMapping`	For each received signal instance representing an `I-SIGNAL` or an `I-SIGNAL-GROUP`, which is referenced via a `SOURCE-SIGNAL-REF -> I-SIGNAL-TRIGGERING` that belongs to an `I-SIGNAL-MAPPING` in the `GATEWAY` of the imported `ECU-INSTANCE`, a `ComGwMapping` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `I-SIGNAL` or the `I-SIGNAL-GROUP`.
`ComConfig/ComGwMapping/ComGwSource`	The choice container is always set to `ComGwSignal`. `ComGwSignal/ComGwSignalRef` references the corresponding `ComSignal` or `ComSignalGroup` container created for the signal instance used to create the `ComGwMapping` container.
`ComConfig/ComGwMapping/ComGwDestination`	For each sent signal instance representing an `I-SIGNAL` or an `I-SIGNAL-GROUP`, which is referenced via a `TARGET-SIGNAL-REF -> I-SIGNAL-TRIGGERING` that belongs to an `I-SIGNAL-MAPPING` and, which also references the `I-SIGNAL` or `I-SIGNAL-GROUP` of the `ComGwSource`, a `ComGwDestination` container is created . The container name is `GM<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `I-SIGNAL` or the `I-SIGNAL-GROUP`. `ComGwSignal/ComGwSignalRef` references the corresponding `ComSignal` or `ComSignalGroup` container.

3.4.8.1. Configuration for partial networking ERA and EIRA PDUs and signals

Configuration parameters	Mapping description
`ComConfig/ComIPdu`	For every ERA and EIRA `Pdu` container created for PNC-enabled `NM-CLUSTER` elements in the `EcuC` module (see Section 3.4.14, "EcuC"), a `ComIPdu` is created. The container name equals the name of the `Pdu` container in the `EcuC` module. The single `ComIPduSignalRef` entry references the `ComSignal` container also created for this PDU. `ComIPduDirection` is set to `RECEIVE`. The single `ComIPduGroupRef` entry

Configuration parameters	Mapping description
	references the `ComIPduGroup` also created for this PDU. `ComPduIdRef` references the `Pdu` container in the `EcuC` module.
`ComConfig/ComSignal`	For every created `ComIPdu` container, a corresponding `ComSignal` container is created. The container name is `SG<ComIPdu name>`. `ComBitSize` is set to `PNC-VECTOR-LENGTH * 8`. `ComSignalLength` is set to `PNC-VECTOR-LENGTH`. `ComSignalInitValue` is set to `0`, converted to a space separated byte array where `PNC-VECTOR-LENGTH` defines the length of this byte array. If `PNC-VECTOR-LENGTH` is not defined, a warning is issued for `ComBitSize`, `ComSignalLength`, and `ComSignalInitValue` and these parameters are not set. `ComBitPosition` is set to `0`. `ComSignalDirection` is set to `SEND` if the PDU instance is sent by the imported `ECU-INSTANCE`. Otherwise it is set to `RECEIVE`. `ComSignalEndianness` is set to `OPAQUE`. `ComSignalType` is set to `UINT8_N`. `ComNotification` is set to `ComM_COMCbk_<ComSignal name>`.
`ComConfig/ComIPduGroup`	For every created EIRA `ComIPdu` container, a corresponding `ComIPduGroup` container is created. The container name depends on the `EcuC Pdu` container for which the `ComIPdu` has been created: ▸ `RNGEIRACanNm` for the `CanNmPnEiraRxNSdu` container. It references the global `ComIPduGroup` named `RXEIRAPDUS_GLOBAL`. ▸ `RNGEIRAFrNm` for the `FlexRayNmPnEiraRxNSdu` container. It references the global `ComIPduGroup` named `RXEIRAPDUS_GLOBAL`. ▸ `RNGEIRAUdpNm` for the `UdpNmPnEiraRxNSdu` container. It references the global `ComIPduGroup` named `RXEIRAPDUS_GLOBAL`. For every created ERA `ComIPdu` container, a corresponding `ComIPduGroup` container is created. The container name is `RNG<name>` for received PDUs, and `TNG<name>` for sent PDUs, where `<name>` is the mangled name of the `CAN-COMMUNICATION-CLUSTER`, `FLEXRAY-COMMUNICATION-CLUSTER`, or `ETHERNET-PHYSICAL-CHANNEL` on which the PDU is sent or received. The refer-

Configuration parameters	Mapping description
	enced parent `ComIPduGroup` is `RXNMPDUS_GLOBAL` for `ComIPduGroup` containers of received PDUs and `TXNMPDUS_GLOBAL` for `ComIPduGroup` containers of sent PDUs.

3.4.8.2. Configuration for partial networking IRA signals

An *IRA* (internal request array) signal is represented by a `ComSignal` container that is associated with a `ComIPdu` container. This `ComIPdu` container has been created for an `NmUserDataPdu`, which the `ECU-INSTANCE` is sending in a PNC-enabled `NM-CLUSTER`. The bit area of an IRA signal covers the PNC vector which is defined by the `PNC-VECTOR-OFFSET` and `PNC-VECTOR-LENGTH` of the `SYSTEM`. If the `NM-PDU` that represents the `NmUserDataPdu` contains an `I-SIGNAL`, which completely covers the PNC vector, the signal is picked for IRA signal configuration.

If the `NM-PDU` does not contain such an `I-SIGNAL`, a dedicated `ComSignal` container is created. The container name is `<name>NmPnIraTxNSdu`, where `<name>` is the mangled name of the `CAN-COMMUNICATION-CLUSTER`, `FLEXRAY-COMMUNICATION-CLUSTER`, or `ETHERNET-PHYSICAL-CHANNEL` on which the IRA signal is sent.

The following `ComSignal` parameters for IRA signals are configured as described for EIRA and ERA signals in Section 3.4.8.1, "Configuration for partial networking ERA and EIRA PDUs and signals":

▶ `ComBitSize`

▶ `ComSignalDirection`

▶ `ComSignalLength`

▶ `ComSignalEndianness`

▶ `ComSignalType`

`ComBitOffset` is set to `(PNC-VECTOR-OFFSET - <non user data area>) * 8`, where `<non user data area>` is the byte length of the non user data area of the `NM-PDU` as described in Section 3.4.14, "EcuC".

3.4.9. ComM

Configuration parameters	Mapping description
ComMGeneral	`ComMPncGatewayEnabled` is set to `false` if all `COMMUNICATION-CONNECTOR` elements of the imported `ECU-INSTANCE` have the parameter `PNC-GATEWAY-TYPE` set to `NONE`. Otherwise `ComMPncGatewayEnabled` is set to `true`. If `PNC-GATEWAY-TYPE` is undefined for any of the `COMMUNICATION-CONNEC-`

Configuration parameters	Mapping description
	`TOR` elements, `ComMPncGatewayEnabled` is not set and a warning is reported. `ComMPncPrepareSleepTimer` is set to `PNC-PREPARE-SLEEP-TIMER`.
`ComMConfigSet`	`ComMPncEnabled` is set to `true` if any PNC-enabled `NM-CLUSTER` (see [Section 3.4.14, "EcuC"](#)) exists. Otherwise `ComMPncEnabled` is not set.
`ComMConfigSet/ComMUser`	One `ComMUser` container is created for each `COM-MGR-USER-NEEDS` element for which the following conditions hold ▶ The aggregating `SWC-SERVICE-DEPENDENCY` refers via `REPRESENTED-PORT-GROUP-REF` to a valid `PORT-GROUP` ▶ One or more `COM-MANAGEMENT-MAPPING` elements refer to that `PORT-GROUP` as well ▶ The `COM-MANAGEMENT-MAPPING` elements refer to one or more `CAN-CLUSTER`, `J-1939-CLUSTER`, `FLEXRAY-CLUSTER`, `LIN-CLUSTER`, or `ETHERNET-PHYSICAL-CHANNEL` elements to which the imported `ECU-INSTANCE` is connected. The `COM-MANAGEMENT-MAPPING` refers to these elements either directly via `COM-MANAGEMENT-PORT-GROUP-IREFS/COM-MANAGEMENT-PORT-GROUP-IREF/TARGET-REF`, or indirectly via `COM-MANAGEMENT-GROUP-REFS/COM-MANAGEMENT-GROUP-REF`. In the latter case, all referenced `I-SIGNAL-I-PDU-GROUP` elements and their sub groups are retrieved. In the second step, all PDUs belonging to the `I-SIGNAL-I-PDU-GROUP` elements are collected. The set of `CAN-CLUSTER`, `J-1939-CLUSTER`, `FLEXRAY-CLUSTER`, `LIN-CLUSTER`, or `ETHERNET-PHYSICAL-CHANNEL` elements via which the imported `ECU-INSTANCE` sends or receives any of the PDUs are considered the elements indirectly referenced via `COM-MANAGEMENT-GROUP-REFS/COM-MANAGEMENT-GROUP-REF`. The `ComMUser` container is related to all `ComMChannel` containers that are set up for the `CAN-CLUSTER`, `J-1939-CLUSTER`, `FLEXRAY-CLUSTER`, `LIN-CLUSTER`, or `ETHERNET-PHYSICAL-CHANNEL` elements of the `COM-MANAGEMENT-MAPPING` elements. The name of the `ComMUser` container is the mangled name of the `COM-MGR-USER-NEEDS` element.
`ComMConfigSet/ComMChannel`	For each `CAN-CLUSTER`, `J-1939-CLUSTER`, `FLEXRAY-CLUSTER`, `LIN-CLUSTER` and `ETHERNET-PHYSICAL-CHANNEL` to which the imported `ECU-INSTANCE` is connected, a `ComMChannel` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `COMMUNICATION-CLUSTER` or `ETHERNET-PHYSICAL-CHANNEL`. A `ComMChannel` is considered to be a *managing* channel in the following cases

Configuration parameters	Mapping description
	▶ The `ComMChannel` is represented by an `ETHERNET-PHYSICAL-CHANNEL` and that `ETHERNET-PHYSICAL-CHANNEL` refers via `MANAGED-PHYSICAL-CHANNEL-REF` to one or more `ETHERNET-PHYSICAL-CHANNEL` elements or to `PHYSICAL-CHANNEL` elements that are contained in `CAN-CLUSTER`, `J-1939-CLUSTER`, `FLEXRAY-CLUSTER`, or `LIN-CLUSTER` elements which in turn represent the *managed* `ComMChannel` elements. ▶ The `ComMChannel` is represented by a `CAN-CLUSTER`, `J-1939-CLUSTER`, `FLEXRAY-CLUSTER`, or `LIN-CLUSTER` and one or more of the `PHYSICAL-CHANNEL` elements of the cluster refer via `MANAGED-PHYSICAL-CHANNEL-REF` to one or more `ETHERNET-PHYSICAL-CHANNEL` elements or to `PHYSICAL-CHANNEL` elements that are contained in `CAN-CLUSTER`, `J-1939-CLUSTER`, `FLEXRAY-CLUSTER`, or `LIN-CLUSTER` elements which in turn represent the *managed* `ComMChannel` elements. `ComMBusType` is set depending on the type of the `COMMUNICATION-CLUSTER` or `ETHERNET-PHYSICAL-CHANNEL`: ▶ `COMM_BUS_TYPE_CAN` for `CAN-CLUSTER`. ▶ `COMM_BUS_TYPE_CAN` for `J-1939-CLUSTER`. ▶ `COMM_BUS_TYPE_FR` for `FLEXRAY-CLUSTER`. ▶ `COMM_BUS_TYPE_ETH` for `ETHERNET-PHYSICAL-CHANNEL`. ▶ `COMM_BUS_TYPE_LIN` for `LIN-CLUSTER`. ▶ For any other cluster type `ComMBusType` is not set and a warning is issued. `ComMPncGatewayType` is not configured if the `ComMChannel` is a *managed* `ComMChannel`. Otherwise `ComMPncGatewayType` is set depending on the `PNC-GATEWAY-TYPE` of the `COMMUNICATION-CONNECTOR` that connects the imported `ECU-INSTANCE` to the `COMMUNICATION-CLUSTER` or `ETHERNET-PHYSICAL-CHANNEL`: ▶ `COMM_GATEWAY_TYPE_ACTIVE` for `ACTIVE` ▶ `COMM_GATEWAY_TYPE_PASSIVE` for `PASSIVE` `ComMPncGatewayType` is not configured if `PNC-GATEWAY-TYPE` is not defined or set to `NONE`. If the `ComMChannel` is a *managed* `ComMChannel`, `ComMManageReference` is configured to refer to the corresponding *managing* `ComMChannel` container.

Configuration parameters	Mapping description
	If any `ComMUser` container has been created which is associated with this `ComMChannel` container, one `ComMUserPerChannel` container is created. The name of the container is the same as the name of the `ComMUser` container. The reference `ComMUserChannel` is configured to refer to the `ComMUser` container. `ComMNetworkManagement/ComMNmVariant` is set to `LIGHT` for *managed* `ComMChannel` containers, and to `FULL` for *managing* `ComMChannel` containers. If the `ComMChannel` is neither *managed* nor *managing* the following rules apply. ▶ For `LIN-CLUSTER` elements, `ComMNmVariant` is set to `LIGHT` ▶ If no `NM-NODE` exists that connects an `NM-CLUSTER` to the `COMMUNICATION-CLUSTER`, or if the `NM-NODE` does not reference any sent or received `NM-PDU` elements, `ComMNmVariant` is set to `NONE` unless `ComMNmVariant` has been manually configured as `LIGHT`, in which case that value is left unchanged. ▶ If an `NM-NODE` exists that connects a `NM-CLUSTER` to the `COMMUNICATION-CLUSTER`, and if the `NM-NODE` references at least one sent or received `NM-PDU` element, `ComMNmVariant` is set to `PASSIVE` if the `NM-NODE`'s `NM-PASSIVE-MODE-ENABLED` is set to `true`. If `NM-PASSIVE-MODE-ENABLED` is not set or set to `false`, `ComMNmVariant` is set to `FULL`.
`ComMConfigSet/ComMPnc`	For every `PNC-MAPPING` for which a valid and unique `Id` can be calculated and which fulfills at least one of the following conditions: ▶ The `PNC-MAPPING` is referencing an `I-SIGNAL-I-PDU-GROUP` which is also referenced by the imported `ECU-INSTANCE` via `ASSOCIATED-COM-I-PDU-GROUP-REFS` ▶ The `PNC-MAPPING` is referencing at least one `PHYSICAL-CHANNEL` to which the configured ECU is connected as well , a `ComMPnc` container is created. Each `ComMPnc` is identified by its `Id`. The formula for calculating this `Id` depends on the AUTOSAR specification version that the `ComM` module implements. For `ComM` modules that implement AUTOSAR 4.1.0 or lower, the formula `Id := PNC-IDENTIFIER - (PNC-VECTOR-OFFSET * 8)` is applied, where `PNC-VECTOR-OFFSET` is defined at the `SYSTEM` to which the `PNC-IDENTIFIER` belongs. For `ComM` modules that implement AUTOSAR 4.1.1 or higher, `Id := PNC-IDENTIFIER` is applied.

Configuration parameters	Mapping description
	The container name of the `ComMPnc` is `ComMPnc_<Id>`. If `Id` cannot be calculated due to a missing `PNC-IDENTIFIER` or `PNC-VECTOR-OFFSET`, a warning is issued and no `ComMPnc` container is created. If multiple `PNC-MAPPING` elements exist that have the same `PNC-IDENTIFIER`, a warning is issued and only the first `PNC-MAPPING` is used. `ComMPncId` is set to `Id`. For every `COMMUNICATION-CLUSTER` or `ETHERNET-PHYSICAL-CHANNEL` through which either any of the `I-SIGNAL-I-PDU-GROUP`'s PDU elements are sent, and/or received, and/or routed, or which is referenced via `PNC-MAPPING/PHYSICAL-CHANNEL-REF`, one `ComMChannelPerPnc` reference is created. This `ComMChannelPerPnc` references the `ComMChannel` that has been created for the `COMMUNICATION-CLUSTER` or `ETHERNET-PHYSICAL-CHANNEL`. If an `EthIfSwitchPortGroup` container has been created in the `EthIf` module for the `PNC-MAPPING`, `ComMPncEthIfSwitchPortGroupRef` is configured to refer to this container. Section 3.4.17, "EthIf" describes the configuration of `EthIfSwitchPortGroup` containers in detail.
`ComMConfigSet/ComMPnc/ComMPncComSignal`	If a PNC-enabled `CAN-NM-CLUSTER` that sends/receives a PDU of the `PNC-MAPPING` exists, one `ComMPncComSignal` container named `CanNmPnEiraRxNSdu` is created. If a PNC-enabled `UDP-NM-CLUSTER` that sends/receives a PDU of the `PNC-MAPPING` exists, one `ComMPncComSignal` container named `UdpNmPnEiraRxNSdu` is created. If a PNC-enabled `FLEXRAY-NM-CLUSTER` that sends/receives a PDU of the `PNC-MAPPING` exists, one `ComMPncComSignal` container named `FlexRayNmPnEiraRxNSdu` is created. For every `ComSignal` container that has been set up in `Com` for an EIRA, ERA, or IRA signal that belongs to this `ComMPnc` container, one `ComMPncComSignal` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the signal. The creation of `ComSignal` containers for EIRA, ERA, and IRA signals is described in Section 3.4.8.1, "Configuration for partial networking ERA and EIRA PDUs and signals" and in Section 3.4.8.2, "Configuration for partial networking IRA signals". For detailed information about the PNC Vector, see Section 3.4.4, "CanNm".

Configuration parameters	Mapping description
	The *IRA signal kind* of an IRA signals depends on the `PNC-GATEWAY-TYPE` of the `COMMUNICATION-CONNECTOR` that connects the imported `ECU-INSTANCE` to the `PHYSICAL-CHANNEL` on which the signal is sent. If the `PNC-GATEWAY-TYPE` is set to `PASSIVE`, the *IRA signal kind* of the IRA signal is `ERA`. For all other values, its *IRA signal kind* is `EIRA`. The following parameters of `ComMPncComSignal` are set: ▶ `ComMPncComSignalDirection` is set to `RX` for ERA and EIRA signals, and to `TX` for IRA signals. ▶ `ComMPncComSignalKind` is set to `EIRA` for EIRA and to `ERA` for ERA signals. For IRA signals, the *IRA signal kind* is configured. ▶ `ComMPncComSignalRef` references the corresponding `ComSignal` container in the `Com` module. ▶ For ERA signals and for IRA signals having an *IRA signal kind* of ERA, `ComMPncComSignalChannelRef` references the `ComMChannel` container that has been created for the PNC-enabled `NM-CLUSTER`.

3.4.10. Csm

Several client BSW modules are built onto of the cryptographic services which the `Csm` BSW module is providing. Since each of these client BSW modules requires a differently configured `Csm` infrastructure, this chapter describes the configuration process for each client in a dedicated sub chapter.

3.4.10.1. Csm for SecOC

Configuration parameters	Mapping description
`Csm/CsmJobs/CsmJob`	For every `SECURED-I-PDU` the configured `ECU-INSTANCE` is sending or receiving, the associated `PDU-TRIGGERING` is retrieved. If `PDU-TRIGGERING/SEC-OC-CRYPTO-MAPPING-REF` refers to a valid `SEC-OC-CRYPTO-SERVICE-MAPPING` element, one `CsmJob` container is created. The container name is `<Primitive>_<PduInstName>`, where `<PduInstName>` is the name of the PDU instance, and `<Primitive>` is `CsmMacVerify` for received PDU instances and `CsmMacGenerate` for sent PDU instances.

Configuration parameters	Mapping description
	`CsmJobKeyRef` is set to reference the container that has been created for the `CRYPTO-SERVICE-KEY` referenced via `SEC-OC-CRYPTO-SERVICE-MAP-PING/CRYPTO-SERVICE-KEY-REF`. `CsmJobPrimitiveRef` is set to reference the container that has been created for the `CRYPTO-SERVICE-PRIMITIVE` referenced via `SEC-OC-CRYP-TO-SERVICE-MAPPING/AUTHENTICATION-REF`. `CsmJobQueueRef` is set to reference the container that has been created for the `CRYPTO-SERVICE-QUEUE` referenced via `SEC-OC-CRYPTO-SERVICE-MAPPING/CRYPTO-SERVICE-QUEUE-REF`.
`Csm/CsmQueues/CsmQueue`	For every `CRYPTO-SERVICE-QUEUE` which is referenced by a `SEC-OC-CRYP-TO-SERVICE-MAPPING` of a `SECURED-I-PDU` the configured `ECU-INSTANCE` is sending or receiving, one `CsmQueue` container is created. The container name is the `SHORT-NAME` of the `CRYPTO-SERVICE-QUEUE`. `CsmQueueSize` is set to `CRYPTO-SERVICE-QUEUE/QUEUE-SIZE`.
`Csm/CsmKeys/CsmKey`	For every `CRYPTO-SERVICE-KEY` which is referenced by a `SEC-OC-CRYP-TO-SERVICE-MAPPING` of a `SECURED-I-PDU` the configured `ECU-INSTANCE` is sending or receiving, one `CsmKey` container is created. The container name is the `SHORT-NAME` of the `CRYPTO-SERVICE-KEY`.
`Csm/CsmPrimitives`	For every `CRYPTO-SERVICE-PRIMITIVE` which is referenced by a `SEC-OC-CRYPTO-SERVICE-MAPPING` of a `SECURED-I-PDU` the configured `ECU-INS-TANCE` is sending or receiving, one `CsmPrimitives` container is created. The container name is `<Primitive>_<PduInstName>`, where `<PduInstName>` is the name of the PDU instance, and `<Primitive>` is `CsmMacVerify` for received PDUs and `CsmMacGenerate` for sent PDUs.
`Csm/CsmPrimi-tives/CsmMacVeri-fy/CsmMacVerifyCon-fig`	If the `SECURED-I-PDU` is received, one `Csm/CsmPrimitives/CsmMacVer-ify/CsmMacVerifyConfig` container is created, the container name is set to `CsmMacVerify_<PduInstName>`, where `<PduInstName>` is the name of the PDU instance. `CsmMacVerifyAlgorithmFamiliy` is set to `CRYPTO-SERVICE-PRIMITIVE/ALGORITHM-FAMILY` unless `CRYPTO-SERVICE-PRIMITIVE/ALGORITHM-FAMILY` is either `CRYPTO_ALGOFAM_SHAKE128` or `CRYPTO_AL-GOFAM_SHAKE256`. in the former case, `CRYPTO_ALGOFAM_SHA3_SHAKE128` is configured, in the latter case, `CRYPTO_ALGOFAM_SHA3_SHAKE256` is configured.

Configuration parameters	Mapping description
	`CsmMacVerifyAlgorithmKeyLength` is set to `CRYPTO-SERVICE-KEY/LENGTH` of the `CRYPTO-SERVICE-KEY` referenced via `SEC-OC-CRYPTO-SERVICE-MAPPING/CRYPTO-SERVICE-KEY-REF`. `CsmMacVerifyAlgorithmSecondaryFamily` is set to `CRYPTO_ALGOFAM_CUSTOM` if `CRYPTO-SERVICE-PRIMITIVE/ALGORITHM-SECONDARY-FAMILY` is configured. `CsmMacVerifyAlgorithmSecondaryFamilyCustom` is set to `CRYPTO-SERVICE-PRIMITIVE/ALGORITHM-SECONDARY-FAMILY`.
`Csm/CsmPrimitives/CsmMacGenerate/CsmMacGenerateConfig`	If the `SECURED-I-PDU` is sent, one `Csm/CsmPrimitives/CsmMacGenerate/CsmMacGenerateConfig` container is created, the container name is set to `CsmMacGenerate_<PduInstName>`, where `<PduInstName>` is the name of the PDU instance. `CsmMacGenerateAlgorithmFamiliy` is configured in the same way as `CsmMacVerifyAlgorithmFamiliy` for received `SECURED-I-PDU` elements. `CsmMacGenerateAlgorithmKeyLength` is configured in the same way as `CsmMacVerifyAlgorithmKeyLength` for received `SECURED-I-PDU` elements. `CsmMacGenerateAlgorithmSecondaryFamily` is configured in the same way as `CsmMacVerifyAlgorithmSecondaryFamily` for received `SECURED-I-PDU` elements. `CsmMacGenerateAlgorithmSecondaryFamilyCustom` is configured in the same way as `CsmMacVerifyAlgorithmSecondaryFamilyCustom` for received `SECURED-I-PDU` elements.

3.4.10.2. Csm for Tls

Configuration containers are configured for every `TLS-CRYPTO-SERVICE-MAPPING` for which a `TlsConnection` container was created and a valid cipher suite could be determined as described in Section 3.4.47, "Tls".

For each `TlsConnection` container, `CsmJob` containers are created for

▶ `PRFPreMasterSecret`

▶ `PRFMasterSecret`

▶ `MACGenerate`

► `MACVerify`

► `Hash`

► `Random`

If the cipher suite supports encryption, additional jobs are created for

► `Encrypt`

► `Decrypt`

If the cipher suite uses key exchange, additional job is created for `SignatureGenerate`.

The names of the `CsmJob` is `<Prefix>_<SocketAddress>`, where `<Prefix>` corresponds to the job type described above and `<SocketAddress>` is the name of the `SOCKET-ADDRESS` associated with the `TlsConnection`.

For each of the created `CsmJob` containers, one `CsmKey` and one `CsmPrimitive` container are created. The container names equal the name for the `CsmJob` container. The `CsmJob` container references the `CsmKey` container via `CsmJobKeyRef` and the `CsmPrimitive` container via `CsmJobPrimitiveRef`.

For `PRFPreMasterSecret CsmPrimitive` containers a `CsmMacGenerate` sub container is created, the following parameters are configured:

Module Configuration Parameter	Value
CsmMacGenerateAlgorithmFamiliy	Enumeration value that corresponds to the hash algorithm and hash length of the cipher suite, e.-g. `CRYPTO_ALGOFAM_SHA2_256` for `TLS_PSK_-WITH_AES_128_GCM_SHA256`.
CsmMacGenerateAlgorithmKeyLength	For PSK cipher suites: 2 * `<PSKLen>` + 4, where `<PSKLen>` is the byte length of the pre-shared key.
CsmMacGenerateAlgorithmMode	`CRYPTO_ALGOMODE_HMAC`
CsmMacGenerateResultLength	Byte length of the hash function of the cipher suite
CsmMacGenerateDataMaxLength	`MAXIMUM-TRANSMISSION-UNIT` of the `ETHER-NET-COMMUNICATION-CONNECTOR` associated with the local `SOCKET-ADDRESS` for which the `TlsConnection` was created.

For `PRFMasterSecret CsmPrimitive` containers a `CsmMacGenerate` sub container is created, the following parameters are configured:

Module Configuration Parameter	Value
CsmMacGenerateAlgorithmFamiliy	Enumeration value that corresponds to the hash algorithm and hash length of the cipher suite, e.-

Module Configuration Parameter	Value
	g. `CRYPTO_ALGOFAM_SHA2_256` for `TLS_PSK_-WITH_AES_128_GCM_SHA256`.
CsmMacGenerateAlgorithmKeyLength	`48`
CsmMacGenerateAlgorithmMode	`CRYPTO_ALGOMODE_HMAC`
CsmMacGenerateResultLength	Byte length of the hash function of the cipher suite
CsmMacGenerateDataMaxLength	`MAXIMUM-TRANSMISSION-UNIT` of the `ETHER-NET-COMMUNICATION-CONNECTOR` associated with the local `SOCKET-ADDRESS` for which the `TlsConnection` was created.

For `MACGenerate CsmPrimitive` containers a `CsmMacGenerate` sub container is created, the following parameters are configured:

Module Configuration Parameter	Value
CsmMacGenerateAlgorithmFamiliy	Enumeration value that corresponds to the hash algorithm and hash length of the cipher suite, e.-g. `CRYPTO_ALGOFAM_SHA2_256` for `TLS_PSK_-WITH_AES_128_GCM_SHA256`.
CsmMacGenerateAlgorithmKeyLength	Byte length of the hash function of the cipher suite
CsmMacGenerateAlgorithmMode	`CRYPTO_ALGOMODE_HMAC`
CsmMacGenerateResultLength	Byte length of the hash function of the cipher suite
CsmMacGenerateDataMaxLength	`MAXIMUM-TRANSMISSION-UNIT` of the `ETHER-NET-COMMUNICATION-CONNECTOR` associated with the local `SOCKET-ADDRESS` for which the `TlsConnection` was created.

For `MACVerify CsmPrimitive` containers a `CsmMacVerify` sub container is created, the following parameters are configured:

Module Configuration Parameter	Value
CsmMacVerifyAlgorithmFamiliy	Enumeration value that corresponds to the hash algorithm and hash length of the cipher suite, e.-g. `CRYPTO_ALGOFAM_SHA2_256` for `TLS_PSK_-WITH_AES_128_GCM_SHA256`.
CsmMacVerifyAlgorithmKeyLength	Byte length of the hash function of the cipher suite
CsmMacVerifyAlgorithmMode	`CRYPTO_ALGOMODE_HMAC`
CsmMacVerifyCompareLength	Byte length of the hash function of the cipher suite

Module Configuration Parameter	Value
CsmMacVerifyDataMaxLength	MAXIMUM-TRANSMISSION-UNIT of the ETHER-NET-COMMUNICATION-CONNECTOR associated with the local SOCKET-ADDRESS for which the TlsConnection was created.

For `Hash CsmPrimitive` containers a `CsmHash` sub container is created, the following parameters are configured:

Module Configuration Parameter	Value
CsmHashAlgorithmFamiliy	Enumeration value that corresponds to the hash algorithm and hash length of the cipher suite, e.g. CRYPTO_ALGOFAM_SHA2_256 for TLS_PSK_WITH_AES_128_GCM_SHA256.
CsmHashResultLength	Byte length of the hash function of the cipher suite
CsmHashDataMaxLength	MAXIMUM-TRANSMISSION-UNIT of the ETHER-NET-COMMUNICATION-CONNECTOR associated with the local SOCKET-ADDRESS for which the TlsConnection was created.

For `Random CsmPrimitive` containers a `CsmRandomGenerate` sub container is created, the following parameters are configured:

Module Configuration Parameter	Value
CsmRandomGenerateAlgorithmFamiliy	CRYPTO_ALGOFAM_RNG
CsmRandomGenerateResultLength	Byte length of the hash function of the cipher suite

For `Encrypt CsmPrimitive` containers a `CsmAEADEncrypt` sub container is created, the following parameters are configured:

Module Configuration Parameter	Value
CsmAEADEncryptAlgorithmFamiliy	Enumeration value that corresponds to the hash algorithm and hash length of the cipher suite, e.g. CRYPTO_ALGOFAM_AES for TLS_PSK_WITH_AES_128_GCM_SHA256.
CsmAEADEncryptAlgorithmKeyLength	Byte length of the key of the cipher suite's encryption function
CsmAEADEncryptAlgorithmMode	Enumeration value that corresponds to the mode of the encryption algorithm of the cipher suite, e.g. CRYPTO_ALGOMODE_GCM for TLS_PSK_WITH_AES_128_GCM_SHA256.

Module Configuration Parameter	Value
CsmAEADEncryptTagLength	Byte length of the key of the cipher suite's encryption function
CsmAEADEncryptAssociatedDataMaxLength	`MAXIMUM-TRANSMISSION-UNIT` of the `ETHER-NET-COMMUNICATION-CONNECTOR` associated with the local `SOCKET-ADDRESS` for which the `TlsConnection` was created.
CsmAEADEncryptCiphertextMaxLength	Same as `CsmAEADEncryptAssociatedDataMaxLength`
CsmAEADEncryptPlaintextMaxLength	Same as `CsmAEADEncryptAssociatedDataMaxLength`
CsmAEADEncryptKeyRef	Refers to the `CsmKey` container the `CsmJob` references via `CsmJobKeyRef`.

For `Decrypt CsmPrimitive` containers a `CsmAEADDecrypt` sub container is created, the following parameters are configured:

Module Configuration Parameter	Value
CsmAEADDecryptAlgorithmFamiliy	Enumeration value that corresponds to the hash algorithm and hash length of the cipher suite, e.g. `CRYPTO_ALGOFAM_AES` for `TLS_PSK_WITH_AES_-128_GCM_SHA256`.
CsmAEADDecryptAlgorithmKeyLength	Byte length of the key of the cipher suite's encryption function
CsmAEADDecryptAlgorithmMode	Enumeration value that corresponds to the mode of the encryption algorithm of the cipher suite, e.-g. `CRYPTO_ALGOMODE_GCM` for `TLS_PSK_WITH_-AES_128_GCM_SHA256`.
CsmAEADDecryptTagLength	Byte length of the key of the cipher suite's encryption function
CsmAEADDecryptAssociatedDataMaxLength	`MAXIMUM-TRANSMISSION-UNIT` of the `ETHER-NET-COMMUNICATION-CONNECTOR` associated with the local `SOCKET-ADDRESS` for which the `TlsConnection` was created.
CsmAEADDecryptCiphertextMaxLength	Same as `CsmAEADDecryptAssociatedDataMaxLength`
CsmAEADDecryptPlaintextMaxLength	Same as `CsmAEADDecryptAssociatedDataMaxLength`

Module Configuration Parameter	Value
CsmAEADDecryptKeyRef	Refers to the `CsmKey` container the `CsmJob` references via `CsmJobKeyRef`.

For `SignatureGenerate CsmPrimitive` containers a `CsmSignatureGenerate` sub container is created, the following parameters are configured:

Module Configuration Parameter	Value
CsmSignatureGenerateAlgorithmFamiliy	The parameter is configured depending on the first curve that is associated with the cipher suite. Its value is set to one of the following values: ▶ `CRYPTO_ALGOFAM_ED25519` for the `x25519` curve ▶ `CRYPTO_ALGOFAM_ED448` for the `x448` curve ▶ `CRYPTO_ALGOFAM_ECCNIST` for the NIST curves `secp256r1`, `secp384r1`, and `secp521r1` Section 3.4.10.3, "Retrieval of curve information" describes how the curves are retrieved from the cipher suite.
CsmSignatureGenerateAlgorithmSecondaryFamily	The parameter is configured depending on the first signature scheme that is associated with the cipher suite. Its value is set to one of the following values: ▶ `CRYPTO_ALGOFAM_SHA2_256` for the `ecdsa_secp256r1_sha256` scheme ▶ `CRYPTO_ALGOFAM_SHA2_384` for the `ecdsa_secp384r1_sha384` scheme ▶ `CRYPTO_ALGOFAM_SHA2_512` for the `ecdsa_secp521r1_sha512` scheme Section 3.4.10.4, "Retrieval of signature scheme information" describes how the signature schemes are retrieved from the cipher suite.

3.4.10.3. Retrieval of curve information

Curves are retrieved by collecting all `CRYPTO-SERVICE-PRIMITIVE` elements that are referenced via `TLS-CRYPTO-CIPHER-SUITE/KEY-EXCHANGE-REFS`. A curve is identified by its `TLS/groupId` SDG which is compared to the IDs listed in _TLS Supported Groups_.

3.4.10.4. Retrieval of signature scheme information

The signature schemes of a `TLS-CRYPTO-CIPHER-SUITE` are retrieved by collecting the comma separated IDs contained in the `TLS/signatureSchemes` SDG. These IDs are compared with the IDs listed in _TLS SignatureScheme_.

3.4.11. Dcm

Configuration parameters	Mapping description
`DcmConfigSet`	Check if a `DcmConfigSet` container is created inside the module. If one already exists then an additional one is not created, otherwise a new `DcmConfigSet` container is created.
`DcmConfigSet/DcmDsl/DcmDslProtocol/DcmDslProtocolRow`	For each `DIAGNOSTIC-SERVICE-TABLE` that is assigned to the configured `ECU-INSTANCE`, one `DcmDslProtocolRow` container is created. A `DIAGNOSTIC-CONNECTION` refers to a PDU sent or received by the configured `ECU-INSTANCE` in the following cases: ▶ The `DIAGNOSTIC-CONNECTION` references a `PDU-TRIGGERING` via `PERIODIC-RESPONSE-UUDT-REF` which in turn references an outgoing `I-PDU-PORT` of the `ECU-INSTANCE`. Moreover, the `PDU-TRIGGERING` must reference a PDU of type `DCM-I-PDU`. ▶ The `DIAGNOSTIC-CONNECTION` references a `TP-CONNECTION-IDENT` element via `FUNCTIONAL-REQUEST-REF`, `RESPONSE-REF`, `RESPONSE-ON-EVENT-REF`, or `PHYSICAL-REQUEST-REF` which in turn refers to one or more PDUs that are sent or received by the configured `ECU-INSTANCE`. A `TP-CONNECTION-IDENT` refers to a PDU in one of the following cases: ▶ The `TP-CONNECTION-IDENT` is aggregated by a `CAN-TP-CONNECTION` which refers to the PDU via `TP-SDU-REF`. ▶ The `TP-CONNECTION-IDENT` is aggregated by a `FLEXRAY-TP-CONNECTION` which refers to the PDU either via `DIRECT-TP-SDU-REF` or via `REVERSED-TP-SDU-REF`.

Configuration parameters	Mapping description
	▶ The `TP-CONNECTION-IDENT` is aggregated by a `LIN-TP-CONNECTION` which refers to the PDU via `LIN-TP-N-SDU-REF`. ▶ The `TP-CONNECTION-IDENT` is aggregated by a `DO-IP-TP-CONNEC-TION`, which refers to the PDU via `TP-SDU-REF`.
`DcmDslProtocol-Row/DcmDslProto-colID`	Each `DiagnosticProtocol` refers to at most one `DiagnosticServic-eTable` and to a collection of `DiagnosticConnection` elements. The `Diag-nosticProtocolID` parameter is configured to a certain category based on specific conditions for each category: ▶ If the reference `DiagnosticConnection.periodicResponseTp` exists and the reference `TpConnectionIdent.ident` belongs to a `CanTpCon-nection`, or the `DiagnosticConnection.periodicResponseUudt` is not empty and the `PhysicalChannel` belongs to `AbstractCanPhysi-calChannel`, then the `DcmDslProtocolID` is set to the `DCM_PERIODIC-TRANS_ON_CAN` category. ▶ If the reference `DiagnosticConnection.periodicResponseTp` exists and the reference `TpConnectionIdent.ident` belongs to a `FlexRayT-pConnection`, or the `DiagnosticConnection.periodicRespon-seUudt` is not empty and the `PhysicalChannel` belongs to `Flexray-PhysicalChannel`, then the `DcmDslProtocolID` is set to the `DCM_-PERIODIC_ON_FLEXRAY` category. ▶ If the reference `DiagnosticConnection.periodicResponseTp` ex-ists and the reference `TpConnectionIdent.ident` belongs to a `DoIpT-pConnection`, or the `DiagnosticConnection.periodicRespon-seUudt` is not empty and the `PhysicalChannel` belongs to `Ethernet-PhysicalChannel`, then the `DcmDslProtocolID` is set to the `DCM_-PERIODIC_ON_IP` category. ▶ If the reference `DiagnosticConnection.responseOnEvent` exists and the reference `TpConnectionIdent.ident` belongs to a `CanTpConnec-tion`, then the `DcmDslProtocolID` is set to the `DCM_ROE_ON_CAN` cate-gory. ▶ If the reference `DiagnosticConnection.responseOnEvent` exists and the reference `TpConnectionIdent.ident` belongs to a `FlexRayT-pConnection`, then the `DcmDslProtocolID` is set to `DCM_ROE_ON_-FLEXRAY` category. ▶ If the reference `DiagnosticConnection.responseOnEvent` exists and the reference `TpConnectionIdent.ident` belongs to a `DoIpTpCon-`

Configuration parameters	Mapping description
	nection, then the `DcmDslProtocolID` is set to the `DCM_ROE_ON_IP` category. ► If the reference `DiagnosticConnection.physicalRequest` exists and the reference `TpConnectionIdent.ident` belongs to a `CanTpConnection`, then the `DcmDslProtocolID` is set to the `DCM_UDS_ON_CAN` category. ► If the reference `DiagnosticConnection.physicalRequest` exists and the reference `TpConnectionIdent.ident` belongs to a `FlexRayTpConnection`, then the `DcmDslProtocolID` is set to the `DCM_UDS_ON_-FLEXRAY` category. ► If the reference `DiagnosticConnection.physicalRequest` exists and the reference `TpConnectionIdent.ident` belongs to a `DoIpTpConnection`, then the `DcmDslProtocolID` is set to the `DCM_UDS_ON_IP` category. ► If the reference `DiagnosticConnection.physicalRequest` exists and the reference `TpConnectionIdent.ident` belongs to a `CanTpConnection` and if the `DiagnosticProtocol.protocolKind` is valid and contains `OBD` as a String or if the `DiagnosticServiceTable.serviceInstance` has one or more references to `DiagnosticServiceInstance` supported by `OBD`, then the `DcmDslProtocolID` is set to the `DCM_OBD_-ON_CAN` category. ► If the reference `DiagnosticConnection.physicalRequest` exists and the reference `TpConnectionIdent.ident` belongs to a `FlexRayTpConnection` and if the `DiagnosticProtocol.protocolKind` is valid and contains `OBD` as a String or if the `DiagnosticServiceTable.serviceInstance` has one or more references to `DiagnosticServiceInstance` supported by `OBD`, then the `DcmDslProtocolID` is set to the `DCM_OBD_ON_FLEXRAY` category. ► If the reference `DiagnosticConnection.physicalRequest` exists and the reference `TpConnectionIdent.ident` belongs to a `DoIpTpConnection` and if the `DiagnosticProtocol.protocolKind` is valid and contains `OBD` as a String or if the `DiagnosticServiceTable.serviceInstance` has one or more references to `DiagnosticServiceInstance` supported by `OBD`, then the `DcmDslProtocolID` is set to the `DCM_OBD_ON_IP` category.
`DcmDslProtocol-Row/DcmDslProtocol-Priority`	Each `DiagnosticProtocol` refers to at most one `DiagnosticServiceTable` and to a collection of `DiagnosticConnection` elements.

Configuration parameters	Mapping description
	If the reference `DiagnosticProtocol.serviceTable` exists, and the value of `DiagnosticProtocol.protocolKind` is identical to the value of `DiagnosticServiceTable.protocolKind`, then the `DcmDslProtocolPriority` is set and it represents the priority of the diagnostic protocol in comparison to other diagnostic protocols.
DcmDslBuffer	A `DcmDslBuffer` is configured. The size of `DcmDslBuffer` is set to have the default value, 8.
`DcmDslProtocol-Row/DcmDslProtocolSessionRef`	For every `DcmDslProtocolRow` a `DcmDslProtocolSessionRef` reference is set.
`DcmDslProtocol-Row/DcmDemClientRef`	The `DcmDemClientRef` reference in `DcmDslProtocolRow` is configured to refer to a `DemClient` container if at least one of the `DiagnosticServiceInstance` elements of the `DiagnosticServiceTable` is of any of the types listed below. The `DcmDemClientRef` is set to one of the following UDS supported services: ▶ `DiagnosticReadDTCInformation` for `DIAGNOSTIC-READ-DTC-INFORMATION`. ▶ `DiagnosticClearDiagnosticInformation` for `DIAGNOSTIC-CLEAR-DIAGNOSTIC-INFORMATION`. ▶ `DiagnosticControlDTCSetting` for `DIAGNOSTIC-CONTROL-DTC-SETTING`. The `DcmDemClientRef` is set to one of the following OBD supported services: ▶ `DiagnosticRequestVehicleInfo` for `DIAGNOSTIC-REQUEST-VEHICLE-INFO`. ▶ `DiagnosticRequestOnBoardMonitoringTestResults` for `DIAGNOSTIC-REQUEST-ON-BOARD-MONITORING-TEST-RESULTS`. ▶ `DiagnosticRequestPowertrainFreezeFrameData` for `DIAGNOSTIC-REQUEST-POWERTRAIN_FREEZE-FRAME-DATA`. ▶ `DiagnosticRequestCurrentPowertrainDiagnosticData` for `DIAGNOSTIC-REQUEST-CURRENT-POWERTRAIN-DIAGNOSTIC-DATA`. ▶ `DiagnosticRequestEmissionRelatedDTCPermanentStatus` for `DIAGNOSTIC-REQUEST-EMISSION-RELATED-DTC-PERMANENT-STATUS`.

Configuration parameters	Mapping description
	▶ `DiagnosticClearResetEmissionRelatedInfo` for `DIAG-NOSTIC-CLEAR-RESET-EMISSION-RELATED-INFO`.
`DcmDslProtocol-Row/DcmDslRespon-seOnEvent`	A `DcmDslResponseOnEvent` subcontainer is created if the `DIAG-NOSTIC-CONNECTION` elements of the `DIAGNOSTIC-SERVICE-TABLE` refer to a PDU that the configured `ECU-INSTANCE` sends via `RESPONSE-ON-EVENT-REF`. The name of the container is set to `<PREFIX><name>_ResponseOn-Event`, where `<name>` is the `SHORT-NAME` of the `DIAGNOSTIC-CONNECTION`. `DcmDslRoeTxPduRef` is configured to refer to the `EcuC` container that was created for the sent PDU.
`DcmDslProtocol-Row/DcmDslMainCon-nection`	A `DcmDslMainConnection` subcontainer is created for a `DIAGNOSTIC-CON-NECTION` of a `DIAGNOSTIC-SERVICE-TABLE` if the `DIAGNOSTIC-CONNEC-TION` refers via `FUNCTIONAL-REQUEST-REF, PHYSICAL-REQUEST-REF,` or `RESPONSE-REF` to one or more `TP-CONNECTION-IDENT` elements that are in turn contained in `TP-CONNECTION` elements via which the configured `ECU-INSTANCE` sends or receives PDUs. The name of the container is set to `<PREFIX><name>_Main`, where `<name>` is the `SHORT-NAME` of the `DIAG-NOSTIC-CONNECTION`. `DcmDslProtocolRxTesterSourceAddr` is set to the diagnostic address of the remote node. The retrieval of that address depends on the type of `TP-CON-NECTION` instance that contains the `TP-CONNECTION-IDENT`. Any type of `TP-CONNECTION` that is not listed below is not supported. ▶ `CAN-TP-CONNECTION` instances: If `CAN-TP-CONNECTION/TRANSMIT-TER` does not refer to the configured `ECU-INSTANCE`, `CAN-TP-CONNEC-TION/TRANSMITTER` represents the remote node. If `CAN-TP-CONNEC-TION/TRANSMITTER` represents the configured `ECU-INSTANCE`, the remote node can only be determined if the `CAN-TP-CONNECTION` sends its PDUs to a physical address. In that case, the remote node is the first `CAN-TP-NODE` referenced by `CAN-TP-CONNECTION/RECEIVER`. The tester address is retrieved by taking the `TP-ADDRESS` value of the `CAN-TP-AD-DRESS` entry which the remote node refers to via `TP-ADDRESS`. ▶ `FLEXRAY-TP-CONNECTION` instances: If `FLEXRAY-TP-CONNEC-TION/TRANSMITTER` does not refer to the configured `ECU-INSTANCE`, `FLEXRAY-TP-CONNECTION/TRANSMITTER` represents the remote node. If `FLEXRAY-TP-CONNECTION/TRANSMITTER` represents the configured `ECU-INSTANCE`, the remote node can only be determined if the `FLEXRAY-TP-CONNECTION` sends its PDUs to a physical address. In that case, the remote node is the first `FLEXRAY-TP-NODE` referenced by `FLEXRAY-TP-`

Configuration parameters	Mapping description
	CONNECTION/RECEIVER. The tester address is retrieved by taking the TP-ADDRESS/TP-ADDRESS value which the remote FLEXRAY-TP-NODE refers to via TP-ADDRESS. ▶ DO-IP-TP-CONNECTION instances: If the TP-CONNECTION-IDENT is referenced via DIAGNOSTIC-CONNECTION/FUNCTIONAL-REQUEST or DIAGNOSTIC-CONNECTION/PHYSICAL-REQUEST, DO-IP-TP-CONNECTION/DO-IP-SOURCE-ADDRESS represents the address of the tester node. If the TP-CONNECTION-IDENT is referenced via DIAGNOSTIC-CONNECTION/RESPONSE, DO-IP-TP-CONNECTION/DO-IP-TARGET-ADDRESS represents the address of the tester node. The actual tester address value is taken from DO-IP-LOGIC-ADDRESS/ADDRESS. All diagnostic addresses are collected for all TP-CONNECTION-IDENT elements which the DIAGNOSTIC-CONNECTION references via FUNCTIONAL-REQUEST-REF, PHYSICAL-REQUEST-REF, or RESPONSE-REF. The first element of the resulting list of diagnostic addresses is used for the configuration of DcmDslProtocolRxTesterSourceAddr. If the DIAGNOSTIC-CONNECTION references a PDU via RESPONSE-ON-EVENT-REF, DcmDslROEConnectionRef is configured to reference the EcuC container that was created for that PDU.
DcmDslProtocolRow/DcmDslMainConnection/DcmDslProtocolTx	A DcmDslProtocolTx subcontainer is created if the DIAGNOSTIC-CONNECTION refers to one or more PDUs that the configured ECU-INSTANCE sends via RESPONSE-REF. The container name is <PREFIX><name>, where <name> is the SHORT-NAME of the received PDU. DcmDslProtocolTxPduRef is configured to reference the EcuC container that was created for the sent PDU.
DcmDslProtocolRow/DcmDslMainConnection/DcmDslProtocolRx	A DcmDslProtocolRx subcontainer is created if the DIAGNOSTIC-CONNECTION refers to one or more PDUs that the configured ECU-INSTANCE receives via FUNCTIONAL-REQUEST_REF or PHYSICAL_REQUEST-REF. The container name is <PREFIX><name>, where <name> is the SHORT-NAME name of the received PDU. DcmDslProtocolRxAddrType is set to DCM_PHYSICAL_TYPE if the received PDU is referenced by the DIAGNOSTIC-CONNECTION via PHYSICAL-REQUEST-REF. It is set to DCM_FUNCTIONAL_TYPE if the received PDU is referenced by the DIAGNOSTIC-CONNECTION via FUNCTIONAL-REQUEST-REF. DcmDslProtocolRxPduRef is configured to reference the EcuC container that was configured for the received PDU.

Configuration parameters	Mapping description
	If a `ComMChannel` container was created in the `ComM` for the `COMMUNICATION-CLUSTER` or the `ETHERNET-PHYSICAL-CHANNEL` via which the PDU is received, `DcmDslProtocolRxComMChannelRef` is configured to reference that `ComMChannel` container.
`DcmDslProtocolRow/DcmDslPeriodicTransmission`	A `DcmDslPeriodicTransmission` subcontainer is created if the `DIAGNOSTIC-CONNECTION` elements of the `DIAGNOSTIC-SERVICE-TABLE` refer via `PERIODIC-RESPONSE-UUDT-REF` to one or more PDUs that the configured `ECU-INSTANCE` sends. The name of the container is set to `<PREFIX><name>_PeriodicTransmission`, where `<name>` is the mangled name of the `DIAGNOSTIC-CONNECTION`. For each PDU that the configured `ECU-INSTANCE` sends in the context of the `DIAGNOSTIC-CONNECTION`, one `DcmDslPeriodicConnection` container is created. Its name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` name of the PDU. `DcmDslPeriodicTxPduRef` is configured to reference the `EcuC` container that was created for the sent PDU.
`DcmConfigSet/DcmDsp`	`DcmDspMaxDidToRead` is set to the maximum `MAX-DID-TO-READ` value of any `DIAGNOSTIC-READ-DATA-BY-IDENTIFIER-CLASS` that is referenced by a `DIAGNOSTIC-READ-DATA-BY-IDENTIFIER` element of the imported `ECU-INSTANCE`. `DcmDspMaxPeriodicDidToRead` is set to the maximum `MAX-PERIODIC-DID-TO-READ` value of any `DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID-CLASS` that is referenced by a `DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID` element of the imported `ECU-INSTANCE`. For the configuration of `DcmDspDDDIDcheckPerSourceDID`, the `CHECK-PER-SOURCE-ID` values of all `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER-CLASS` entities referenced by `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER` elements of the imported `ECU-INSTANCE` are collected. If at least one `CHECK-PER-SOURCE-ID` value is set to `true`, `DcmDspDDDIDcheckPerSourceDID` is set to `true`. If all `CHECK-PER-SOURCE-ID` values are set to `false`, then `DcmDspDDDIDcheckPerSourceDID` is set to `false`.
`DcmConfigSet/DcmDsp/DcmDspMemory`	A single `DcmDspMemory` sub container is created for all the `DIAGNOSTIC-READ-MEMORY-BY-ADRESS` and all the `DIAGNOSTIC-WRITE-MEMORY-BY-ADRESS` services referenced by a `DIAGNOSTIC-SERVICE-TABLE` for the current ECU.
`DcmConfigSet/DcmDsp/DcmDspMemo-`	For each different `DIAGNOSTIC-MEMORY-IDENTIFIER` referenced by a `DIAGNOSTIC-READ-MEMORY-BY-ADRESS` or a `DIAGNOSTIC-WRITE-MEMORY-BY-`

Configuration parameters	Mapping description
`ry/DcmDspMemoryIdInfo`	`ADRESS` service, belonging to a `DIAGNOSTIC-SERVICE-TABLE`, a single `DcmDspMemoryIdInfo` is created. `DcmDspMemoryIdValue` is set to the `ID` value of the `DIAGNOSTIC-MEMORY-IDENTIFIER`.
`DcmConfigSet/DcmDsp/DcmDspMemory/DcmDspMemoryIdInfo/DcmDspWriteMemoryRangeInfo`	A single `DcmDspWriteMemoryRangeInfo` subcontainer is created for each `DIAGNOSTIC-MEMORY-IDENTIFIER` that is referenced by a `DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS`. `DcmDspWriteMemoryRangeHigh` is set to the `MEMORY-HIGH-ADDRESS` value of the `DIAGNOSTIC-MEMORY-IDENTIFIER` referenced by a `DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS` service. `DcmDspWriteMemoryRangeLow` is set to the `MEMORY-LOW-ADDRESS` value of the `DIAGNOSTIC-MEMORY-IDENTIFIER` referenced by a `DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS` service. `DcmDspWriteMemoryRangeSecurityLevelRef` references all the corresponding `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS` service.
`DcmConfigSet/DcmDsp/DcmDspMemory/DcmDspMemoryIdInfo/DcmDspReadMemoryRangeInfo`	A single `DcmDspReadMemoryRangeInfo` subcontainer is created for each `DIAGNOSTIC-MEMORY-IDENTIFIER` that is referenced by a `DIAGNOSTIC-READ-MEMORY-BY-ADDRESS`. `DcmDspReadMemoryRangeHigh` is set to the `MEMORY-HIGH-ADDRESS` value of the `DIAGNOSTIC-MEMORY-IDENTIFIER` referenced by a `DIAGNOSTIC-READ-MEMORY-BY-ADDRESS` service. `DcmDspReadMemoryRangeLow` is set to the `MEMORY-LOW-ADDRESS` value of the `DIAGNOSTIC-MEMORY-IDENTIFIER` referenced by a `DIAGNOSTIC-READ-MEMORY-BY-ADDRESS` service. `DcmDspReadMemoryRangeSecurityLevelRef` references all the corresponding `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-READ-MEMORY-BY-ADDRESS` service.

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsp/DcmDspMemory/DcmDspMemoryIdInfo/DcmDspWriteMemoryRangeByLabelInfo`	A single `DcmDspWriteMemoryRangeByLabelInfo` subcontainer is created for each `DIAGNOSTIC-MEMORY-IDENTIFIER` that is referenced by a `DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS`. `DcmDspWriteMemoryRangeByLabelHigh` is set to the `MEMORY-HIGH-ADDRESS-LABEL` value of the `DIAGNOSTIC-MEMORY-IDENTIFIER` referenced by a `DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS` service. `DcmDspWriteMemoryRangeByLabelLow` is set to the `MEMORY-LOW-ADDRESS-LABEL` value of the `DIAGNOSTIC-MEMORY-IDENTIFIER` referenced by a `DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS` service. `DcmDspWriteMemoryRangeSecurityLevelRef` references all the corresponding `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS` service.
`DcmConfigSet/DcmDsp/DcmDspMemory/DcmDspMemoryIdInfo/DcmDspReadMemoryRangeByLabelInfo`	A single `DcmDspReadMemoryRangeByLabelInfo` subcontainer is created for each `DIAGNOSTIC-MEMORY-IDENTIFIER` that is referenced by a `DIAGNOSTIC-READ-MEMORY-BY-ADDRESS`. `DcmDspReadMemoryRangeByLabelHigh` is set to the `MEMORY-HIGH-ADDRESS-LABEL` value of the `DIAGNOSTIC-MEMORY-IDENTIFIER` referenced by a `DIAGNOSTIC-READ-MEMORY-BY-ADDRESS` service. `DcmDspReadMemoryRangeByLabelLow` is set to the `MEMORY-LOW-ADDRESS-LABEL` value of the `DIAGNOSTIC-MEMORY-IDENTIFIER` referenced by a `DIAGNOSTIC-READ-MEMORY-BY-ADDRESS` service. `DcmDspReadMemoryRangeSecurityLevelRef` references all the corresponding `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-READ-MEMORY-BY-ADDRESS` service.
`DcmConfigSet/DcmDsp/DcmDspComControl/DcmDspComControlAllChannel`	A `DcmDspComControlAllChannel` container is created for every different `DIAGNOSTIC-ALL-CHANNEL` element of a `DIAGNOSTIC-COM-CONTROL-CLASS` of the current ECU. The container name is `<name><SUFFIX>`, where `<name>` is derived from the short name of the `DIAGNOSTIC-ALL-CHANNEL` element and the `<SUFFIX>` is `_AC_` and the number of the current container created.

Configuration parameters	Mapping description
	`DcmDspAllComMChannelRef` is set to `DIAGNOSTIC-COM-CONTROL-CLASS/ALL-CHANNEL` reference.
`DcmConfigSet/DcmDsp/DcmDspComControl/DcmDspComControlSpecificChannel`	A `DcmDspComControlSpecificChannel` container is created for every different pair of `SPECIFIC-CHANNEL` and `SUBNET-NUMBER` elements from a `DIAGNOSTIC-COM-CONTROL-SPECIFIC-CHANNEL` of a `DIAGNOSTIC-COM-CONTROL-CLASS` of the current ECU. The container name is `<name><SUFFIX>`, where `<name>` is derived from the short name of the `SPECIFIC-CHANNEL` element and the `<SUFFIX>` is `_SC_` and the number of the current container created. `DcmDspSubnetNumber` is set to the value of `DIAGNOSTIC-COM-CONTROL-SPECIFIC-CHANNEL/SUBNET-NUMBER`. `DcmDspSpecificComMChannelRef` is set to `DIAGNOSTIC-COM-CONTROL-CLASS/SPECIFIC-CHANNEL` reference.
`DcmConfigSet/DcmDsp/DcmDspComControl/DcmDspComControlSubNode`	A `DcmDspComControlSubNode` container is created for every different pair of `SUBNODE-CHANNEL` and `SUBNODE-NUMBER` elements from a `DIAGNOSTIC-COM-CONTROL-SUBNODE-CHANNEL` of a `DIAGNOSTIC-COM-CONTROL-CLASS` of the current ECU. The container name is `<name><SUFFIX>`, where `<name>` is derived from the short name of the `SUBNODE-CHANNEL` element and the `<SUFFIX>` is `_SubNode_` and the number of the current container created. `DcmDspComControlSubNodeId` is set to the value of `DIAGNOSTIC-COM-CONTROL-SUBNODE-CHANNEL/SUBNODE-NUMBER`. `DcmDspComControlSubNodeComMChannelRef` is set to `DIAGNOSTIC-COM-CONTROL-SUBNODE-CHANNEL/SUBNODE-CHANNEL` reference.
`DcmConfigSet/DcmDsp/DcmDspSession`	A `DcmDspSession` subcontainer is created for every `DIAGNOSTIC-SESSION` referenced by a `DIAGNOSTIC-ACCESS-PERMISSION` associated with any `DiagnosticServiceInstance` of the current ECU. For more information on how the `DIAGNOSTIC-ACCESS-PERMISSION` of a `DiagnosticServiceInstance` is retrieved, see Section 3.4.11.22, "Determining the access permissions for diagnostic service instances". The container name is `DIAGNOSTIC-SESSION/SHORT-NAME` of the `DIAGNOSTIC-SESSION-CONTROL`. `DcmDspSessionForBoot` is set to `DIAGNOSTIC-SESSION/JUMP-TO-BOOT-LOADER`. `DcmDspSessionLevel` is set to `DIAGNOSTIC-SESSION/ID`.

Configuration parameters	Mapping description
	`DcmDspSessionP2ServerMax` is set to `DIAGNOSTIC-SESSION/P-2-SERVER-MAX`. `DcmDspSessionP2StarServerMax` is set to `DIAGNOSTIC-SESSION/P-2-STAR-SERVER-MAX`.
`DcmConfigSet/DcmDsp/DcmDspSecurity`	A `DcmDspSecurity` subcontainer is created for every `DIAGNOSTIC-SECURITY-ACCESS` referenced by a `DIAGNOSTIC-ACCESS-PERMISSION` associated with any `DiagnosticServiceInstance` of the current ECU. For more information on how the `DIAGNOSTIC-ACCESS-PERMISSION` of a `DiagnosticServiceInstance` is retrieved, see [Section 3.4.11.22, "Determining the access permissions for diagnostic service instances"](). The container name is `DIAGNOSTIC-SECURITY-LEVEL/SHORT-NAME` of the `DIAGNOSTIC-SECURITY-ACCESS`. `DcmDspSecurityADRSize` is set to `DIAGNOSTIC-SECURITY-LEVEL/ACCESS-DATA-RECORD-SIZE`. `DcmDspSecurityDelayTimeOnBoot` is set to `DIAGNOSTIC-SECURITY-LEVEL/SECURITY-DELAY-TIME`. `DcmDspSecurityKeySize` is set to `DIAGNOSTIC-SECURITY-LEVEL/KEY-SIZE`. `DcmDspSecurityLevel` is set to (`DIAGNOSTIC-SECURITY-ACCESS/REQUEST-SEED-ID` + 1) / 2. `DcmDspSecurityNumAttDelay` is set to `DIAGNOSTIC-SECURITY-LEVEL/NUM-FAILED-SECURITY-ACCESS`. `DcmDspSecuritySeedSize` is set to `DIAGNOSTIC-SECURITY-LEVEL/SEED-SIZE`.
`DcmConfigSet/DcmDsp/DcmDspDataUsePort`	A `DcmDspDataUsePort` subcontainer indicates the interface that shall be used to access the data. `USE_DATA_ASYNCH_CLIENT_SERVER` if `DiagnosticProcessingStyleEnum` is equal to `processingStyleAsynchronous`. The `DiagnosticProcessingStyleEnum` is taken from the `DiagnosticServiceSwMapping` that has a `SwcServiceDependency` taken directly from `MappedFlatSwcServiceDependency` or indirectly from `MappedSwcServiceDependency` with a target value taken from the `SwcServiceDependencyInCompositionRef`.

Configuration parameters	Mapping description
	`USE_DATA_ASYNCH_FNC` if `DiagnosticServiceSwMapping` has a `BswServiceDependency` and `DiagnosticProcessingStyleEnum` is equal to `processingStyleAsynchronous`. `USE_DATA_SYNCH_CLIENT_SERVER` if `DiagnosticProcessingStyleEnum` is equal to `processingStyleSynchronous`. The `DiagnosticProcessingStyleEnum` is taken from the `DiagnosticServiceSwMapping` that has a `SwcServiceDependency` taken directly from `MappedFlatSwcServiceDependency` or indirectly from `MappedSwcServiceDependency` with a target value taken from the `SwcServiceDependencyInCompositionRef`. `USE_DATA_SYNCH_FNC` if `DiagnosticServiceSwMapping` has a `BswServiceDependency` and `DiagnosticProcessingStyleEnum` is equal to `processingStyleSynchronous`. If `USE_DATA_ASYNCH_CLIENT_SERVER` or `USE_DATA_SYNCH_CLIENT_-SERVER` is set on `DcmDspDataUsePort` then the value of the parameter `/Dcm/DcmConfigSet/DcmGeneral/DcmRteUsage` is set to true.
`DcmConfigSet/DcmDsp/DcmDspControlDTCSetting`	A `DcmDspControlDTCSetting` subcontainer is created for the first occurence of `DIAGNOSTIC-CONTROL-DTC-SETTING`. The container name is set to `DcmDspControlDTCSetting`. `DcmSupportDTCSettingControlOptionRecord` is set to `DIAGNOSTIC-CONTROL-DTC-SETTING-CLASS/CONTROL-OPTION-RECORD-PRESENT` referenced by `DIAGNOSTIC-CONTROL-DTC-SETTING`.
`DcmConfigSet/DcmDsp/DiagnosticReadDataByPeriodicID`	A `DiagnosticReadDataByPeriodicID` subcontainer is created for the first occurence of `DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID`. The container name is set to `DiagnosticReadDataByPeriodicID`. `DiagnosticPeriodicRate` is set to `DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID-CLASS/DIAGNOSTIC-PERIODIC-RATE` referenced by `DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID`. referenced by `DIAGNOSTIC-PERIODIC-TRANSMISSION` `DcmDspMaxPeriodicDidScheduler` is set to maximum number of periodicDataIdentifiers that can be scheduled concurrently `DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID-CLASS/SCHEDULER-MAX-NUMBER` referenced by `DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID`.
`DcmConfigSet/DcmDsp/DcmDspPeriodicTransmission`	For the configuration of `DcmDspPeriodicTransmissionSlowRate`, `DcmDspPeriodicTransmissionMediumRate`, and `DcmDspPeriodicTransmissionFastRate`, all `DIAGNOSTIC-PERIODIC-RATE` elements of all `DIAG-`

Configuration parameters	Mapping description
	`NOSTIC-READ-DATA-BY-PERIODIC-ID-CLASS` elements referenced by `DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID` of the imported `ECU-INSTANCE` are collected and grouped according to their `PERIODIC-RATE-CATEGORY` value. `DcmDspPeriodicTransmissionSlowRate` is set to the smallest value of category `PERIODIC-RATE-SLOW`. `DcmDspPeriodicTransmissionMediumRate` is set to the smallest value of category `PERIODIC-RATE-MEDIUM`. `DcmDspPeriodicTransmissionMediumRate` is set to the smallest value of category `PERIODIC-RATE-FAST`.
`DcmConfigSet/DcmDsp/DiagnosticDynamicallyDefineDataIdentifier`	A `DiagnosticDynamicallyDefineDataIdentifier` subcontainer is created for the first occurence of `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER`. The container name is set to `DiagnosticDynamicallyDefineDataIdentifier`. `DcmDDDIDStorage` is set to `true` if all `DiagnosticDynamicallyDefineDataIdentifier` are stored `NON-VOLATILE` > `DcmDDDIDStorage` is set to `false` if any of `DiagnosticDynamicallyDefineDataIdentifier` are stored as `VOLATILE` `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER-CLASS/CONFIGURATION-HANDLING` referenced by `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER`.
`DcmConfigSet/DcmDsp/DiagnosticRequestControlOfOnBoardDevice`	A `DiagnosticRequestControlOfOnBoardDevice` subcontainer is created for the first occurence of `DIAGNOSTIC-REQUEST-CONTROL-OF-ON-BOARD-DEVICE`. The container name is set to `DiagnosticRequestControlOfOnBoardDevice`. `DiagnosticTestRoutineIdentifier` is set to `DIAGNOSTIC-REQUEST-CONTROL-OF-ON-BOARD-DEVICE/DIAGNOSTIC-TEST-ROUTINE-IDENTIFIER`. For every `DiagnosticTestRoutineIdentifier` `DcmDspRequestControlTestId` is set to `DIAGNOSTIC-TEST-ROUTINE-IDENTIFIER/ID`. `DcmDspRequestControlInBufferSize` is set to `DIAGNOSTIC-TEST-ROUTINE-IDENTIFIER/REQUEST-DATA-SIZE`. `DcmDspRequestControlOutBufferSize` is set to `DIAGNOSTIC-TEST-ROUTINE-IDENTIFIER/RESPONSE-DATA-SIZE`.
`DcmConfigSet/DcmDsp/DcmDspData`	For every `DIAGNOSTIC-DATA-IDENTIFIER/DIAGNOSTIC-PARAMETER/DIAGNOSTIC-DATA-ELEMENT` referenced by `DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER` and `DIAGNOSTIC-READ-DATA-BY-IDENTIFIER` service instance which belongs to the imported `ECU-INSTANCE`, a `DcmDspData` container is created. The container name is `<PREFIX><name>`, where `<PREFIX>`

Configuration parameters	Mapping description
	is `DspData_` + `I` where `I` is a counter and `<name>` is the `DIAGNOSTIC-DATA-ELEMENT/SHORT-NAME`. `DcmDspDataType` is set depending on `SW-BASE-TYPE/BASE-TYPE-ENCODING` and `SW-BASE-TYPE/BASE-TYPE-SIZE`: ► `UINT8` for `BASE-TYPE-ENCODING NONE` and `BASE-TYPE-SIZE 8` ► `UINT16` for `BASE-TYPE-ENCODING NONE` and `BASE-TYPE-SIZE 16` ► `UINT32` for `BASE-TYPE-ENCODING NONE` and `BASE-TYPE-SIZE 32` ► `SINT8` for `BASE-TYPE-ENCODING 2C` and `BASE-TYPE-SIZE 8` ► `SINT16` for `BASE-TYPE-ENCODING 2C` and `BASE-TYPE-SIZE 16` ► `SINT32` for `BASE-TYPE-ENCODING 2C` and `BASE-TYPE-SIZE 32` ► `BOOLEAN` for `BASE-TYPE-ENCODING BOOLEAN` and `BASE-TYPE-SIZE 1` or `BASE-TYPE-SIZE 8` `DcmDspDataSize` is set depending on the `ARRAY-SIZE-SEMANTICS` if it is `FIXED-SIZE`. Then the `BASE-TYPE-SIZE` associated with the `DIAGNOSTIC-DATA-ELEMENT` is obtained and set to the value `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` * `BASE-TYPE-SIZE`. If the `ARRAY-SIZE-SEMANTICS` value of the `DIAGNOSTIC-DATA-ELEMENT` is `VARIABLE-SIZE`, the `DcmDspDataSize` value is set to `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS`*8. If `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` is missing then its value is considered 0. If `ARRAY-SIZE-SEMANTICS` is not set then the value of `DcmDspDataSize` is set to the size given by `BASE-TYPE-SIZE`. For example 8 for UINT8, 16 for UINT16. `DcmDspDataInfoRef` is set to `DcmConfigSet/DcmDsp/DcmDspDataInfo/DcmDspDataInfo_0` if the `ArraySizeSemantics` value of the `DiagnosticDataElement` is `FIXED-SIZE` else it references `DcmConfigSet/DcmDsp/DcmDspDataInfo/DcmDspDataInfo_1` if the `ArraySizeSemantics` value of the `DiagnosticDataElement` is `VARIABLE-SIZE`. `DcmDspDataEndianness` is set to `BIG_ENDIAN` if `BASE-TYPE/BYTE-ORDER` is defined as `MOST-SIGNIFICANT-BYTE-FIRST` or set to `LITTLE_ENDIAN` if `BASE-TYPE/BYTE-ORDER` is defined as `MOST-SIGNIFICANT-BYTE-LAST` or set to `OPAQUE` if `BASE-TYPE/BYTE-ORDER` is defined as `OPAQUE`.
`DcmConfigSet/DcmDsp/DcmDspDidInfo`	For each different Diagnostic-Data-Identifier referenced by a `DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER`, `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER` or `DIAGNOSTIC-READ-`

Configuration parameters	Mapping description
	`DATA-BY-IDENTIFIER` service instance which belongs to the imported `ECU-INSTANCE` a single DcmDspDidInfo is created if `DcmDspDidDynamicallyDefined`,`DcmDspDDDIDMaxElements`, and `DcmDspDidAccess` parameters are the same in every service instance. If the service instance is a `DIAGNOSTIC-IO-CONTROL` then the following parameters are also checked: `FreezeCurrentState`, `ResetToDefault`, and `ShortTermAdjustment`. If they are different, a new `DcmDspDidInfo` is created.If any two of the above service instances refers a different Diagnostic-Data-Identifier,however have the same AccessPermission then an extra check is done in order to see if there is already a matching DcmDspDidInfo that can be used or if a new one needs to be created. The information contained by `DcmDspDidInfo` will not be the same. The container name is set to `DcmDspDidInfo_<suffix>`, where the `<suffix>` is set to the number of the container created. `DcmDspDidDynamicallyDefined` is set to true or false according to `DataIdentifier` from `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER`. `DcmDspDDDIDMaxElements` is set to the maximum `MAX-SOURCE-ELEMENT` value of all `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER` elements that refer via `DYNAMICALLY-DEFINE-DATA-IDENTIFIER-CLASS-REF` to the `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER-CLASS` element for which the `DcmDspDidInfo` container has been created.
`DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidAccess/DcmDspDidRead`	If at least one `DIAGNOSTIC-READ-DATA-BY-IDENTIFIER` exists, a `DcmDspDidRead` container is created. `DcmDspDidReadSecurityLevelRef` references all the corresponding `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-READ-DATA-BY-IDENTIFIER` references. For more information, see also [Section 3.4.11.22, "Determining the access permissions for diagnostic service instances"](#). `DcmDspDidReadSessionRef` references all the associated `DcmDspSession` containers that correspond to the `DIAGNOSTIC-SESSION` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-READ-DATA-BY-IDENTIFIER` references. For more information, see also [Section 3.4.11.22, "Determining the access permissions for diagnostic service instances"](#).
`DcmConfigSet/DcmDsp/DcmDspDidIn-`	If at least one `DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER` exists, a `DcmDspDidWrite` container is created.

Configuration parameters	Mapping description
fo/DcmDspDidAccess/DcmDspDidWrite	DcmDspDidWriteSecurityLevelRef references all the corresponding DcmDspSecurity containers that correspond to the DIAGNOSTIC-SECURITY-LEVEL entities which the DIAGNOSTIC-ACCESS-PERMISSION associated with the DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER references. For more information, see also Section 3.4.11.22, "Determining the access permissions for diagnostic service instances". DcmDspDidWriteSessionRef references all the associated DcmDspSession containers that correspond to the DIAGNOSTIC-SESSION entities which the DIAGNOSTIC-ACCESS-PERMISSION associated with the DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER references. For more information, see also Section 3.4.11.22, "Determining the access permissions for diagnostic service instances".
DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidAccess/DcmDspDidControl	If at least one DIAGNOSTIC-IO-CONTROL exists, a DcmDspDidControl container is created. DcmDspDidControlSecurityLevelRef references all the corresponding DcmDspSecurity containers that correspond to the DIAGNOSTIC-SECURITY-LEVEL entities which the DIAGNOSTIC-ACCESS-PERMISSION associated with the DIAGNOSTIC-IO-CONTROL references. For more information, see also Section 3.4.11.22, "Determining the access permissions for diagnostic service instances". DcmDspDidControlSessionRef references all the associated DcmDspSession containers that correspond to the DIAGNOSTIC-SESSION entities which the DIAGNOSTIC-ACCESS-PERMISSION associated with the DIAGNOSTIC-IO-CONTROL references. For more information, see also Section 3.4.11.22, "Determining the access permissions for diagnostic service instances".
DcmConfigSet/DcmDsp/DcmDspDid	For every DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER and DIAGNOSTIC-READ-DATA-BY-IDENTIFIER service instance which belongs to the imported ECU-INSTANCE, a DcmDspDid container is created. The container name is <PREFIX><name>, where <PREFIX> is DspDid_ + I where I is a counter and <name> is set to DIAGNOSTIC-DATA-IDENTIFIER/SHORT-NAME of DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER and DIAGNOSTIC-READ-DATA-BY-IDENTIFIER. DcmDspDidIdentifier is set to DIAGNOSTIC-DATA-IDENTIFIER/ID. DcmDspDidInfoRef references the corresponding DcmDspDidInfo container.

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidSignal`	For every `DIAGNOSTIC-DATA-IDENTIFIER/DIAGNOSTIC-PARAMETER` referenced by `DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER` and `DIAGNOSTIC-READ-DATA-BY-IDENTIFIER` service instance which belongs to the imported `ECU-INSTANCE`, a `DcmDspDidSignal` container is created. The container name is `DcmDspDidSignal_<suffix>`, where `<suffix>` is set to the zero-based position index of the `DIAGNOSTIC-PARAMETER` within the `DIAGNOSTIC-DATA-IDENTIFIER`. `DcmDspDidDataPos` is set to `DIAGNOSTIC-PARAMETER/BIT-OFFSET`. `DcmDspDidDataRef` references the `DcmDspDid` created which was created to represent the `DIAGNOSTIC-DATA-ELEMENT` of the current `DIAGNOSTIC-PARAMETER`.
`DcmConfigSet/DcmDsp/DcmDspRoutine`	For every `DIAGNOSTIC-ROUTINE-CONTROL` service instance which belongs to the imported `ECU-INSTANCE`, a `DcmDspRoutine` container is created. The container name is set to `DIAGNOSTIC-ROUTINE/SHORT-NAME` of `DIAGNOSTIC-ROUTINE-CONTROL`. `DcmDspRoutineIdentifier` is set to `DIAGNOSTIC-ROUTINE/ID`. `DcmDspRoutineInfoRef` references the corresponding `DcmDspRoutineInfo` container. `DcmDspRoutineUsePort` is set to `true` if the `DIAGNOSTIC-ROUTINE` is configured for a `SWC-SERVICE-DEPENDENCY` based on a `MAPPED-FLAT-SWC-SERVICE-DEPENDENCY` or a `MAPPED-SWC-SERVICE-DEPENDENCY` via `DIAGNOSTIC-SERVICE-SW-MAPPING`. If at least one `DcmDspRoutineUsePort` is set to true, the value of the parameter `/Dcm/DcmConfigSet/DcmGeneral/DcmRteUsage` is set to true. Is set to `true`, if the `DiagnosticRoutine` has a `DiagnosticRequestRoutineResults` configured. Is set to `true`, if the `DiagnosticRoutine` has a `DiagnosticStopRoutine` configured. `DcmDspRoutineFixedLength` is set to `false` if all the signals referenced by the `DIAGNOSTIC-ROUTINE` are of type `VARIABLE_SIZE`. Otherwise, the `DcmDspRoutineFixedLength` is set to `true`.
`DcmConfigSet/DcmDsp/DcmDspRoutineInfo`	For every `DIAGNOSTIC-ROUTINE-CONTROL` service instance which belongs to the imported `ECU-INSTANCE`, a `DcmDspRoutineInfo` container is created. The container name is set to `DcmDspRoutineInfo_<suffix>`, where `<suf-`

Configuration parameters	Mapping description
	`fix>` is set to `DIAGNOSTIC-ROUTINE/ID` of `DIAGNOSTIC-ROUTINE-CONTROL`.
`DcmConfigSet/DcmDsp/DcmDspRoutineInfo/DcmDspRoutineAuthorization`	If at least one `DIAGNOSTIC-ROUTINE-CONTROL` exists, a `DcmDspRoutineAuthorization` container is created. `DcmDspRoutineSecurityLevelRef` references all the corresponding `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-ROUTINE-CONTROL` references. For more information, see [Section 3.4.11.22, "Determining the access permissions for diagnostic service instances"](#). `DcmDspRoutineSessionRef` references all the associated `DcmDspSession` containers that correspond to the `DIAGNOSTIC-SESSION` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-ROUTINE-CONTROL` references. For more information, see [Section 3.4.11.22, "Determining the access permissions for diagnostic service instances"](#).
`DcmConfigSet/DcmDsp/DcmDspRoutineInfo/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal`	For every request `DIAGNOSTIC-PARAMETER` aggregated by `DIAGNOSTIC-ROUTINE/START` a `DcmDspStartRoutineInSignal` container is created. The container name is set to `DIAGNOSTIC-DATA-ELEMENT/SHORT-NAME` aggregated by `DIAGNOSTIC-PARAMETER` `DcmDspRoutineSignalLength` is set depending on the `ARRAY-SIZE-SEMANTICS` if it is `FIXED-SIZE`. Then the `BASE-TYPE-SIZE` associated with the `DIAGNOSTIC-DATA-ELEMENT` is obtained and set to the value `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` * `BASE-TYPE-SIZE`. If the `ARRAY-SIZE-SEMANTICS` value of the `DIAGNOSTIC-DATA-ELEMENT` is `VARIABLE-SIZE`, the `DcmDspRoutineSignalLength` value is set to `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS`*8. If `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` is missing then its value is considered 0. If `ARRAY-SIZE-SEMANTICS` is not set then the value of `DcmDspRoutineSignalLength` is set to the size given by `BASE-TYPE-SIZE`. For example 8 for UINT8, 16 for UINT16 etc. See also [Section 3.4.11.23, "Determining the type for a routine signal"](#). `DcmDspRoutineSignalPos` is set to `DIAGNOSTIC-PARAMETER/BIT-OFFSET`. `DcmDspRoutineSignalType` is set based on [Section 3.4.11.23, "Determining the type for a routine signal"](#).

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsp/DcmDspRoutineInfo/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal`	For every response `DIAGNOSTIC-PARAMETER` aggregated by `DIAGNOSTIC-ROUTINE/START` a `DcmDspStartRoutineOutSignal` container is created. The container name is set to `DIAGNOSTIC-DATA-ELEMENT/SHORT-NAME` aggregated by `DIAGNOSTIC-PARAMETER` `DcmDspRoutineSignalLength` is set depending on the `ARRAY-SIZE-SEMANTICS` if it is `FIXED-SIZE`. Then the `BASE-TYPE-SIZE` associated with the `DIAGNOSTIC-DATA-ELEMENT` is obtained and set to the value `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` * `BASE-TYPE-SIZE`. If the `ARRAY-SIZE-SEMANTICS` value of the `DIAGNOSTIC-DATA-ELEMENT` is `VARIABLE-SIZE`, the `DcmDspRoutineSignalLength` value is set to `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS`*8. If `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` is missing then its value is considered 0. If `ARRAY-SIZE-SEMANTICS` is not set then the value of `DcmDspRoutineSignalLength` is set to the size given by `BASE-TYPE-SIZE`. For example 8 for UINT8, 16 for UINT16 etc. See also Section 3.4.11.23, "Determining the type for a routine signal". `DcmDspRoutineSignalPos` is set to `DIAGNOSTIC-PARAMETER/BIT-OFFSET`. `DcmDspRoutineSignalType` is set based on Section 3.4.11.23, "Determining the type for a routine signal".
`DcmConfigSet/DcmDsp/DcmDspRoutineInfo/DcmDspRoutineStopIn/DcmDspRoutineStopInSignal`	For every request `DIAGNOSTIC-PARAMETER` aggregated by `DIAGNOSTIC-ROUTINE/STOP` a `DcmDspRoutineStopInSignal` container is created. The container name is set to `DIAGNOSTIC-DATA-ELEMENT/SHORT-NAME` aggregated by `DIAGNOSTIC-PARAMETER` `DcmDspRoutineSignalLength` is set depending on the `ARRAY-SIZE-SEMANTICS` if it is `FIXED-SIZE`. Then the `BASE-TYPE-SIZE` associated with the `DIAGNOSTIC-DATA-ELEMENT` is obtained and set to the value `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` * `BASE-TYPE-SIZE`. If the `ARRAY-SIZE-SEMANTICS` value of the `DIAGNOSTIC-DATA-ELEMENT` is `VARIABLE-SIZE`, the `DcmDspRoutineSignalLength` value is set to `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS`*8. If `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` is missing then its value is considered 0. If `ARRAY-SIZE-SEMANTICS` is not set then the value of `DcmDspRoutineSignalLength` is set to the size given by `BASE-TYPE-SIZE`. For example 8 for UINT8, 16 for UINT16 etc. See also Section 3.4.11.23, "Determining the type for a routine signal".

Configuration parameters	Mapping description
	`DcmDspRoutineSignalPos` is set to `DIAGNOSTIC-PARAMETER/BIT-OFFSET`. `DcmDspRoutineSignalType` is set based on Section 3.4.11.23, "Determining the type for a routine signal".
`DcmConfigSet/DcmDsp/DcmDspRoutineInfo/DcmDspRoutineStopOut/DcmDspRoutineStopOutSignal`	For every response `DIAGNOSTIC-PARAMETER` aggregated by `DIAGNOSTIC-ROUTINE/STOP` a `DcmDspRoutineStopOutSignal` container is created. The container name is set to `DIAGNOSTIC-DATA-ELEMENT/SHORT-NAME` aggregated by `DIAGNOSTIC-PARAMETER` `DcmDspRoutineSignalLength` is set depending on the `ARRAY-SIZE-SEMANTICS` if it is `FIXED-SIZE`. Then the `BASE-TYPE-SIZE` associated with the `DIAGNOSTIC-DATA-ELEMENT` is obtained and set to the value `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS * BASE-TYPE-SIZE`. If the `ARRAY-SIZE-SEMANTICS` value of the `DIAGNOSTIC-DATA-ELEMENT` is `VARIABLE-SIZE`, the `DcmDspRoutineSignalLength` value is set to `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS*8`. If `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` is missing then its value is considered 0. If `ARRAY-SIZE-SEMANTICS` is not set then the value of `DcmDspRoutineSignalLength` is set to the size given by `BASE-TYPE-SIZE`. For example 8 for UINT8, 16 for UINT16 etc. See also Section 3.4.11.23, "Determining the type for a routine signal". `DcmDspRoutineSignalPos` is set to `DIAGNOSTIC-PARAMETER/BIT-OFFSET`. `DcmDspRoutineSignalType` is set based on Section 3.4.11.23, "Determining the type for a routine signal".
`DcmConfigSet/DcmDsp/DcmDspRoutineInfo/DcmDspRoutineRequestResIn/DcmDspRoutineRequestResInSignal`	If the `REQUESTS` tag exists inside the DiagnosticExtract, then for every request `DIAGNOSTIC-PARAMETER` aggregated by `DIAGNOSTIC-ROUTINE/REQUEST-RESULT` a `DcmDspRoutineRequestResInSignal` container is created. The container name is set to `DIAGNOSTIC-DATA-ELEMENT/SHORT-NAME` aggregated by `DIAGNOSTIC-PARAMETER` `DcmDspRoutineSignalLength` is set depending on the `ARRAY-SIZE-SEMANTICS` if it is `FIXED-SIZE`. Then the `BASE-TYPE-SIZE` associated with the `DIAGNOSTIC-DATA-ELEMENT` is obtained and set to the value `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS * BASE-TYPE-SIZE`. If the `ARRAY-SIZE-SEMANTICS` value of the `DIAGNOSTIC-DATA-ELEMENT` is `VARIABLE-SIZE`, the `DcmDspRoutineSignalLength` value is set to `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS*8`. If `DIAG-`

Configuration parameters	Mapping description
	`NOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` is missing then its value is considered 0. If `ARRAY-SIZE-SEMANTICS` is not set then the value of `DcmDspRoutineSignalLength` is set to the size given by `BASE-TYPE-SIZE`. For example 8 for UINT8, 16 for UINT16 etc. See also [Section 3.4.11.23, "Determining the type for a routine signal"](#). `DcmDspRoutineSignalPos` is set to `DIAGNOSTIC-PARAMETER/BIT-OFFSET`. `DcmDspRoutineSignalType` is set based on [Section 3.4.11.23, "Determining the type for a routine signal"](#).
`DcmConfigSet/DcmDsp/DcmDspRoutineInfo/DcmDspRoutineRequestResOut/DcmDspRoutineRequestResOutSignal`	For every response `DIAGNOSTIC-PARAMETER` aggregated by `DIAGNOSTIC-ROUTINE/REQUEST-RESULT` a `DcmDspRoutineRequestResOutSignal` container is created. The container name is set to `DIAGNOSTIC-DATA-ELEMENT/SHORT-NAME` aggregated by `DIAGNOSTIC-PARAMETER` `DcmDspRoutineSignalLength` is set depending on the `ARRAY-SIZE-SEMANTICS` if it is `FIXED-SIZE`. Then the `BASE-TYPE-SIZE` associated with the `DIAGNOSTIC-DATA-ELEMENT` is obtained and set to the value `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` * `BASE-TYPE-SIZE`. If the `ARRAY-SIZE-SEMANTICS` value of the `DIAGNOSTIC-DATA-ELEMENT` is `VARIABLE-SIZE`, the `DcmDspRoutineSignalLength` value is set to `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS`*8. If `DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` is missing then its value is considered 0. If `ARRAY-SIZE-SEMANTICS` is not set then the value of `DcmDspRoutineSignalLength` is set to the size given by `BASE-TYPE-SIZE`. For example 8 for UINT8, 16 for UINT16 etc. See also [Section 3.4.11.23, "Determining the type for a routine signal"](#). `DcmDspRoutineSignalPos` is set to `DIAGNOSTIC-PARAMETER/BIT-OFFSET`. `DcmDspRoutineSignalType` is set based on [Section 3.4.11.23, "Determining the type for a routine signal"](#).
`DcmConfigSet/DcmDsp/DcmDspVehInfo`	For every `DIAGNOSTIC-REQUEST-VEHICLE-INFO` service instance which belongs to the imported `ECU-INSTANCE`, a `DcmDspVehInfo` container is created. The container name is set to `DcmDspVehInfo<suffix>`, where `<suffix>` is an underscore followed by `VEHICLE-INFO-TYPE/ID`.

Configuration parameters	Mapping description
	`DcmDspVehInfoInfoType` is set with the value of `VEHICLE-INFO-TYPE/ID`.
`DcmConfigSet/DcmDsp/DcmDspVehInfo/DcmDspVehInfoData`	The `DcmDspVehInfoData` subcontainer is created for every request `DIAGNOSTIC-PARAMETER` aggregated by `DIAGNOSTIC-REQUEST-VEHICLE-INFO`. The name of `DcmDspVehInfoData` sub container is derived from the short name of the `DIAGNOSTIC-DATA-ELEMENT` aggregated by the `DIAGNOSTIC-PARAMETER`.
	`DcmDspVehInfoDataOrder` parameter will be take the value from `DATA-ELEMENT/bitOffset` that it is bellonged to `DIAGNOSTIC-INFO-TYPE` aggregated by the `DIAGNOSTIC-REQUEST-VEHICLE-INFO`.
	`DcmDspVehInfoDataReadFnc` parameter will take the value based on the `DIAGNOSTIC-SERVICE-SW-MAPPING` that has set the reference to a `MAPPED-BSW-SERVICE-DEPENDENCY`. The `BSW-SERVICE-DEPENDENCY` should have a `ROLE-BASED-BSW-MODULE-ENTRY-ASSIGNMENT` that in turn has attribute role set to `Xxx_GetInfotypeValueData` and points to a `BSW-MODULE-ENTRY`.
	`DcmDspVehInfoDataSize` parameter will take the value from `DATA-ELEMENT/maxNumberOfElements` that it is bellonged to `DIAGNOSTIC-INFO-TYPE` aggregated by the `DIAGNOSTIC-REQUEST-VEHICLE-INFO`.
	`DcmDspVehInfoDataUsePort` parameter will be set to true if the reference `DIAGNOSTIC-SERVICE-SW-MAPPING/mappedSwServiceDepenedecy` exists or will be set to false if the reference `DIAGNOSTIC-SERVICE-SW-MAPPING/mappedBswServiceDependency` exists. The `DIAGNOSTIC-SERVICE-SW-MAPPING` shall have a link with a `DIAGNOSTIC-DATA-ELEMENT` that is aggregated by a `DIAGNOSTIC-REQUEST-VEHICLE-INFO`.
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService`	For every *first* occurrence of a `DIAGNOSTIC-READ-DATA-BY-IDENTIFIER, DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER, DIAGNOSTIC-SESSION-CONTROL, DIAGNOSTIC-SECURITY-ACCESS, DIAGNOSTIC-ROUTINE-CONTROL, DIAGNOSTIC-READ-DTC-INFORMATION, DIAGNOSTIC-ECU-RESET, DIAGNOSTIC-COM-CONTROL, DIAGNOSTIC-DIAGNOSTIC-REQUEST-DOWNLOAD, DIAGNOSTIC-REQUEST-UPLOAD, DIAGNOSTIC-DATA-TRANSFER, DIAGNOSTIC-REQUEST-TRANSFER-EXIT, DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID, DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER, DIAGNOSTIC-IO-CONTROL, DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS, DIAGNOSTIC-READ-MEMORY-BY-ADDRESS, DIAGNOSTIC-REQUEST-CONTROL-OF-ON-BOARD-DEVICE DIAGNOSTIC-CLEAR-DIAGNOSTIC-IN-`

Configuration parameters	Mapping description
	FORMATION, DIAGNOSTIC-REQUEST-POWERTRAIN_FREEZE-FRAME-DATA, DIAGNOSTIC-REQUEST-CURRENT-POWERTRAIN-DIAGNOSTIC-DATA, DIAGNOSTIC-REQUEST-EMISSION-RELATED-DTC-PERMANENT-STATUS, DIAGNOSTIC-CONTROL-DTC-SETTING, DIAGNOSTIC-REQUEST-ON-BOARD-MONITORING-TEST-RESULTS, DIAGNOSTIC-REQUEST-VEHICLE-INFO, DIAGNOSTIC-CLEAR-RESET-EMISSION-RELATED-INFO service instance which belongs to the imported ECU-INSTANCE, a DcmDsdService container is created. The container name is set to one of the following: ▶ DiagnosticReadDataByIdentifier for DIAGNOSTIC-READ-DATA-BY-IDENTIFIER. ▶ DiagnosticWriteDataByIdentifier for DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER. ▶ DiagnosticSessionControl for DIAGNOSTIC-SESSION-CONTROL. ▶ DiagnosticSecurityAccess for DIAGNOSTIC-SECURITY-ACCESS. ▶ DiagnosticRoutineControl for DIAGNOSTIC-ROUTINE-CONTROL. ▶ DiagnosticReadDTCInformation for DIAGNOSTIC-READ-DTC-INFORMATION. ▶ DiagnosticEcuReset for DIAGNOSTIC-ECU-RESET. ▶ DiagnosticComControl for DIAGNOSTIC-COM-CONTROL. ▶ DiagnosticRequestVehicleInfo for DIAGNOSTIC-REQUEST-VEHICLE-INFO. ▶ DiagnosticRequestOnBoardMonitoringTestResults for DIAGNOSTIC-REQUEST-ON-BOARD-MONITORING-TEST-RESULTS. ▶ DiagnosticRequestDownload for DIAGNOSTIC-REQUEST-DOWNLOAD. ▶ DiagnosticRequestUpload for DIAGNOSTIC-REQUEST-UPLOAD. ▶ DiagnosticDataTransfer for DIAGNOSTIC-DATA-TRANSFER. ▶ DiagnosticRequestTransferExit for DIAGNOSTIC-REQUEST-TRANSFER-EXIT. ▶ DiagnosticReadDataByPeriodicIdentifier for DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID. ▶ DiagnosticWriteMemoryByAddress for DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS. ▶ DiagnosticReadMemoryByAddress for DIAGNOSTIC-READ-MEMORY-BY-ADDRESS.

Configuration parameters	Mapping description
	▶ `DiagnosticIOControl` for `DIAGNOSTIC-IO-CONTROL`. ▶ `DiagnosticControlDTCSetting` for `DIAGNOSTIC-CONTROL-DTC-SETTING`. ▶ `DiagnosticDynamicallyDefineDataIdentifier` for `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER`. ▶ `DiagnosticRequestControlOfOnBoardDevice` for `DIAGNOSTIC-REQUEST-CONTROL-OF-ON-BOARD-DEVICE`. ▶ `DiagnosticClearDiagnosticInformation` for `DIAGNOSTIC-CLEAR-DIAGNOSTIC-INFORMATION`. ▶ `DiagnosticRequestPowertrainFreezeFrameData` for `DIAGNOSTIC-REQUEST-POWERTRAIN_FREEZE-FRAME-DATA`. ▶ `DiagnosticRequestCurrentPowertrainDiagnosticData` for `DIAGNOSTIC-REQUEST-CURRENT-POWERTRAIN-DIAGNOSTIC-DATA`. ▶ `DiagnosticRequestEmissionRelatedDTCPermanentStatus` for `DIAGNOSTIC-REQUEST-EMISSION-RELATED-DTC-PERMANENT-STATUS`. ▶ `DiagnosticClearResetEmissionRelatedInfo` for `DIAGNOSTIC-CLEAR-RESET-EMISSION-RELATED-INFO`. For every *first* occurrence of a `DIAGNOSTIC-REQUEST-EMISSION-RELATED-DTC` service instance which belongs to the imported `ECU-INSTANCE`, two `DcmDsdService` containers are created. The containers name are set to the following: ▶ `DiagnosticRequestEmissionRelatedDTC_03` for `DIAGNOSTIC-REQUEST-EMISSION-RELATED-DTC`. ▶ `DiagnosticRequestEmissionRelatedDTC_07` for `DIAGNOSTIC-REQUEST-EMISSION-RELATED-DTC`. From the second occurrence onwards, only the configuration parameters `DcmDsdSidTabSessionLevelRef` and `DcmDsdSidTabSecurityLevelRef` are updated. If `AccessPermissionValidity` is set to `ACCES-PERRMISSION-SERVICE-INSTANCE` then `DcmDsdSidTabSessionLevelRef` and `DcmDsdSidTabSecurityLevelRef` are empty. The `DcmDsdSidTabSessionLevelRef` and `DcmDsdSidTabSecurityLevelRef` are configurable only from data identifiers (DIDs). `DcmDsdSidTabServiceId` is set to one of the following:

Configuration parameters	Mapping description
	▶ `0x10` for `DIAGNOSTIC-SESSION-CONTROL`.
	▶ `0x27` for `DIAGNOSTIC-SECURITY-ACCESS`.
	▶ `0x22` for `DIAGNOSTIC-READ-DATA-BY-IDENTIFIER`.
	▶ `0x2E` for `DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER`.
	▶ `0x31` for `DIAGNOSTIC-ROUTINE-CONTROL`.
	▶ `0x19` for `DIAGNOSTIC-READ-DTC-INFORMATION`.
	▶ `0x11` for `DIAGNOSTIC-ECU-RESET`.
	▶ `0x28` for `DIAGNOSTIC-COM-CONTROL`.
	▶ `0x09` for `DIAGNOSTIC-REQUEST-VEHICLE-INFO`.
	▶ `0x06` for `DIAGNOSTIC-REQUEST-ON-BOARD-MONITORING-TEST-RESULTS`.
	▶ `0x34` for `DIAGNOSTIC-REQUEST-DOWNLOAD`.
	▶ `0x35` for `DIAGNOSTIC-REQUEST-UPLOAD`.
	▶ `0x36` for `DIAGNOSTIC-DATA-TRANSFER`.
	▶ `0x37` for `DIAGNOSTIC-REQUEST-TRANSFER-EXIT`.
	▶ `0x2A` for `DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID`.
	▶ `0x2F` for `DIAGNOSTIC-IO-CONTROL`.
	▶ `0x3D` for `DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS`.
	▶ `0x23` for `DIAGNOSTIC-READ-MEMORY-BY-ADDRESS`.
	▶ `0x85` for `DIAGNOSTIC-CONTROL-DTC-SETTING`.
	▶ `0x2C` for `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER`.
	▶ `0x14` for `DIAGNOSTIC-CLEAR-DIAGNOSTIC-INFORMATION`.
	▶ `0x08` for `DIAGNOSTIC-REQUEST-CONTROL-OF-ON-BOARD-DEVICE`.
	▶ `0x03` for `DIAGNOSTIC-REQUEST-EMISSION-RELATED-DTC`.
	▶ `0x07` for `DIAGNOSTIC-REQUEST-EMISSION-RELATED-DTC`.
	▶ `0x0A` for `DIAGNOSTIC-REQUEST-EMISSION-RELATED-DTC-PERMA-NENT-STATUS`.
	▶ `0x02` for `DIAGNOSTIC-REQUEST-POWERTRAIN_FREEZE-FRAME-DATA`.
	▶ `0x01` for `DIAGNOSTIC-REQUEST-CURRENT-POWERTRAIN-DIAG-NOSTIC-DATA`.
	▶ `0x04` for `DIAGNOSTIC-CLEAR-RESET-EMISSION-RELATED-INFO`.

Configuration parameters	Mapping description
	`DcmDsdSidTabSessionLevelRef` references the following: ▶ `Nothing` for `DIAGNOSTIC-SESSION-CONTROL` diagnostic service instance. ▶ All the `DcmDspSession` containers that correspond to the `DIAGNOSTIC-SESSION` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the diagnostic service instance references. For more information, see also Section 3.4.11.22, "Determining the access permissions for diagnostic service instances". `DcmDsdSidTabSecurityLevelRef` references the following: ▶ `Nothing` for `DIAGNOSTIC-SECURITY-ACCESS`. ▶ All the `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the diagnostic service instance references. For more information, see also Section 3.4.11.22, "Determining the access permissions for diagnostic service instances". **Important:** If Section 3.4.11.22, "Determining the access permissions for diagnostic service instances" yields no valid `DIAGNOSTIC-ACCESS-PERMISSION` or a `DIAGNOSTIC-ACCESS-PERMISSION` that does not reference `DIAGNOSTIC-SECURITY-LEVEL` or `DIAGNOSTIC-SESSION` elements, then the corresponding `DcmDsdSidTabSecurityLevelRef` or `DcmDsdSidTabSessionLevelRef` containers for that `DiagnosticServiceInstance` will be empty. `DcmDsdSidTabSubfuncAvail` is set to one the following: ▶ Not set for `DIAGNOSTIC-SESSION-CONTROL`. ▶ Not set for `DIAGNOSTIC-SECURITY-ACCESS`. ▶ Set to `false` for `DIAGNOSTIC-READ-DATA-BY-IDENTIFIER`. ▶ Set to `false` for `DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER`. ▶ Set to `true` for `DIAGNOSTIC-ROUTINE-CONTROL`. ▶ Set to `true` for `DIAGNOSTIC-READ-DTC-INFORMATION`. ▶ Set to `true` for `DIAGNOSTIC-ECU-RESET`. ▶ Set to `true` for `DIAGNOSTIC-COM-CONTROL`. ▶ Set to `false` for `DIAGNOSTIC-REQUEST-ON-BOARD-MONITORING-TEST-RESULTS`.

Configuration parameters	Mapping description
	▶ Set to `false` for DIAGNOSTIC-REQUEST-VEHICLE-INFO. ▶ Set to `false` for DIAGNOSTIC-REQUEST-DOWNLOAD. ▶ Set to `false` for DIAGNOSTIC-REQUEST-UPLOAD. ▶ Set to `false` for DIAGNOSTIC-DATA-TRANSFER. ▶ Set to `false` for DIAGNOSTIC-REQUEST-TRANSFER-EXIT. ▶ Set to `true` for DIAGNOSTIC-CONTROL-DTC-SETTING. ▶ Set to `false` for DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID. ▶ Set to `true` for DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDEN-TIFIER. ▶ Set to `false` for DIAGNOSTIC-IO-CONTROL. ▶ Set to `false` for DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS. ▶ Set to `false` for DIAGNOSTIC-READ-MEMORY-BY-ADDRESS. ▶ Set to `false` for DIAGNOSTIC-CLEAR-DIAGNOSTIC-INFORMATION. ▶ Set to `false` for DIAGNOSTIC-REQUEST-CONTROL-OF-ON-BOARD-DE-VICE. ▶ Set to `false` for DIAGNOSTIC-REQUEST-EMISSION-RELATED-DTC. ▶ Set to `false` for DIAGNOSTIC-REQUEST-EMISSION-RELATED-DTC-PERMANENT-STATUS. ▶ Set to `false` for DIAGNOSTIC-REQUEST-CURRENT-POWERTRAIN-DIAG-NOSTIC-DATA. ▶ Set to `false` for DIAGNOSTIC-REQUEST-POWERTRAIN_FREEZE-FRAME-DATA. ▶ Set to `false` for DIAGNOSTIC-CLEAR-RESET-EMISSION-RELATED-IN-FO.

3.4.11.1. Configuration of subservices for Diagnostic Session Control

Configuration parameters	Mapping description
DcmCon-figSet/DcmDsd/ DcmDsdServic-eTable/DcmDsdSer-	For every DIAGNOSTIC-SESSION-CONTROL a DcmDsdSubService container is created. The container name is set to DIAGNOSTIC-SESSION/SHORT-NAME of DIAGNOSTIC-SESSION-CONTROL.

Configuration parameters	Mapping description
`vice/DcmDsdSubService`	`DcmDsdSubServiceId` is set to `DIAGNOSTIC-SESSION/ID`. `DcmDsdSubServiceSessionLevelRef` references all the associated `DcmDspSession` containers that correspond to the `DIAGNOSTIC-SESSION` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-SESSION-CONTROL` references. For more information, see also Section 3.4.11.22, "Determining the access permissions for diagnostic service instances". `DcmDsdSubServiceSecurityLevelRef` references all the associated `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-SESSION-CONTROL` references. For more information, see also Section 3.4.11.22, "Determining the access permissions for diagnostic service instances".

3.4.11.2. Configuration of subservices for Diagnostic Security Access

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	For every `DIAGNOSTIC-SECURITY-ACCESS` *two* `DcmDsdSubService` containers are created. The name for the first container is `<name>_requestSeed`, where `<name>` is set to `DIAGNOSTIC-SECURITY-ACCESS/SHORT-NAME`. The name for the second container is `<name>_sendKey`, where `<name>` is set to `DIAGNOSTIC-SECURITY-ACCESS/SHORT-NAME`. `DcmDsdSubServiceId` is set to `DIAGNOSTIC-SECURITY-ACCESS/REQUEST-SEED-ID` for the first `DcmDsdService` container. `DcmDsdSubServiceId` is set to `DIAGNOSTIC-SECURITY-ACCESS/REQUEST-SEED-ID`+1 for the second `DcmDsdService` container. `DcmDsdSubServiceSessionLevelRef` references all the associated `DcmDspSession` containers that correspond to the `DIAGNOSTIC-SESSION` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-SECURITY-ACCESS` references. For more information, see also Section 3.4.11.22, "Determining the access permissions for diagnostic service instances". The `DcmDsdSubServiceSecurityLevelRef` will not be configured.

3.4.11.3. Configuration of subservices for Diagnostic Read Data By Identifier

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-READ-DATA-BY-IDENTIFIER` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.4. Configuration of subservices for Diagnostic Write Data By Identifier

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-WRITE-DATA-BY-IDENTIFIER` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.5. Configuration of subservices for Diagnostic IOControl

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-IO-CONTROL` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.6. Configuration of subservices for Diagnostic Write Memory By Address

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-WRITE-MEMORY-BY-ADDRESS` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.7. Configuration of subservices for Diagnostic Read Memory By Address

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-READ-MEMORY-BY-ADDRESS` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.8. Configuration of subservices for Diagnostic Routine Control

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-ROUTINE-CONTROL` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.9. Configuration of subservices for Diagnostic Read Data By Periodic Identifier

Configuration parameters	Mapping description
DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService	Subservices are not applicable for the `DIAGNOSTIC-READ-DATA-BY-PERIODIC-ID` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.10. Configuration of subservices for Diagnostic Dynamically Define Data Identifier

Configuration parameters	Mapping description
DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService	For every `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER` that has its `Subfunction` set, a `DcmDsdSubService` container is created. The container name is set to `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER/SUBFUNCTIONS`. `DcmDsdSubServiceId` is set according to ISO-14229 to the corresponding value of `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER/SUBFUNCTIONS`. `DcmDsdSubServiceSessionLevelRef` references all the associated `DcmDspSession` containers that correspond to the `DIAGNOSTIC-DYNAMICALLY-DEFINE-DATA-IDENTIFIER` entities with which the `DIAGNOSTIC-ACCESS-PERMISSION` is associated. For more information, see Section 3.4.11.22, "Determining the access permissions for diagnostic service instances". `DcmDsdSubServiceSecurityLevelRef` references all the associated `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities with which the `DIAGNOSTIC-ACCESS-PERMISSION` is associated. For more information, see Section 3.4.11.22, "Determining the access permissions for diagnostic service instances".

3.4.11.11. Configuration of subservices for Diagnostic Clear Diagnostic Information

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-CLEAR-DIAGNOSTIC-INFORMATION` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.12. Configuration of subservices for Diagnostic Read DTC Information

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	For every `DIAGNOSTIC-READ-DTC-INFORMATION` that has its `CATEGORY` set, a `DcmDsdSubService` container is created. The container name is set to `DIAGNOSTIC-READ-DTC-INFORMATION/CATEGORY`. `DcmDsdSubServiceId` is set according to ISO-14229 to the corresponding value of `DIAGNOSTIC-READ-DTC-INFORMATION/CATEGORY`. `DcmDsdSubServiceSessionLevelRef` references all the associated `DcmDspSession` containers that correspond to the `DIAGNOSTIC-READ-DTC-INFORMATION` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated. For more information, see Section 3.4.11.22, "Determining the access permissions for diagnostic service instances". `DcmDsdSubServiceSecurityLevelRef` references all the associated `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-READ-DTC-INFORMATION` references. For more information, see Section 3.4.11.22, "Determining the access permissions for diagnostic service instances".

3.4.11.13. Configuration of diagnostic subservices EcuReset

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	For every `DIAGNOSTIC-ECU-RESET` that has its `CATEGORY` set, a `DcmDsdSubService` container is created. The container name is set to `DIAGNOSTIC-ECU-RESET/CATEGORY`. `DcmDsdSubServiceId` is set according to ISO-14229 to the corresponding value of `DIAGNOSTIC-ECU-RESET/CATEGORY`. In case you want to configure a custom sub service, then set the value of the sub service id based on `ECU-RESET/CUSTOM-SUB-FUNCTION-NUMBER`. `DcmDsdSubServiceSessionLevelRef` references all the associated `DcmDspSession` containers that correspond to the `DIAGNOSTIC-ECU-RESET` entities with which the `DIAGNOSTIC-ACCESS-PERMISSION` is associated. For more information, see Section 3.4.11.22, "Determining the access permissions for diagnostic service instances". `DcmDsdSubServiceSecurityLevelRef` references all the associated `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-ECU-RESET` references. For more information, see Section 3.4.11.22, "Determining the access permissions for diagnostic service instances".

3.4.11.14. Configuration of diagnostic subservices ComControl

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	For every `DIAGNOSTIC-COM-CONTROL` that has its `CATEGORY` set, a `DcmDsdSubService` container is created. The container name is set to `DIAGNOSTIC-COM-CONTROL/CATEGORY`. `DcmDsdSubServiceId` is set according to ISO-14229 to the corresponding value of `DIAGNOSTIC-COM-CONTROL/CATEGORY`. In case you want to configure a custom sub service, then set the value of the sub service id based on `COM-CONTROL/CUSTOM-SUB-FUNCTION-NUMBER`. `DcmDsdSubServiceSessionLevelRef` references all the associated `DcmDspSession` containers that correspond to the `DIAGNOSTIC-COM-CONTROL` entities with which the `DIAGNOSTIC-ACCESS-PERMISSION` is associated. For

Configuration parameters	Mapping description
	more information, see Section 3.4.11.22, "Determining the access permissions for diagnostic service instances". `DcmDsdSubServiceSecurityLevelRef` references all the associated `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities which the `DIAGNOSTIC-ACCESS-PERMISSION` associated with the `DIAGNOSTIC-COM-CONTROL` references. For more information, see Section 3.4.11.22, "Determining the access permissions for diagnostic service instances".

3.4.11.15. Configuration of subservices for Diagnostic RequestOnBoardMonitoringTestResults

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-REQUEST-ON-BOARD-MONITORING-TEST-RESULTS` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.16. Configuration of subservices for Diagnostic RequestVehicleInfo

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-REQUEST-VEHICLE-INFO` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.17. Configuration of subservices for Diagnostic RequestDownload

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-REQUEST-DOWNLOAD` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.18. Configuration of subservices for Diagnostic RequestUpload

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-REQUEST-UPLOAD` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.19. Configuration of subservices for Diagnostic DataTransfer

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-DATA-TRANSFER` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.20. Configuration of subservices for Diagnostic RequestTransferExit

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	Subservices are not applicable for the `DIAGNOSTIC-REQUEST-TRANSFER-EXIT` diagnostic service instance and hence `DcmDsdSubService` containers are not created.

3.4.11.21. Configuration of diagnostic subservices ControlDTCSetting

Configuration parameters	Mapping description
`DcmConfigSet/DcmDsd/DcmDsdServiceTable/DcmDsdService/DcmDsdSubService`	For every `DIAGNOSTIC-CONTROL-DTC-SETTING`, a `DcmDsdSubService` container is created. The container name is set to `DIAGNOSTIC-CONTROL-DTC-SETTING/SHORTNAME`. `DcmDsdSubServiceId` is set to `DIAGNOSTIC-CONTROL-DTC-SETTING/DTC-SETTING-PARAMETER`. `DcmDsdSubServiceSessionLevelRef` references all the associated `DcmDspSession` containers that correspond to the `DIAGNOSTIC-CONTROL-DTC-SETTING` entities with which the `DIAGNOSTIC-ACCESS-PERMISSION` is associated. For more information, see Section 3.4.11.22, "Determining the access permissions for diagnostic service instances". `DcmDsdSubServiceSecurityLevelRef` references all the associated `DcmDspSecurity` containers that correspond to the `DIAGNOSTIC-SECURITY-LEVEL` entities with which the `DIAGNOSTIC-ACCESS-PERMISSION` is associated . For more information, see Section 3.4.11.22, "Determining the access permissions for diagnostic service instances".

3.4.11.22. Determining the access permissions for diagnostic service instances

The access permission of a diagnostic service instance is determined as follows:

► If the `DIAGNOSTIC-SESSION-CONTROL-CLASS/ACCESS-PERMISSION-VALIDITY` of the diagnostic service instance is `ACCESS-PERMISSION-SERVICE-INSTANCE`, the access permission of the diagnostic service instance is used.

▶ If the `DIAGNOSTIC-SESSION-CONTROL-CLASS/ACCESS-PERMISSION-VALIDITY` of the diagnostic service instance is `ACCES-PERRMISSION-SERVICE-CLASS`, the access permission of the diagnostic service instance class is used.

▶ If the `DIAGNOSTIC-SESSION-CONTROL-CLASS/ACCESS-PERMISSION-VALIDITY` of the diagnostic service instance is `ACCESS-PERMISSION-INSTANCE-OVERRIDES-CLASS`, the access permission of the diagnostic service instance is used. If the access permission of the diagnostic service instance is not available, then the access permission of the diagnostic service instance class is used.

3.4.11.23. Determining the type for a routine signal

▶ `UINT8` for `BASE-TYPE-ENCODING NONE` and `BASE-TYPE-SIZE 8`

▶ `UINT16` for `BASE-TYPE-ENCODING NONE` and `BASE-TYPE-SIZE 16`

▶ `UINT32` for `BASE-TYPE-ENCODING NONE` and `BASE-TYPE-SIZE 32`

▶ `SINT8` for `BASE-TYPE-ENCODING 2C` and `BASE-TYPE-SIZE 8`

▶ `SINT16` for `BASE-TYPE-ENCODING 2C` and `BASE-TYPE-SIZE 16`

▶ `SINT32` for `BASE-TYPE-ENCODING 2C` and `BASE-TYPE-SIZE 32`

▶ `BOOLEAN` for `BASE-TYPE-ENCODING BOOLEAN` and `BASE-TYPE-SIZE 1` or `BASE-TYPE-SIZE 8`

▶ `VARIABLE-LENGTH` for `ARRAY-SIZE-SEMANTICS VARIABLE-SIZE` , `BASE-TYPE-ENCODING NONE`, and `BASE-TYPE-SIZE 8`

3.4.11.24. Configuration of `DcmProcessingConditions`

▶ A `DcmModeCondition` is created based on `DiagnosticEnvModeCondition` that is valid. The `DcmModeCondition` has the following parameter set: A `DcmConditionType` that has set the value of the `CompareType` attribute of `DiagnosticEnvModeCondition`, and two references `DcmBswModeRef` and `DcmSwcModeRef`. Only one reference is set at a time depending on the `ModeElement` attribute from the `DiagnosticEnvModeCondition`. The `DcmBswModeRef` has a reference to `DiagnosticEnvBswModeElement` and the `DcmSwcModeRef` has a reference to `DiagnosticEnvSwcModeElement`.

▶ A `DcmModeRule` is created based on the `DiagnosticEnvConditionFormula`. The `DcmModeRule` has the following parameters set: A `DcmLogicalOperator` that has set the value of the `Op` attribute of `DiagnosticEnvConditionFormula`. A `DcmModeRuleNrcValue` that has set the value of the `NrcValues` attribute of `DiagnosticEnvConditionFormula` and a `DcmArgumentRef`. The `DcmArgumentRef` has references to the `parts` of the `DiagnosticEnvConditionFormula`.

3.4.11.25. Configuration of `DcmDspPid`

▶ The `DcmDspPidIdentifier` is created if the id of the `DiagnosticParameterIdentifier` is valid.

▶ The `DcmDspPidService` has the following parameter set: A `DCM_SERVICE_01`, `DCM_SERVICE_-02` if `DiagnosticParameterIdentifier` is referenced by `DiagnosticRequestCurrentPowertrainData` or `DiagnosticRequestPowertrainFreezeFrameData` services,and the value `DCM_-SERVICE_01_02` if it's referenced by both services.

▶ The `DcmDspPidSize` has the `pidSize` parameter of the `DiagnosticParameterIdentifier`if it's valid.

3.4.11.25.1. Configuration of `DcmDspPidData`

▶ The `DcmDspPidDataPos` value is set to `maxNumberOfElements` from the `DiagnosticDataElement` of the `DiagnosticParameterIdentifier`.

▶ The `DcmDspPidDataSize` value is set depending on the `maxNumberOfElements` from the `DiagnosticDataElement` of the `DiagnosticParameterIdentifier`. If it's referenced by a `DiagnosticServiceSwMapping` and has a valid `MappedFlatSwcServiceDependency` mapping that has `RoleBasedPortAssignment` that have a valid `PortPrototype` mapping then, if the `PortPrototype` has `ProvidedInterface` it's considered of type `SenderReceiverInterface` and the value will be calculated like this: `maxNumberOfElements * (baseType.getBaseTypeSize() / 8)` If `ProvidedInterface` is of type `ClientServerInterface` then the value will be the `maxNumberOfElements` of the `DiagnosticDataElement`.

3.4.11.25.2. Configuration of `DcmDspPidDataSupportInfo`

▶ The `DcmDspPidDataSupportInfoBit` value is set from `supportInfoBit` value of the `DiagnosticParameterIdentifier`.

▶ The `DcmDspPidDataSupportInfoRef` value is a reference set to `DcmDspPidSupportInfo` based on the `supportInfoByte` value of the `DiagnosticParameterIdentifier`.

3.4.11.25.3. Configuration of `DcmDspPidService01`

▶ The `DcmDspPidDataReadFnc` value is set based on the `DiagnosticDataElement` if it's referenced in a `DiagnosticServiceSwMapping` and has a valid `MappedBswServiceDependency` mapping with a `BswServiceDepedency` that has a `RoleBasedBswModuleEntryAssignment` then the value will be the `Role` value.

▶ The `DcmDspPidDataUsePort` value is set depending on the `DiagnosticServiceSwMapping`.The values will be the following:`USE_DATA_SYNCH_FNC` if the `DiagnosticServiceSwMapping` has a valid `MappedBswServiceDependency` mapping or `USE_DATA_SYNCH_CLIENT_SERVER` if the `DiagnosticServiceSwMapping` has a valid `MappedFlatSwcServiceDependency` or `SwcServiceDependecy` mapping.

3.4.11.25.4. Configuration of `DcmDspPidSupportInfo`

▶ The `DcmDspPidSupportInfoLen` value is set to `Size` from the parameter `DiagnosticSupportIn-foByte` of the `DiagnosticParameterIdentifier`.

▶ The `DcmDspPidSupportInfoPos` value is set to `Position` from the parameter `DiagnosticSupport-InfoByte` of the `DiagnosticParameterIdentifier`.

3.4.12. Dem

Configuration parameters	Mapping description
DemConfigSet	Check if a `DemConfigSet` container is created inside the module. If one already exists then an additional one is not created, otherwise a new `DemConfigSet` container is created.
DemConfigSet/DemDTCClass	For every `DIAGNOSTIC-TROUBLE-CODE-UDS` that is assigned to the configured `ECU-INSTANCE` one `DemDTCClass` container is created. The container name is `<PREFIX><name>`, where `<name>` is the mangled name of the `DIAGNOSTIC-TROUBLE-CODE-UDS`. `DemUdsDTC` is set to `DIAGNOSTIC-TROUBLE-CODE-UDS/UDS-DTC-VALUE`. `DemDTCFunctionalUnit` is set to `DIAGNOSTIC-TROUBLE-CODE-UDS/FUNCTIONAL-UNIT`. `DemDTCSeverity` is set to `DIAGNOSTIC-TROUBLE-CODE-UDS/SEVERITY`. `DemImmediateNvStorage` is set to `DIAGNOSTIC-TROUBLE-CODE-UDS/DIAGNOSTIC-TROUBLE-CODE-PROPS/IMMEDIATE-NV-DATA-STORAGE`. `DemObdDTC` is set to `DIAGNOSTIC-TROUBLE-CODE-OBD/OBD-DTC-VALUE`.
DemGeneral/DemGroupOfDTC	For every `DIAGNOSTIC-TROUBLE-CODE-GROUP` that is assigned to the configured `ECU-INSTANCE` one `DemGroupOfDTC` container is created. The container name is `<PREFIX><name>`, where `<name>` is the mangled name of the `DIAGNOSTIC-TROUBLE-CODE-GROUP`. `DemGroupDTCs` is set to `DIAGNOSTIC-TROUBLE-CODE-GROUP/GROUP-NUMBER`.
DemGeneral/DemEnableCondition	For every `DIAGNOSTIC-ENABLE-CONDITION` that is assigned to the configured `ECU-INSTANCE` one `DemEnableCondition` container is created. The container name is `<name>`, where `<name>` is the mangled name of the `DIAGNOSTIC-ENABLE-CONDITION`.

Configuration parameters	Mapping description
	`DemEnableConditionStatus` is set to `DIAGNOSTIC-ENABLE-CONDITION/INIT-VALUE`.
`DemGeneral/DemEnableCondition-Group`	For every different `DIAGNOSTIC-ENABLE-CONDITION-GROUP` that contains different sets of `DIAGNOSTIC-ENABLE-CONDITION` references, that is assigned to the configured `ECU-INSTANCE`, a single `DemEnableCondition-Group` container is created. The container name is `<name>`, where `<name>` is the mangled name of the `DIAGNOSTIC-ENABLE-CONDITION-GROUP`. `DemEnableConditionRef` references all the associated `DemEnableCondition` containers that correspond to the `DIAGNOSTIC-ENABLE-CONDITION` elements aggregated within this `DIAGNOSTIC-ENABLE-CONDITION-GROUP`.
`DemGeneral/DemOperationCycle`	A `DemOperationCycle` container is created for every `DIAGNOSTIC-OPERATION-CYCLE` that is assigned to the configured `ECU-INSTANCE` and which is used as either a `DIAGNOSTIC-OPERATION-CYCLE-REF` or as a `HEALING-CYCLE` attribute for a `DIAGNOSTIC-CONNECTED-INDICATOR`. The container name is `<PREFIX><name>`, where `<name>` is the mangled name of the `DIAGNOSTIC-OPERATION-CYCLE`. `DemOperationCycleAutomaticEnd` is set to `DIAGNOSTIC-OPERATION-CYCLE/AUTOMATIC-END`. `DemOperationCycleType` is set to `DIAGNOSTIC-OPERATION-CYCLE/TYPE`.
`DemGeneral/DemAgingCycle`	A `DemAgingCycle` container is created for every `DIAGNOSTIC-OPERATION-CYCLE` that is assigned to the configured `ECU-INSTANCE` and which is used as either a `DIAGNOSTIC-OPERATION-CYCLE-REF` or as `AGING-CYCLE` for a `DIAGNOSTIC-AGING`. The container name is `<PREFIX><name>`, where `<PREFIX>` is `DEM_AGCYCLE_` and `<name>` is the mangled name of the `DIAGNOSTIC-OPERATION-CYCLE`.
`DemGeneral`	Check if a `DemGeneral` container is created inside the module. If a container already exists, no additional container is created. Otherwise, a new `DemGeneral` container is created. `DemEnableConditionSupport` is set to true if at least one `DIAGNOSTIC-ENABLE-CONDITION` exists that is assigned to the configured `ECU-INSTANCE`. `DemTypeOfFreezeFrameRecordNumeration` is set to `DEM_FF_RECNUM_CONFIGURED` if at least one `DIAGNOSTIC-TROUBLE-CODE-PROPS` exists that is assigned to the configured `ECU-INSTANCE` and contains at least one `DiagnosticFreezeFrame` with a valid `RecordNumber` parameter.

Configuration parameters	Mapping description
	`DemDebounceCounterBasedSupport` is set to true if the `DIAG-NOSTIC-EVENT` that is assigned to the configured `ECU-INSTANCE` has support for `DIAG-EVENT-DEBOUNCE-COUNTER-BASED`. `DemDebounceTimeBasedSupport` is set to true if the `DIAGNOSTIC-EVENT` that is assigned to the configured `ECU-INSTANCE` has support for `DIAG-EVENT-DEBOUNCE-TIME-BASED`. If at least one `DIAGNOSTIC-COMMON-ELEMENT` exists that is an instance of `DIAGNOSTIC-ECU-INSTANCE-PROPS` with `DTC-STATUS-AVAILABILITY-MASK` set and is assigned to the configured `ECU-INSTANCE`, then the value of the `DemDtcStatusAvailabilityMask` attribute is configured based on the `DTC-STATUS-AVAILABILITY-MASK` from the first `DIAGNOSTIC-ECU-INSTANCE-PROPS` configured. If at least one `DIAGNOSTIC-COMMON-ELEMENT` exists that is an instance of `DIAGNOSTIC-ECU-INSTANCE-PROPS` with `OBD-SUPPORT` set with a different value then `NO-OBD-SUPPORT` and is assigned to the configured `ECU-INSTANCE`, then the value of the `DemOBDSupport` attribute is set to true and the `DemOBDSupportKind` attribute is configured based on the `OBD-SUPPORT` from the first `DIAGNOSTIC-ECU-INSTANCE-PROPS` configured.Otherwise the `DemOBDSupport` attribute is set to false and the `DemOBDSupportKind` is disabled
`DemGeneral/DemIndicator`	For every `DIAGNOSTIC-INDICATOR` that is assigned to the configured `ECU-INSTANCE` one `DemIndicator` container is created. The container name is `<PREFIX><name>`, where `<name>` is the mangled name of the `DIAGNOSTIC-INDICATOR`. `DemIndicatorID` is set to an auto-calculated value which specifies the index of this container.
`DemGeneral/DemClient`	If any `DiagnosticServiceInstance` of a `DiagnosticServiceTable` of the configured `EcuInstance` depends on `Dem`, one or two `DemClient` containers are created. The container names are `DemClient<ProtocolType>`, where `<ProtocolType>` is `OBD` and/or `UDS`.
`DemGeneral/DemDataElementClass`	A DemDataElementClass choice container is created for all the DiagnosticDataElements aggregated by a DiagnosticTroubleCodeProps,either through the DiagnosticExtendedDataRecords or through the FreezeFrameContent For each of the DiagnosticDataElements linked inside a DiagnosticDemProvidedDataMapping element assigned to the configured EcuInstance a DemInternalDataElementClass choice container is created. The container name is

Configuration parameters	Mapping description
	`<PREFIX><name>`,where `<PREFIX>`is DemInternalDataElementClass_ and `<name>` is the mangled shortName of the DiagnosticDataElement. `DemInternalDataElement` is set to the `DIAGNOSTIC-DATA-PROVIDER` category only if the values of DataProvider are reserved by the AUTOSAR standard and are one of the following: ► DEM_AGINGCTR_DOWNCNT ► DEM_AGINGCTR_UPCNT ► DEM_CURRENT_FDC ► DEM_CYCLES_SINCE_FIRST_FAILED ► DEM_CYCLES_SINCE_LAST_FAILED ► DEM_OCCCTR ► DEM_OVFLIND ► DEM_SIGNIFICANCE If `DIAGNOSTIC-DATA-PROVIDER` has a different value than the ones reserved by the AUTOSAR standard, then `DemInternalDataElement` is set to the default value DEM_AGINGCTR and a warning is issued. `DemDataElementDataSize` is set to (`DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` * `DIAGNOSTIC-DATA-ELEMENT/SW-DATA-DEF-PROPS/SW-DATA-DEF-PROPS-CONDITIONAL/SW-BASE-TYPE/BASE-TYPE-SIZE`)/8. By default if the DiagnosticDataElement is not linked inside a DiagnosticDemProvidedDataMapping then a DemExternalCSDataElementClass choice container is created. The container name is `<PREFIX><name>`,where `<PREFIX>`is DemExternalCSDataElementClass_ and `<name>` is the mangled shortName of the DiagnosticDataElement. `DemDataElementDataSize` is set to (`DIAGNOSTIC-DATA-ELEMENT/MAX-NUMBER-OF-ELEMENTS` * `DIAGNOSTIC-DATA-ELEMENT/SW-DATA-DEF-PROPS/SW-DATA-DEF-PROPS-CONDITIONAL/SW-BASE-TYPE/BASE-TYPE-SIZE`)/8.
`DemGeneral/DemDem-DidClass`	For every `DIAGNOSTIC-DATA-IDENTIFIER` that is assigned to the configured `ECU-INSTANCE` one `DemDemDidClass` container is created. The container name is `<PREFIX><name>`, where `<name>` is the mangled name of the `DIAGNOSTIC-DATA-IDENTIFIER` and the `<PREFIX>` is `DemDidClass_`.

Configuration parameters	Mapping description
	`DemDidIdentifier` is set to `DIAGNOSTIC-DATA-IDENTIFIER/ID`. `DemDidDataElementClassRef` references all the associated `DemDataElementClass` containers that correspond to the `DIAGNOSTIC-DATA-ELEMENT` elements aggregated within this `DIAGNOSTIC-DATA-IDENTIFIER`.
`DemGeneral/Dem-FreezeFrameClass`	For every different set of `DIAGNOSTIC-DATA-IDENTIFIER` elements from `DIAGNOSTIC-DATA-IDENTIFIER-SET` aggregated by each `DIAGNOSTIC-TROUBLE-CODE-PROPS` that is assigned to the configured `ECU-INSTANCE`, only one `DemFreezeFrameClass` container is created. The container name is `<name><SUFFIX>`, where `<name>` is `DiagnosticFreezeFrameClass_` and the `<SUFFIX>` is the number of the `DemFreezeFrameClass` containers created. `DemDidClassRef` references all the associated DiagnosticDataIdentifiers elements aggregated within a `DIAGNOSTIC-DATA-IDENTIFIER-SET`.
`DemGeneral/Dem-FreezeFrameRecNum-Class`	For every different set of `DIAGNOSTIC-FREEZE-FRAME-RECORD-NUMBER` elements aggregated by each `DIAGNOSTIC-TROUBLE-CODE-PROPS` that is assigned to the configured `ECU-INSTANCE` only one `DemFreezeFrameRecNum-Class` container is created. The container name is `<name><SUFFIX>`, where `<name>` is `DiagnosticFreezeFrameRecNumClass_` and the `<SUFFIX>` is the number of the `DemFreezeFrameRecNumClass` containers created. `DemFreezeFrameRecordNumber` is set to the `DIAGNOSTIC-FREEZE-FRAME-RECORD-NUMBER` aggregated by a `DIAGNOSTIC-TROUBLE-CODE-PROPS`.
`DemConfigSet/De-mEventParameter`	For each `DIAGNOSTIC-EVENT` that is assigned to the configured `ECU-INSTANCE`, one `DemEventParameter` container is created. The container name is `<PREFIX><name>`, where `<name>` is the mangled name of the `DIAGNOSTIC-EVENT`. `DemEventId` is set to an auto-calculated value which specified the index of this container. `DemEventKind` is set to `DIAGNOSTIC-EVENT/EVENT-KIND`. `DemDTCClassRef` references the associated `DemDTCClass` container that corresponds to the `DIAGNOSTIC-TROUBLE-CODE-UDS` which is mapped to the `DIAGNOSTIC-EVENT` via `DIAGNOSTIC-EVENT-TO-TROUBLE-CODE-UDS-MAPPING`. If a `DemExtendedDataClass` container has been created for the `DIAGNOSTIC-TROUBLE-CODE-PROPS` element of this `DIAGNOSTIC-EVENT`, De-

Configuration parameters	Mapping description
	`mExtendedDataClassRef` is configured to refer to that `DemExtendedData-Class` container. `DemFreezeFrameClassRef` references the associated `DemFreezeFrame-Class` container that corresponds to the `DIAGNOSTIC-TROUBLE-CODE-PROPS/FREEZE-FRAME-CONTENT-REF`, where `DIAGNOSTIC-TROUBLE-CODE-PROPS` can be obtained via `DIAGNOSTIC-EVENT-TO-TROUBLE-CODE-UDS-MAPPING`. `DemFreezeFrameRecNumClassRef` references the associated `DemDTCClass` container that corresponds to the `DIAGNOSTIC-TROUBLE-CODE-PROPS/DIAGNOSTIC-EXTENDED-DATA-RECORD-REF-CONDITIONAL`, where `DIAGNOSTIC-TROUBLE-CODE-PROPS` can be obtained via `DIAGNOSTIC-EVENT-TO-TROUBLE-CODE-UDS-MAPPING`. `DemMaxNumberFreezeFrameRecords` is set to the `DIAGNOSTIC-TROUBLE-CODE-PROPS/MAX-NUMBER-FREEZE-FRAME-RECORDS`, where `DIAGNOSTIC-TROUBLE-CODE-PROPS` can be obtained via `DIAGNOSTIC-EVENT-TO-TROUBLE-CODE-UDS-MAPPING`.
`DemConfigSet/De-mEventParame-ter/DemCallback-ClearEventAllowed`	A `DemCallbackClearEventAllowed` container is created under `DemEvent-Parameter` if `DIAGNOSTIC-EVENT/EVENT-CLEAR-ALLOWED` is set. The `<name>` is set to `DemCallbackClearEventAllowed`.
`DemConfigSet/De-mEventParameter/De-mEventClass`	A `DemEventClass` container is created under `DemEventParameter`. The `<name>` is set to `DemEventClass`. `DemEventSignificance` is set to the `DIAGNOSTIC-TROUBLE-CODE-UDS/DIAGNOSTIC-TROUBLE-CODE-PROPS/SIGNIFICANCE` linked to the `Diag-nosticEvent` if the `DIAGNOSTIC-TROUBLE-CODE-PROPS/SIGNIFICANCE` is valid. Otherwise, `DemEventSignificance` is set to the `DIAGNOSTIC-TROU-BLE-CODE-OBD/DIAGNOSTIC-TROUBLE-CODE-PROPS/SIGNIFICANCE` linked to the `DiagnosticEvent` via the `DIAGNOSTIC-TROUBLE-CODE-UDS` if the `DIAGNOSTIC-TROUBLE-CODE-PROPS/SIGNIFICANCE` is valid. `DemEventDestination` is set to the first `MEMORY-DESTINATION-REF` from the `DIAGNOSTIC-TROUBLE-CODE-UDS/DIAGNOSTIC-TROUBLE-CODE-PROPS/MEMORY-DESTINATION-REFS` linked to the `DiagnosticEvent` if the `DIAGNOSTIC-TROUBLE-CODE-PROPS/MEMORY-DESTINATION-REF` is valid. Otherwise, `DemEventDestination` is set to the first `MEMO-RY-DESTINATION-REF` from the `DIAGNOSTIC-TROUBLE-CODE-OBD/DIAG-NOSTIC-TROUBLE-CODE-PROPS/MEMORY-DESTINATION-REFS` linked to the

Configuration parameters	Mapping description
	`DiagnosticEvent` via the `DIAGNOSTIC-TROUBLE-CODE-UDS` if the `DIAGNOSTIC-TROUBLE-CODE-PROPS/MEMORY-DESTINATION-REF` is valid. `DemFFPrestorageSupported` is set to `DIAGNOSTIC-EVENT/PRESTORAGE-FREEZE-FRAME`. `DemEventPriority` is set to `DIAGNOSTIC-TROUBLE-CODE-PROPS/PRIORITY`. `DemAgingAllowed` is set to `DIAGNOSTIC-EVENT/AGING-ALLOWED`. `DemEventFailureCycleCounterThreshold` is set to `DIAGNOSTIC-EVENT/EVENT-FAILURE-CYCLE-COUNTER-THRESHOLD`. `DemEnableConditionGroupRef` references the associated `DemEnableConditionGroup` container that corresponds to the `DIAGNOSTIC-ENABLE-CONDITION-GROUP` which is mapped to the `DIAGNOSTIC-EVENT` via `DIAGNOSTIC-EVENT-TO-ENABLE-CONDITION-GROUP-MAPPING`. `DemOperationCycleRef` references the associated `DemOperationCycle` container that corresponds to the `DIAGNOSTIC-OPERATION-CYCLE` which is mapped to the `DIAGNOSTIC-EVENT` via `DIAGNOSTIC-EVENT-TO-OPERATION-CYCLE-MAPPING`. `DemAgingCycleRef` references the associated `DemAgingCycle` container that corresponds to the `DIAGNOSTIC-AGING` which is mapped to the `DIAGNOSTIC-EVENT` via `DIAGNOSTIC-EVENT-TO-TROUBLE-CODE-UDS-MAPPING`. `DemAgingCycleCounterThreshold` is set to `DIAGNOSTIC-AGING/THRESHOLD` which is mapped to `DemAgingCycle` referred above. `DemConsiderPtoStatus` is set to `DIAGNOSTIC-TROUBLE-CODE-OBD/CONSIDER-PTO-STATUS`. `DemEventOBDReadinessGroup` is set to `DIAGNOSTIC-TROUBLE-CODE-OBD/EVENT-OBD-READINESS-GROUP`.
`DemConfigSet/DemEventParameter/DemEventClass/DemIndicatorAttribute`	The `DemIndicatorAttribute` container is created for every `DIAGNOSTIC-CONNECTED-INDICATOR` aggregated by the `DIAGNOSTIC-EVENT`. The container name is `<PREFIX><name>`, where `<name>` is the mangled name of the `DIAGNOSTIC-CONNECTED-INDICATOR`.

Configuration parameters	Mapping description
	`DemIndicatorBehaviour` **is set to** `DIAGNOSTIC-CONNECTED-INDICATOR/BEHAVIOR`. `DemIndicatorHealingCycleRef` **references the associated** `DemOperationCycle` **container that corresponds to** `DIAGNOSTIC-CONNECTED-INDICATOR/HEALING-CYCLE-REF`. `DemIndicatorRef` **references the associated** `DemIndicator` **container that corresponds to** `DIAGNOSTIC-CONNECTED-INDICATOR/INDICATOR-REF`. `DemIndicatorHealingCycleCounterThreshold` **is set to** `DIAGNOSTIC-CONNECTED-INDICATOR/INDICATOR-REF/HEALING-CYCLE-COUNTER-THRESHOLD`.
`DemConfigSet/DemEventParameter/DemEventClass/DemDebounceAlgorithmClass`	A `DemDebounceAlgorithmClass` **container is created for every** `DemEventClass` **based on the** `DiagnosticDebounceAlgorithmProps` **mapped to the** `DiagnosticEvent` **via** `DiagnosticEventToDebounceAlgorithmMapping`. **The** `DemDebounceAlgorithmClass` **can be one of the following types:** ► `DemDebounceMonitorInternal` ► `DemDebounceCounterBased` ► `DemDebounceTimeBase`
`DemConfigSet/DemEventParameter/DemEventClass/DemDebounceAlgorithmClass/DemDebounceTimeBase`	`DemDebounceTimeFailedThreshold` **is set to** `DIAGNOSTIC-DEBOUNCE-ALGORITHM-PROPS/DIAG-EVENT-DEBOUNCE-TIME-BASED/TIME-FAILED-THRESHOLD`. `DemDebounceTimePassedThreshold` **is set to** `DIAGNOSTIC-DEBOUNCE-ALGORITHM-PROPS/DIAG-EVENT-DEBOUNCE-TIME-BASED/TIME-PASSED-THRESHOLD`.
`DemConfigSet/DemEventParameter/DemEventClass/DemDebounceAlgorithmClass/DiagEventDebounceCounterBased`	`DemDebounceCounterDecrementStepSize` **is set to** `DIAGNOSTIC-DEBOUNCE-ALGORITHM-PROPS/DIAG-EVENT-DEBOUNCE-COUNTER-BASED/COUNTER-DECREMENT-STEP-SIZE`. `DemDebounceCounterFailedThreshold` **is set to** `DIAGNOSTIC-DEBOUNCE-ALGORITHM-PROPS/DIAG-EVENT-DEBOUNCE-COUNTER-BASED/COUNTER-FAILED-THRESHOLD`. `DemDebounceCounterIncrementStepSize` **is set to** `DIAGNOSTIC-DEBOUNCE-ALGORITHM-PROPS/DIAG-EVENT-DEBOUNCE-COUNTER-BASED/COUNTER-INCREMENT-STEP-SIZE`.

Configuration parameters	Mapping description
	`DemDebounceCounterJumpDown` is set to `DIAGNOSTIC-DEBOUNCE-ALGO-RITHM-PROPS/DIAG-EVENT-DEBOUNCE-COUNTER-BASED/COUNTER-JUMP-DOWN`. `DemDebounceCounterJumpDownValue` is set to `DIAGNOSTIC-DE-BOUNCE-ALGORITHM-PROPS/DIAG-EVENT-DEBOUNCE-COUNTER-BASED/COUNTER-JUMP-DOWN-VALUE`. `DemDebounceCounterJumpUp` is set to `DIAGNOSTIC-DEBOUNCE-ALGO-RITHM-PROPS/DIAG-EVENT-DEBOUNCE-COUNTER-BASED/COUNTER-JUMP-UP`. `DemDebounceCounterJumpUpValue` is set to `DIAGNOSTIC-DEBOUNCE-AL-GORITHM-PROPS/DIAG-EVENT-DEBOUNCE-COUNTER-BASED/COUNTER-JUMP-UP-VALUE`. `DemDebounceCounterPassedThreshold` is set to `DIAGNOSTIC-DE-BOUNCE-ALGORITHM-PROPS/DIAG-EVENT-DEBOUNCE-COUNTER-BASED/COUNTER-PASSED-THRESHOLD`.
`DemGeneral/DemExtendedDataClass`	For every distinct and non-empty set of `DIAGNOSTIC-EXTENDED-DA-TA-RECORD` elements that is referenced by any `DIAGNOSTIC-TROUBLE-CODE-PROPS` element of the configured `ECU-INSTANCE`, a single `DemExtendedDataClass` container is created. The container name is `<PREFIX><name>`, where `<name>` is the mangled name of one of the `DIAGNOSTIC-TROUBLE-CODE-PROPS` elements referencing that list. `DemExtendedDataRecordClassRef` references all the associated `DemExtendedDataRecordClass` containers that correspond to the `DIAGNOSTIC-EXTENDED-DATA-RECORD` aggregated within the `DIAG-NOSTIC-TROUBLE-CODE-PROPS`.
`DemGeneral/DemExtendedDataRecord-Class`	For every `DIAGNOSTIC-EXTENDED-DATA-RECORD` that is assigned to the configured `ECU-INSTANCE` one `DemExtendedDataRecordClass` container is created. The container `<name>` is the mangled name of the `DIAGNOSTIC-EX-TENDED-DATA-RECORD`. `DemExtendedDataRecordNumber` is set to `DIAGNOSTIC-EXTENDED-DA-TA-RECORD/RECORD-NUMBER`. `DemExtendedDataRecordUpdate` is set to `DIAGNOSTIC-EXTENDED-DA-TA-RECORD/UPDATE`.

Configuration parameters	Mapping description
	`DemDataElementClassRef` references all the associated `DemDataElement-Class` containers that correspond to the `DIAGNOSTIC-DATA-ELEMENT` elements aggregated within this `DIAGNOSTIC-EXTENDED-DATA-RECORD`.

3.4.13. DoIP

Configuration parameters	Mapping description
`DoIPConfigSet`	One `DoIPConfigSet` container and one `DoIPConnections` subcontainer are created if the imported `ECU-INSTANCE` sends and/or receives one or more PDUs that belong to a `DO-IP-TP-CONNECTION`. A PDU belongs to a `DO-IP-TP-CONNECTION` if the `DO-IP-TP-CONNECTION` references the PDU via `TP-SDU-REF`.

The *VLAN* of a `DO-IP-TP-CONNECTION` is the `PHYSICAL-CHANNEL` of the `PDU-TRIGGERING` that the `DO-IP-TP-CONNECTION` references via `TP-SDU-REF`.

If the `ECU-INSTANCE` sends a PDU via a `DO-IP-TP-CONNECTION`, the *local diagnosis address* and the *tester address* of the `DO-IP-TP-CONNECTION` are the `ADDRESS` values of the `DO-IP-LOGIC-ADDRESS` elements that the `DO-IP-TP-CONNECTION` references via `DO-IP-SOURCE-ADDRESS`, respectively via `DO-IP-TARGET-ADDRESS`.

If the `ECU-INSTANCE` receives a PDU via a `DO-IP-TP-CONNECTION`, the *local diagnosis address* and the *tester address* of the `DO-IP-TP-CONNECTION` are the `ADDRESS` values of the `DO-IP-LOGIC-ADDRESS` elements that the `DO-IP-TP-CONNECTION` references via `DO-IP-TARGET-ADDRESS`, respectively via `DO-IP-SOURCE-ADDRESS`.

A *DoIP channel* consists of the set of all `DO-IP-TP-CONNECTION` elements via which the `ECU-INSTANCE`:

▸ sends `DCM-I-PDU` elements that have their `DIAG-PDU-TYPE` either set to `DIAG-RESPONSE` or not set at all;

▸ receives `DCM-I-PDU` elements that have their `DIAG-PDU-TYPE` either set to `DIAG-REQUEST` or not set at all. |

Configuration parameters	Mapping description
	Moreover, the `DO-IP-TP-CONNECTION` elements have to share the same *VLAN*, *local diagnosis address*, and *tester address* in order to belong to the same *DoIP channel*.
`DoIPConfigSet/DoIPConnections/DoIPTcpConnection`, `DoIPConfigSet/DoIPConnections/DoIPUdpVehicleAnnouncement`, `DoIPConfigSet/DoIPConnections/DoIPUdpConnection`	If the imported `ECU-INSTANCE` sends and/or receives one or more PDUs of type `GENERAL-PURPOSE-PDU` with `CATEGORY` set to `DoIP` via one or several *local* `SOCKET-ADDRESS` elements, one or more subcontainers are added to `DoIPConfigSet/DoIPConnections`. If the PDUs are sent and/or received via Tcp `SOCKET-ADDRESS` elements, one `DoIPTcpConnection` container is created. For more information about `SOCKET-ADDRESS`, see [Section 3.4.43, "SoAd"](). The container name is `<chn>_<remote_addr>Tcp`, where `<chn>` is the mangled name of the `ETHERNET-PHYSICAL-CHANNEL` the PDUs are transmitted on and `<remote_addr>` is the mangled name of the *remote* `SOCKET-ADDRESS`. If there are two or more *local* Tcp `SOCKET-ADDRESS` elements that exchange `DoIP` PDUs using the same *remote* `SOCKET-ADDRESS`, multiple `DoIPTcpConnection` containers are created. A suffix is appended to the container names of the second, third, etc. of these containers to ensure unique container names.

If there is only one PDU sent to the IPv4-limited broadcast address (255.255.255.255) or to the IPv6-limited broadcast address (FF02:0:0:0:0:0:0:1) via one *local* Udp `SOCKET-ADDRESS`, a `DoIPUdpVehicleAnnouncement` container is created. The container name is `<chn>_<remote_addr>UdpVehicleAnnouncement`.

If there are PDUs sent and/or received via a *local* Udp `SOCKET-ADDRESS`, the PDUs are grouped according to the *remote* `SOCKET-ADDRESS` they are sent to or received from. If there are more than two PDUs for a given remote *remote* address, these PDUs are further grouped according to the *local* `SOCKET-ADDRESS` they belong to. For each of the resulting groups, one `DoIPUdpConnection` container is created. The container name is `<chn>_<remote_addr>Udp`. A suffix is appended to the container names of the second, third, etc. of these containers to ensure unique container names. |
| `DoIPConfigSet/DoIPConnections/DoIPTcpConnection/DoIPSoAdRxPdu`, `DoIPConfigSet/DoIPConnections/DoIPUdpConnection/DoIPSoAdRxPdu` | If received PDUs exist for which the container `DoIPTcpConnection` or `DoIPUdpConnection` has been created, one `DoIPSoAdRxPdu` subcontainer is added. `DoIPSoAdRxPduRef` references the representation of the first received PDU in the `EcuC` module. A warning is issued if multiple received PDUs were found for the `DoIPTcpConnection/DoIPUdpConnection` container. |

Configuration parameters	Mapping description
`DoIPConfigSet/DoIPConnections/DoIPTcpConnection/DoIPSoAdTxPdu`, `DoIPConfigSet/DoIPConnections/DoIPUdpConnection/DoIPSoAdTxPdu`	If sent PDUs exist for which the container `DoIPTcpConnection` or `DoIPUdpConnection` has been created, one `DoIPSoAdTxPdu` subcontainer is added. `DoIPSoAdTxPduRef` references the representation of the first sent PDU in the `EcuC` module. A warning is issued if multiple sent PDUs were found for the `DoIPTcpConnection/DoIPUdpConnection` container.
`DoIPConfigSet/DoIPConnections/DoIPUdpVehicleAnnouncement/DoIPSoAdTxPdu`	If sent PDUs exist for which the container `DoIPUdpVehicleAnnouncement` has been created, one `DoIPSoAdTxPdu` sub container is added. `DoIPSoAdTxPduRef` references the representation of the first sent PDU in the `EcuC` module. A warning is issued if multiple sent PDUs were found for the `DoIPUdpVehicleAnnouncement` container.
`DoIPConfigSet/DoIPConnections/DoIPTargetAddress`	One `DoIPTargetAddress` container is created for each valid and unique *local diagnosis address* value of any `DO-IP-TP-CONNECTION` of the `ECU-INSTANCE`. The container name is `DoIPTargetAddress_<address>`, where `<address>` is the value of the *local diagnosis address*. This value is also configured in `DoIPTargetAddressValue`.
`DoIPConfigSet/DoIPTester`	One `DoIPTester` container is created for each valid and unique *tester address* value. The container name is `DoIPTester_<address>`, where `<address>` is the value of the *tester address*. This value is also configured in `DoIPTesterSA`.
`DoIPConfigSet/DoIPChannel`	For each *DoIP channel*, one `DoIPChannel` container is created. If the *DoIP channel* contains one or more `DO-IP-TP-CONNECTION` elements that receive a PDU, the container name is the mangled name of the first of these elements, otherwise the name is the mangled name of the first contained `DO-IP-TP-CONNECTION` that sends a PDU. `DoIPChannelSARef` is set up to reference the `DoIPConfigSet/DoIPTester` container that has been created for the common *tester address* of the `DO-IP-TP-CONNECTION` elements of the *DoIP channel*. `DoIPChannelTARef` is set up to reference the `DoIPConfigSet/DoIPConnections/DoIPTargetAddress` container that has been created for the common *local diagnosis address* of the `DO-IP-TP-CONNECTION` elements of the *DoIP channel*.
`DoIPConfigSet/DoIPChannel/DoIPPduRRxPdu`	If one or more of the `DO-IP-TP-CONNECTION` elements of the *DoIP channel* receive a PDU, one `DoIPPduRRxPdu` subcontainer is created. Its `DoIPPduRRxPduRef` references the representation of the first of these received PDUs in the

Configuration parameters	Mapping description
	`EcuC` module. A warning is issued if none of the `DO-IP-TP-CONNECTION` elements of the *DoIP channel* receives a PDU.
`DoIPConfigSet/DoIPChannel/DoIPPduRTxPdu`	If one or more of the `DO-IP-TP-CONNECTION` elements of the *DoIP channel* send a PDU, one `DoIPPduRTxPdu` subcontainer is created. Its `DoIPPduRTxPduRef` references the representation of the first of these sent PDUs in the `EcuC` module.

3.4.14. EcuC

Configuration parameters	Mapping description
`EcucPduCollection/Pdu`	For every PDU instance (see Section 3.3.1.3, "Instance handling") sent, received or routed (see Section 3.3.6, "PDU routing") by the imported `ECU-INSTANCE`, a `Pdu` container is created. This also includes:

For every PDU instance (see Section 3.3.1.3, "Instance handling") sent, received or routed (see Section 3.3.6, "PDU routing") by the imported `ECU-INSTANCE`, a `Pdu` container is created. This also includes:

► PDU instances that represent PDUs referenced in `DYNAMIC-PARTS` or `STATIC-PARTS` of `MULTIPLEXED-I-PDU` elements.

► PDU instances that represent PDUs that are transmitted within `CONTAINER-I-PDU` elements. For information about `CONTAINER-I-PDU`, see Section 3.4.29, "IpduM").

The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU belonging to the PDU instance.

For every PDU referenced by a `TP-CONNECTION` belonging to the imported `ECU-INSTANCE`, a `Pdu` container is created as well. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the PDU.

If the `ECU-INSTANCE` receives multiple `NM-PDU` elements in one `CAN-CLUSTER` or `J-1939-CLUSTER`, only one `Pdu` container is created representing the PDU with the lowest CAN-ID.

For an `NM-PDU` an `NmUserDataPdu` container is created if at least one of the following conditions is met:

► The `NM-PDU` contains an `I-SIGNAL-TO-I-PDU-MAPPING` that references an `I-SIGNAL` or an `I-SIGNAL-GROUP`.

► The `NM-PDU` is sent and belongs to a *PNC-enabled* `NM-CLUSTER`.

An `NM-CLUSTER` is considered PNC-enabled in one of the following cases:

Configuration parameters	Mapping description
	▶ The NM-CLUSTER has its NM-PNC-PARTICIPATION not set to false ▶ A PNC-MAPPING references an I-SIGNAL-I-PDU-GROUP which contains PDUs - either directly or via one of its subordinate I-SIGNAL-I-PDU-GROUP elements - that belong to the COMMUNICATION-CLUSTER referenced by the NM-CLUSTER. The referenced I-SIGNAL-I-PDU-GROUP or any of its subordinate I-SIGNAL-I-PDU-GROUP elements must also be referenced by ASSOCIATED-COM-I-PDU-GROUP-REFS of the imported ECU-INSTANCE ▶ A PNC-MAPPING references the ETHERNET-PHYSICAL-CHANNEL which is referenced from the NM-CLUSTER as well ▶ A PNC-MAPPING references a CAN-PHYSICAL-CHANNEL or a FLEXRAY-PHYSICAL-CHANNEL which is contained in a CAN-CLUSTER or a FLEXRAY-CLUSTER which is in turn referenced by the NM-CLUSTER The container name of the NmUserDataPdu container is `<PREFIX><name>_-NmComUserData<INSTSUFFIX>`. If any PNC-enabled NM-CLUSTER is a FLEXRAY-NM-CLUSTER, an additional *EIRA* (external and internal request array) Pdu container named FrNmPnEiraRxNSdu is created. If any of the PNC-enabled NM-CLUSTER elements is a CAN-NM-CLUSTER, an additional Pdu container named CanNmPnEiraRxNSdu is created. If any of the PNC-enabled NM-CLUSTER elements is a UDP-NM-CLUSTER, an additional Pdu container named UdpNmPnEiraRxNSdu is created. If the following conditions are met, one additional *ERA* (external request array) Pdu container is created for each COMMUNICATION-CONNECTOR of the ECU-INSTANCE: ▶ PNC-GATEWAY-TYPE of the COMMUNICATION-CONNECTOR is set to ACTIVE or to PASSIVE. ▶ COMMUNICATION-CONNECTOR is connected to an ETHERNET-PHYSICAL-CHANNEL, a CAN-CLUSTER, or a FLEXRAY-CLUSTER, which in turn is related to a PNC-enabled NM-CLUSTER. The name of the Pdu container is `<PREFIX><channel>NmPnEraRxNSdu`, where `<channel>` is the SHORT-NAME of the FLEXRAY-CLUSTER, CAN-CLUSTER, or ETHERNET-PHYSICAL-CHANNEL. N-PDU elements of LIN-FRAME-TRIGGERING elements which have their IDENTIFIER set to 60 or 61 are not imported.

Configuration parameters	Mapping description
	If a sent `N-PDU` is referenced by two `CAN-TP-CONNECTION` elements a second `Pdu` container is created. The Tx `CAN-TP-CONNECTION` must reference this `N-PDU` via `DATA-PDU-REF`, the Rx `CAN-TP-CONNECTION` via `FLOW-CON-TROL-PDU-REF`. The direction of the `CAN-TP-CONNECTION` is determined by the PDU referenced by `TP-SDU-REF`. The container name of the second `Pdu` container is `<PREFIX><name>_D<INSTSUFFIX>`. `PduLength` is calculated according to Section 3.3.7, "PDU length calculation", except for the following cases: For an `NmUserDataPdu` container that is created for an `NM-PDU`, `PduLength` is calculated depending on the type of `NM-CLUSTER` to which the `NM-PDU` belongs. First the length of the non-user data is calculated: ▶ `CAN-NM-CLUSTER`, `UDP-NM-CLUSTER`: `max(NM-CBV-POSITION, NM-NID-POSITION) + 1`. If one the parameters is not defined or greater than `1`, it is assumed to be `-1` in the formula above. ▶ `FLEXRAY-NM-CLUSTER`: `2`, if `NM-NODE-ID-ENABLED` of the `NM-ECU` to which the `NM-PDU` belongs is set to `true`, `1` otherwise. `PduLength` = (the length of the PDU according to Section 3.3.7, "PDU length calculation") - `<length of the non-user data>`. If the calculated `PduLength` is zero or less, a warning is issued and `PduLength` is not set. For `Pdu` containers created for PNC-enabled `NM-CLUSTER` elements `PduLength` is set to `PNC-VECTOR-LENGTH` of the `SYSTEM`. If `PNC-VECTOR-LENGTH` is not defined a warning is issued and `PduLength` is not set. `SysTPduToFrameMappingRef` is set to the AUTOSAR path of the `PDU-TO-FRAME-MAPPING` that references the PDU. If no `PDU-TO-FRAME-MAPPING` references the PDU, `SysTPduToFrameMappingRef` is not configured.

If the configured `ECU-INSTANCE` contains a server component which receives its requests and sends its responses via Ethernet, the request/response PDUs are extended with Meta-Data that allows the server to keep track of the `SocketConnection` via which a request was received in order to use the same `SocketConnection` for the transmission of the response.

Additional Meta-Data is configured if at least one PDU container is created for a PDU instance that fulfills the following conditions:

▶ The PDU is of type `I-SIGNAL-I-PDU`

▶ The `PDU-TRIGGERING` that references the PDU is contained in an `ETHERNET-PHYSICAL-CHANNEL`

▶ The `PDU-TRIGGERING` is sent or received via a *local* `SOCKET-ADDRESS` of the configured `ECU-INS-TANCE` on that `ETHERNET-PHYSICAL-CHANNEL`. For more information about `SOCKET-ADDRESS`, see [Section 3.4.43, "SoAd"](#).

▶ The `I-SIGNAL-I-PDU` contains exactly one `I-SIGNAL`

▶ The `SYSTEM-SIGNAL` of the `I-SIGNAL` is referenced by a `CLIENT-SERVER-TO-SIGNAL-MAPPING` via `CALL-SIGNAL-REF` if the configured `ECU-INSTANCE` receives the `I-SIGNAL`, or via `RETURN-SIG-NAL-REF` if the configured `ECU-INSTANCE` sends the `I-SIGNAL`

In `EcucPduCollection/MetaDataType`, one container is created using `MetaDataTypeSocketConnec-tionId` as name. Into that container a `MetaDataItem` sub container named `MetaDataItemSocketCon-nectionId` is inserted. The `MetaDataItemLength` parameter of the sub container is set to `2`, the `Meta-DataItemType` parameter of the sub container is set to `SOCKET_CONNECTION_ID_16`.

The `MetaDataTypeRef` parameter of every `EcucPduCollection/Pdu` container for which the conditions above apply is configured to refer to the `MetaDataTypeSocketConnectionId` container.

If the configured `ECU-INSTANCE` sends or receives PDUs via a `J-1939-CLUSTER`, the PDUs are extended with Meta-Data that allow the `J1939` BSW modules to access the CAN-Id that was used for sending or receiving that PDU.

CAN-Id related Meta-Data is configured if at least one PDU container is created for a PDU instance that fulfills the following conditions:

▶ The `PDU-TRIGGERING` that references the PDU is contained in a `CAN-PHYSICAL-CHANNEL` which in turn is contained in a `J-1939-CLUSTER`

▶ There is a `CAN-FRAME` which contains a `PDU-TO-FRAME-MAPPING` that refers to the PDU.

▶ There is a `CAN-FRAME-TRIGGERING` that references the `CAN-FRAME` and which belongs to the same `CAN-PHYSICAL-CHANNEL` as the `PDU-TRIGGERING`.

▶ Both `CAN-FRAME` and PDU are either sent or received by the configured `ECU-INSTANCE`

In `EcucPduCollection/MetaDataType`, one container is created using `MetaDataTypeCanId` as name. Into that container a `MetaDataItem` sub container named `MetaDataItemCanId` is inserted. The `Meta-DataItemLength` parameter of the sub container is set to `4`, the `MetaDataItemType` parameter of the sub container is set to `CAN_ID_32`.

The `MetaDataTypeRef` parameter of every `EcucPduCollection/Pdu` container for which the conditions above apply is configured to refer to the `MetaDataTypeCanId` container.

Meta-Data for the `J1939Tp` module is configured if at least one PDU container has been created for a PDU for which the following conditions are true:

▶ The `PDU-TRIGGERING` that references the PDU is contained in a `CAN-PHYSICAL-CHANNE` which in turn is contained in a `J-1939-CLUSTER`

▶ There is a `J-1939-TP-PG` which refers to the PDU.

▶ The `J-1939-TP-PG` is contained in a `J-1939-TP-CONNECTION` which the configured `ECU-INSTANCE` either sends or receives.

In `EcucPduCollection/MetaDataType`, one container is created, named `MetaDataTypeJ1939TpSdu`. The following `MetaDataItem` sub containers are added:

ShortName	MetaDataItemLength	MetaDataItemType
Priority	1	PRIORITY_8
SourceAddress	2	SOURCE_ADDRESS_16
TargetAddress	2	TARGET_ADDRESS_16

The `MetaDataTypeRef` parameter of every PDU container that has been set up for a PDU that fulfills the conditions above is configured to refer to the `MetaDataTypeJ1939TpSdu` container.

3.4.15. EcuM

Configuration parameters	Mapping description

`EcuMConfigura-`
`tion/EcuMCommonCon-`
`figuration/EcuMWake-`
`upSource`
	For every `PNC-MAPPING` for which a `ComMPnc` container has been created in the `ComM` configuration, see Section 3.4.9, "ComM", one `EcuMWake-`
	`upSource` container is created if at least one of the following conditions is met:
	
	▶ The `PNC-SYNCHRONOUS-WAKEUP` field of the imported `ECU-INSTANCE` is set to `true`.
	
	▶ The `PNC-WAKEUP-ENABLE` field of the `PNC-MAPPING` is set to `true`.
	
	The name of the `EcuMWakeupSource` container is `ECUM_WKSOURCE_-`
	`<pncName>`, where `<pncName>` is the name of the `ComMPnc` container in the `ComM` module configuration. The `EcuMComMPNCRef` reference is configured to reference that `ComMPnc` container.
	
	For every `ETHERNET-PHYSICAL-CHANNEL` and for every `COMMUNI-`
	`CATION-CLUSTER` for which a `ComMChannel` container has been created in the `ComM` configuration, see Section 3.4.9, "ComM", one `EcuMWake-`
	`upSource` container is created if the `COMMUNICATION-CONNECTOR` that links the `ETHERNET-PHYSICAL-CHANNEL` or `COMMUNICATION-CLUSTER` to the imported `ECU-INSTANCE` has its `CREATE-ECU-WAKEUP-SOURCE` field set to `true`. The name of the `EcuMWakeupSource` container is `ECUM_WKSOURCE_<ChannelName>`, where `<ChannelName>` is the name of the `ComMChannel` container in the `ComM` module configuration. The

Configuration parameters	Mapping description
	`EcuMComMChannelRef` reference is configured to reference that `ComM-Channel` container.

3.4.16. Eth

Configuration parameters	Mapping description
`EthConfigSet/EthCtrlConfig`	Every `ETHERNET-COMMUNICATION-CONTROLLER` which belongs to the imported `ECU-INSTANCE` and which is connected to at least one `ETHERNET-CLUSTER` is considered a *physical* `ETHERNET-COMMUNICATION-CONTROLLER` of the `ECU-INSTANCE`. For every *physical* `ETHERNET-COMMUNICATION-CONTROLLER` of the `ECU-INSTANCE`, one `EthCtrlConfig` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `ETHERNET-COMMUNICATION-CONTROLLER`. If the `Eth` module provides the parameter `EthCtrlRxBufLenByte` in `EthCtrlConfig`, this parameter is set to `MAXIMUM-RECEIVE-BUFFER-LENGTH`. If `EthCtrlRxBufLenByte` is not present, `MAXIMUM-RECEIVE-BUFFER-LENGTH` is used to configure `EthCtrlConfigIngress/EthCtrlConfigIngressFifo/EthCtrlConfigIngressFifoBufLenByte`. If the `Eth` module provides the parameter `EthCtrlTxBufLenByte` in `EthCtrlConfig`, this parameter is set to `MAXIMUM-TRANSMIT-BUFFER-LENGTH`. If `EthCtrlTxBufLenByte` is not present, `MAXIMUM-TRANSMIT-BUFFER-LENGTH` is used to configure `EthCtrlConfigEgress/EthCtrlConfigEgressFifo/EthCtrlConfigEgressFifoBufLenByte`. `EthCtrlPhyAddress` is set to `MAC-UNICAST-ADDRESS`. If `MAC-UNICAST-ADDRESS` does not represent a valid MAC address, a warning is reported and `EthCtrlPhyAddress` is not set.

3.4.17. EthIf

Configuration parameters	Mapping description
`EthIfGeneral`	If the imported `ECU-INSTANCE` has its `ETH-SWITCH-PORT-GROUP-DERIVATION` set to `true` and at least one of the `ETHERNET-CLUSTER` elements to which the imported `ECU-INSTANCE` is connected has its `COUPLING-PORT-SWITCHOFF-DELAY` defined, the maximum `COUPLING-PORT-SWITCHOFF-`

Configuration parameters	Mapping description
	DELAY value of all connected ETHERNET-CLUSTER elements is configured in EthIfSwitchOffPortTimeDelay. If the imported ECU-INSTANCE has its ETH-SWITCH-PORT-GROUP-DERI-VATION set to true and at least one of the ETHERNET-CLUSTER elements to which the imported ECU-INSTANCE is connected has its COUPLING-PORT-STARTUP-ACTIVE-TIME defined, the maximum COUPLING-PORT-START-UP-ACTIVE-TIME value of all connected ETHERNET-CLUSTER elements is configured in EthIfPortStartupActiveTime.
EthIfCon-figSet/EthIf-PhysController	For each *physical* ETHERNET-COMMUNICATION-CONTROLLER of the imported ECU-INSTANCE (see Section 3.4.16, "Eth"), an EthIfPhysController container is created. The container name is the SHORT-NAME of the ETHER-NET-COMMUNICATION-CONTROLLER. EthIfEthCtrlRef references the container created for the *physical* ETHER-NET-COMMUNICATION-CONTROLLER in the Eth module.
EthIfCon-figSet/EthIfCon-troller	A *physical* ETHERNET-COMMUNICATION-CONTROLLER of the imported ECU-INSTANCE (see Section 3.4.16, "Eth") is connected to one or more ETHER-NET-PHYSICAL-CHANNEL elements. Such a connection exists if an ETHER-NET-COMMUNICATION-CONNECTOR of the imported ECU-INSTANCE references the ETHERNET-COMMUNICATION-CONTROLLER and is referenced by the ETHERNET-PHYSICAL-CHANNEL. Each of the connections represents a *virtual* ETHERNET-COMMUNICATION-CONTROLLER of the ECU-INSTANCE. For each *virtual* ETHERNET-COMMUNICATION-CONTROLLER of the imported ECU-INSTANCE, an EthIfController container is created. The container name is <PREFIX><name_cc>_<name_chn>, where <name_cc> is the SHORT-NAME of the ETHERNET-COMMUNICATION-CONTROLLER and <name_chn> is the SHORT-NAME of the ETHERNET-PHYSICAL-CHANNEL. EthIfPhysControllerRef references the EthIfPhysController container that has been created for the *physical* ETHERNET-COMMUNICATION-CON-TROLLER. If the ETHERNET-PHYSICAL-CHANNEL that belongs to the virtual ETHER-NET-COMMUNICATION-CONTROLLER contains a VLAN that provides a VLAN-IDENTIFIER, EthIfVlanId is set to the value of VLAN-IDENTIFIER. If the ETHERNET-COMMUNICATION-CONNECTOR or the ETHERNET-COM-MUNICATION-CONTROLLER provide a valid MAXIMUM-TRANSMISSION-UNIT, EthIfCtrlMtu is set to the value of MAXIMUM-TRANSMISSION-UNIT.

Configuration parameters	Mapping description
	If an `EthIfSwitchPortGroup` container is created for the `ETHERNET-PHYSICAL-CHANNEL` of this `EthIfController` container, `EthIfSwitchRefOR-PortGroupRef` is set up to reference that `EthIfSwitchPortGroup` container.
`EthIfConfigSet/EthIfSwitch`	For each `COUPLING-ELEMENT` that has its `COUPLING-TYPE` set to `SWITCH` and which references the imported `ECU-INSTANCE` in `ECU-INSTANCE-REF`, one `EthIfSwitch` container is created. The container name is the `SHORT-NAME` of the `COUPLING-ELEMENT`. `EthIfSwitchRef` references the container created for the `COUPLING-ELEMENT` in the `EthSwt` module.
`EthIfConfigSet/EthIfSwitchPortGroup`	If the imported `ECU-INSTANCE` has its `ETH-SWITCH-PORT-GROUP-DERIVATION` parameter set to `true`, the configuration algorithm retrieves the list of all `COUPLING-PORT` elements of any `COUPLING-ELEMENT` for which an `EthIfSwitch` container is created. For each `ETHERNET-PHYSICAL-CHANNEL` and for each `PNC-MAPPING` that is referenced by any `COUPLING-PORT`, one `EthIfSwitchPortGroup` container is created. The container name is either `VLANSwtPortGrp_<VLAN>`, where `<VLAN>` is the name of the `ETHERNET-PHYSICAL-CHANNEL`, or `PNCSwtPortGrp_<PncID>`, where `<PncID>` is the `PNC-IDENTIFIER` of the `PNC-MAPPING`. `EthIfPortRef` references are configured to refer to each `EthSwtPort` container in `EthSwt` that corresponds to a `COUPLING-PORT` which references the `ETHERNET-PHYSICAL-CHANNEL` or the `PNC-MAPPING` for which this `EthIfSwitchPortGroup` container is created. `EthIfSwitchPortGroupRefSemantics` is only configured in `EthIfSwitchPortGroup` containers that are created for an `ETHERNET-PHYSICAL-CHANNEL`. If there is any `PNC-MAPPING` of the configured `ECU-INSTANCE` which is related to any `PDU` which is sent over the `ETHERNET-PHYSICAL-CHANNEL`, the parameter value is set to `ETHIF_SWITCH_-PORT_GROUP_LINK_INFO`, otherwise it is set to `ETHIF_SWITCH_PORT_-GROUP_CONTROL`.

3.4.18. EthSM

Configuration parameters	Mapping description
`EthSMNetwork`	For every `ETHERNET-PHYSICAL-CHANNEL` that is connected to at least one *virtual* `ETHERNET-COMMUNICATION-CONTROLLER` of the imported `ECU-INSTANCE` (see [Section 3.4.17, "EthIf"](#)), an `EthSMNetwork` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `ETHERNET-PHYSICAL-CHANNEL`. `EthSMComMNetworkHandleRef` references the `ComMChannel` container in the `ComM` module configuration, which has been created for the `ETHERNET-PHYSICAL-CHANNEL`. `EthSMEthIfControllerRef` references the `EthIfController` container in the `EthIf` module configuration, which has been created for the first virtual `ETHERNET-COMMUNICATION-CONTROLLER` that is connected to the `ETHERNET-PHYSICAL-CHANNEL`.

3.4.19. EthSwt

Configuration parameters	Mapping description
`EthSwtConfig`	One `EthSwtConfig` container is created for each `COUPLING-ELEMENT` that references the imported `ECU-INSTANCE` via `ECU-INSTANCE-REF` and has its `COUPLING-TYPE` set to `SWITCH`. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `COUPLING-ELEMENT`.
`EthSwtConfig/EthSwtPort`	For each `COUPLING-PORT` contained in a configured `COUPLING-ELEMENT` one `EthSwtPort` subcontainer is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `COUPLING-PORT`. `EthSwtPortRole` is set depending on the value of `COUPLING-PORT/COUPLING-PORT-ROLE`: ▶ `ETHSWT_HOST_PORT` for `HOST-PORT` ▶ `ETHSWT_UP_LINK_PORT` for `UP-LINK-PORT` For `STANDARD-PORT`, the parameter is not configured. `EthSwtPortMacLayerType` is set depending on the value of `COUPLING-PORT/MAC-LAYER-TYPE`:

Configuration parameters	Mapping description
	▶ `ETHSWT_PORT_MAC_LAYER_TYPE_XGMII` for `XG-MII` ▶ `ETHSWT_PORT_MAC_LAYER_TYPE_XMII` for `X-MII` ▶ `ETHSWT_PORT_MAC_LAYER_TYPE_XXGMII` for `XXG-MII` `EthSwtPortPhysicalLayerType` is set depending on the value of `COU-PLING-PORT/PHYSICAL-LAYER-TYPE`: ▶ `ETHSWT_PORT_100BASE_TX` for `100BASE-TX` ▶ `ETHSWT_PORT_1000BASE_T` for `1000BASE-T` ▶ `ETHSWT_PORT_100BASE_T1` for `100BASE-T1` ▶ `ETHSWT_PORT_1000BASE_T1` for `1000BASE-T1`
`EthSwtConfig/EthSwtPort/EthSwtPortPredefinedMacAddresses`	The MAC addresses to configure are taken from the `MAC-MULTICAST-ADDRESS` value of the `MAC-MULTICAST-GROUP` elements which the `COUPLING-PORT` references via `MAC-MULTICAST-ADDRESS-REFS/MAC-MULTICAST-ADDRESS-REF`. For each distinct MAC address value one entry is added to `EthSwtPortPredefinedMacAddresses`.
`EthSwtConfig/EthSwtPort/EthSwtPortIngress`	For each `COUPLING-PORT`, one `EthSwtPort/EthSwtPortIngress` subcontainer is created. `EthSwtPortIngressDefaultVlan` is set to `VLAN/VLAN-IDENTIFIER` if `COUPLING-PORT/DEFAULT-VLAN-REF` references an `ETHERNET-PHYSICAL-CHANNEL` that contains a `VLAN` which in turn defines a valid `VLAN-IDENTIFIER`. `EthSwtPortIngressDefaultPriority` is set to `VLAN-MEMBERSHIP/DEFAULT-PRIORITY` if the `VLAN-MEMBERSHIP` is contained in `COUPLING-PORT/VLAN-MEMBERSHIPS` and `VLAN-MEMBERSHIP/VLAN-REF` references the `ETHERNET-PHYSICAL-CHANNEL` which is also referenced by `COUPLING-PORT/DEFAULT-VLAN-REF`. Otherwise, if the `DEFAULT-PRIORITY` values of all `VLAN-MEMBERSHIPS` of a `COUPLING-PORT` are identical, the `EthSwtPortIngressDefaultPriority` is set to this value. `EthSwtPortTrafficClassAssignment` is set to `COUPLING-PORT-TRAFFIC-CLASS-ASSIGNMENT/TRAFFIC-CLASS` if the `COUPLING-PORT` contains exactly one valid `COUPLING-PORT-DETAILS/ETHERNET-TRAFFIC-CLASS-ASSIGNMENTS/ETHERNET-TRAFFIC-CLASS-ASSIGNMENT` subelement which in turn contains a valid `TRAFFIC-CLASS` element but no `PRIORITYS/PRIORITY` subelements.

Configuration parameters	Mapping description
	`EthSwtPortTrafficClassAssignment` is set to `COUPLING-PORT-TRAF-FIC-CLASS-ASSIGNMENT/TRAFFIC-CLASS` if the `COUPLING-PORT` contains exactly one valid `COUPLING-PORT-DETAILS/ETHERNET-TRAFFIC-CLASS-ASSIGNMENTS/ETHERNET-TRAFFIC-CLASS-ASSIGNMENT` subelement which in turn contains a valid `TRAFFIC-CLASS` element but no `PRIORITYS/PRIORITY` subelements.
`EthSwtConfig/EthSwtPort/EthSwtPortIngress/EthSwtPriorityRegeneration`	If the `COUPLING-PORT` contains a valid `COUPLING-PORT-DETAILS` subelement, then one `EthSwtPriorityRegeneration` container is created for each `ETHERNET-PRIORITY-REGENERATIONS/ETHERNET-PRIORITY-REGENERATION` element contained in `COUPLING-PORT/COUPLING-PORT-DETAILS` that contains a valid and distinct `INGRESS-PRIORITY` value. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `ETHERNET-PRIORITY-REGENERATION`. `EthSwtPriorityRegenerationIngressPriority` is set to `ETHERNET-PRIORITY-REGENERATION/INGRESS-PRIORITY`, `EthSwtPriorityRegenerationRegeneratedPriority` is set to `ETHERNET-PRIORITY-REGENERATION/REGENERATED-PRIORITY`.
`EthSwtConfig/EthSwtPort/EthSwtPortIngress/EthSwtPriorityTrafficClassAssignment`	If the `COUPLING-PORT` contains a valid `COUPLING-PORT-DETAILS` subelement and `COUPLING-PORT-DETAILS/ETHERNET-TRAFFIC-CLASS-ASSIGNMENTS` contains at least one `ETHERNET-TRAFFIC-CLASS-ASSIGNMENT` element which in turn contains at least one `PRIORITY` element in `ETHERNET-TRAFFIC-CLASS-ASSIGNMENT/PRIORITYS`, then one `EthSwtPriorityTrafficClassAssignment` container is created for each distinct value in `PRIORITY`. The container name is `<PREFIX><name>_<prio>`, where `<name>` is the `SHORT-NAME` of the `ETHERNET-TRAFFIC-CLASS-ASSIGNMENT` and `<prio>` is the value of `ETHERNET-TRAFFIC-CLASS-ASSIGNMENT/PRIORITYS/PRIORITY`. `EthSwtPriorityTrafficClassAssignmentPriority` is set to `ETHERNET-TRAFFIC-CLASS-ASSIGNMENT/PRIORITYS/PRIORITY`, `EthSwtPriorityTrafficClassAssignmentTrafficClass` is set to `ETHERNET-TRAFFIC-CLASS-ASSIGNMENT/TRAFFIC-CLASS`.
`EthSwtConfig/EthSwtPort/EthSwtPortEgress`	For each `COUPLING-PORT` that contains at least one entry in `COUPLING-PORT-DETAILS/COUPLING-PORT-STRUCTURAL-ELEMENTS`, one `EthSwtPortEgress` container is created. If `COUPLING-PORT-DETAILS/COUPLING-PORT-STRUCTURAL-ELEMENTS/LAST-EGRESS-SCHEDULER-REF` references a valid `COUPLING-PORT-SCHEDULER` of the `COUPLNG-PORT`, then `EthSwtPortEgressLastSched-`

Configuration parameters	Mapping description
	`ulerRef` is configured to reference the container that has been created for that `COUPLING-PORT-SCHEDULER`.
`EthSwtConfig/EthSwtPort/EthSwtPortEgress/EthSwtPortFifo`	For each `COUPLING-PORT-FIFO` subelement of `COUPLING-PORT-STRUCTURAL-ELEMENTS`, one `EthSwtPortFifo` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `COUPLING-PORT-FIFO`. For each distinct and valid value in `COUPLING-PORT-FIFO/ASSIGNED-TRAFFIC-CLASSS/ASSIGNED-TRAFFIC-CLASS`, one `EthSwtPortFifoTrafficClassAssignment` subcontainer is created that contains that value.
`EthSwtConfig/EthSwtPort/EthSwtPortEgress/EthSwtPortShaper`	For each `COUPLING-PORT-SHAPER` subelement of `COUPLING-PORT-STRUCTURAL-ELEMENTS`, one `EthSwtPortShaper` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `COUPLING-PORT-SHAPER`. `EthSwtPortShaperIdleSlope` is set to `COUPLING-PORT-SHAPER/IDLE-SLOPE`, `EthSwtPortEgressPredecessorFifoRef` is configured to reference the `EthSwtPortFifo` container that has been created for the `COUPLING-PORT-FIFO` which `COUPLING-PORT-SHAPER/REDECESSOR-FIFO-REF` references.
`EthSwtConfig/EthSwtPort/EthSwtPortEgress/EthSwtPortScheduler`	For each `COUPLING-PORT-SCHEDULER` subelement of `COUPLING-PORT-STRUCTURAL-ELEMENTS`, one `EthSwtPortScheduler` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `COUPLING-PORT-SCHEDULER`. `EthSwtPortSchedulerAlgorithm` is set depending on the value of `COUPLING-PORT-SCHEDULER/PORT-SCHEDULER`: ▶ `ETHSWT_SCHEDULER_DEFICIT_ROUND_ROBIN` for `DEFICIT-ROUND-ROBIN` ▶ `ETHSWT_SCHEDULER_STRICT_PRIORITY` for `STRICT-PRIORITY` ▶ `ETHSWT_SCHEDULER_WEIGHTED_ROUND_ROBIN` for `WEIGHTED-ROUND-ROBIN` For each `COUPLING-PORT-STRUCTURAL-ELEMENT` referenced by `COUPLING-PORT-SCHEDULER/PREDECESSOR-REFS/PREDECESSOR-REF` one `EthSwtPortSchedulerPredecessor` subcontainer is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the referenced `COUPLING-PORT-STRUCTURAL-ELEMENT`.

Configuration parameters	Mapping description
	`EthSwtPortSchedulerPredecessorOrder` is set to `10 * (<numberOfElementsInPredecessorList> - <positionInPredecessorList>)`, i.e. `0` for the last referenced element in `COUPLING-PORT-SCHEDULER/PREDECESSOR-REFS`, `10` for the second to last referenced, `20` for the third to last referenced element, etc. `EthSwtPortEgressPredecessorRef` is configured to reference the container that has been created for the `COUPLING-PORT-STRUCTURAL-ELEMENT` which `COUPLING-PORT-SCHEDULER/PREDECESSOR-REFS/PREDECESSOR-REF` references.
`EthSwtConfig/EthSwtPort/EthSwtPortVlanMembership`	The VLAN ID of a `VLAN-MEMBERSHIP` is the `VLAN-IDENTIFIER` of the `VLAN` contained in the `ETHERNET-PHYSICAL-CHANNEL` that `VLAN-MEMBERSHIP/VLAN-REF` refers to. For each `VLAN-MEMBERSHIP` that is associated with a distinct valid VLAN ID, one `EthSwtPortVlanMembership` container is created. The container name is `EthSwtPortVlanMembership_<VLAN_ID>`, where `<VLAN_ID>` is the associated VLAN ID. `EthSwtPortVlanMembershipId` is set to the VLAN ID. `EthSwtPortVlanForwardingType` is set depending on the value of `VLAN-MEMBERSHIP/SEND-ACTIVITY`: ► `ETHSWT_NOT_SENT` for `NOT-SENT` ► `ETHSWT_SENT_TAGGED` for `SENT-TAGGED` ► `ETHSWT_SENT_UNTAGGED` for `SENT-UNTAGGED`

3.4.20. EthTSyn

Configuration parameters	Mapping description
`EthTSynGeneral`	One `EthTSynGeneral` container is created if there is at least one `GLOBAL-TIME-DOMAIN` that contains a `GLOBAL-TIME-ETH-MASTER` or a `GLOBAL-TIME-ETH-SLAVE` element that references an `ETHERNET-COMMUNICATION-CONNECTOR` of the imported `ECU-INSTANCE`. `EthTSynMessageCompliance` is configured if all of the `GLOBAL-TIME-DOMAIN` elements that contain a `ETH-GLOBAL-TIME-DOMAIN-PROPS` element providing a valid value in `MESSAGE-COMPLIANCE` have that value set consistently. If that value is `IEEE802-1AS`, `EthTSynMessageCompliance` is set to `true`. If that value is `IEEE802-1AS-AUTOSAR`, `EthTSynMessageCompliance` is set to `false`.

Configuration parameters	Mapping description
	The parameter `EthTSynDestPhyAddr` is configured by collecting all valid `DESTINATION-PHYSICAL-ADDRESS` values of `ETH-GLOBAL-TIME-DO-MAIN-PROPS` elements that are aggregated by `GLOBAL-TIME-DOMAIN` elements of the imported `ECU-INSTANCE`. If the collected values refer to the same valid MAC address, that address is used to configure `EthTSynDestPhyAddr`.
`EthTSynGlobalTime-Domain`	One `EthTSynGlobalTimeDomain` container is created for each `GLOBAL-TIME-DOMAIN` that contains a `GLOBAL-TIME-ETH-MASTER` or a `GLOBAL-TIME-ETH-SLAVE` element that references an `ETHERNET-COMMUNICATION-CONNECTOR` of the imported `ECU-INSTANCE`. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `GLOBAL-TIME-DOMAIN`. `EthTSynGlobalTimeDomainId` is set to `DOMAIN-ID`. `EthTSynSynchronizedTimeBaseRef` references the corresponding `StbMSynchronizedTimeBase` container in the `StbM` module. If the `GLOBAL-TIME-DOMAIN` is a subdomain of a parent `GLOBAL-TIME-DOMAIN`, `EthTSynSynchronizedTimeBaseRef` references the `StbMSynchronizedTimeBase` container that has been created for the parent `GLOBAL-TIME-DOMAIN`. For further information about the configuration of the `StbM` module, see [Section 3.4.45, "StbM"](#) If the release version of `EthTSyn` is `4.4.0` or newer, `EthTSynGlobalTimeEthIfRef` references the `EthIfController` container that has been created for the `ETHERNET-COMMUNICATION-CONNECTOR` to which the `GLOBAL-TIME-ETH-MASTER` or `GLOBAL-TIME-ETH-SLAVE` refers. For `EthTSyn` versions older than `4.4.0` the parameter is not configured. For further information about the configuration of the `EthIf` module, see [Section 3.4.17, "EthIf"](#). If the release version of `EthTSyn` is `4.4.0` or newer and the Ethernet `GLOBAL-TIME-DOMAIN` contains a valid `ETH-GLOBAL-TIME-DOMAIN-PROPS` subelement that in turn contains a valid `VLAN-PRIORITY` subelement, `EthTSynFramePrio` is configured to hold the value of `VLAN-PRIORITY`. For `EthTSyn` versions older than `4.4.0` the parameter is not configured. If the release version of `EthTSyn` is `4.4.0` or newer, `EthTSynGlobalTimeDebounceTime` is set to `GLOBAL-TIME-DOMAIN/DEBOUNCE-TIME`. For `EthTSyn` versions older than `4.4.0` the parameter is not configured.
`EthTSynGlobalTime-Domain/EthTSyn-`	If the Ethernet `GLOBAL-TIME-DOMAIN` contains a valid `ETH-GLOBAL-TIME-DOMAIN-PROPS` subelement that contains a non-empty ID list in `FUP-DATA-ID-LISTS`, that list is used to configure the container list `EthTSynGlobal-`

Configuration parameters	Mapping description
`GlobalTimeFollowUp-DataIDList`	`TimeDomain/EthTSynGlobalTimeFollowUpDataIDList/EthTSynGlob-alTimeFollowUpDataIDListElement`. One subcontainer is created per element named `Element_<idx>`, where `<idx>` is the zero-based index of the element within the list. `EthTSynGlobal-TimeFollowUpDataIDListIndex` contains the zero-based index of the element within the list, `EthTSynGlobalTimeFollowUpDataIDListValue` is configured to contain the value of the element.
`EthTSynGlobalTime-Domain/EthTSynPort-Config`	One `EthTSynGlobalTimeDomain` is created for the `ETHERNET-COMMUNI-CATION-CONNECTOR` which connects the imported `ECU-INSTANCE` to the Ethernet `GLOBAL-TIME-DOMAIN`. The container name is the same as the name of the parent `EthTSynGlobalTimeDomain` container. If the `GLOBAL-TIME-DOMAIN` for which the `EthTSynGlobalTimeDomain` container has been created contains an `ETH-GLOBAL-TIME-DOMAIN-PROPS` element that in turn contains one or more `ETH-GLOBAL-TIME-MANAGED-COU-PLING-PORT` elements that refer to a `COUPLING-PORT` for which a `EthSwt-Port` container has been created, see [Section 3.4.19, "EthSwt"](#), then one additional `EthTSynPortConfig` is created for each of these `ETH-GLOBAL-TIME-MANAGED-COUPLING-PORT` elements. In that case, the container name is the `SHORT-NAME` of the `COUPLING-PORT` and `EthTSynSwitchManage-mentEthSwitchPortRef` is configured to refer to the `EthSwtPort` container in the `EthSwt` module. If the release version of `EthTSyn` is older than `4.4.0`, `EthTSynGlobal-TimeEthIfRef` references the `EthIfController` container that has been created for the `ETHERNET-COMMUNICATION-CONNECTOR` to which the `GLOB-AL-TIME-ETH-MASTER` or `GLOBAL-TIME-ETH-SLAVE` refers. For `EthTSyn` versions `4.4.0` and newer the parameter is not configured. For further information about the configuration of the `EthIf` module, see [Section 3.4.17, "EthIf"](#). If the release version of `EthTSyn` is older than `4.4.0` and the Ethernet `GLOB-AL-TIME-DOMAIN` contains a valid `ETH-GLOBAL-TIME-DOMAIN-PROPS` subelement that in turn contains a valid `VLAN-PRIORITY` subelement, `EthTSynFramePrio` is configured to hold the value of `VLAN-PRIORITY`. For `EthTSyn` versions `4.4.0` and newer the parameter is not configured. If the release version of `EthTSyn` is older than `4.4.0`, `EthTSynGlobal-TimeDebounceTime` is set to `GLOBAL-TIME-DOMAIN/DEBOUNCE-TIME`. For `EthTSyn` versions `4.4.0` and newer the parameter is not configured.

Configuration parameters	Mapping description
`EthTSynGlobalTime-Domain/EthTSynPort-Config/EthTSynPde-layConfig`	The `EthTSynPdelayConfig` container is created for each `ETH-GLOB-AL-TIME-MANAGED-COUPLING-PORT` of the Ethernet `GLOBAL-TIME-DOMAIN` that refers to a `COUPLING-PORT` that either belongs to an `ETHERNET-COM-MUNICATION-CONTROLLER` or to an `COUPLING-ELEMENT` of the imported `ECU-INSTANCE`. `EthTSynGlobalTimePdelayRespEnable` is set to `ETH-GLOBAL-TIME-MANAGED-COUPLING-PORT/PDELAY-RESPONSE-ENABLED`. `EthTSynGlobalTimePropagationDelay` is set to `COUPLING-PORT/COU-PLING-PORT-DETAILS/GLOBAL-TIME-PROPS/PROPAGATION-DELAY` of the `COUPLING-PORT` that is referenced by `ETH-GLOBAL-TIME-MANAGED-COU-PLING-PORT/COUPLING-PORT-REF`. `EthTSynGlobalTimeTxPdelayReqPeriod` is set to `ETH-GLOBAL-TIME-MANAGED-COUPLING-PORT/PDELAY-REQUEST-PERIOD`. `EthTSynPdelayLatencyThreshold` is set to `ETH-GLOBAL-TIME-MAN-AGED-COUPLING-PORT/PDELAY-LATENCY-THRESHOLD`. `EthTSynPdelayRespAndRespFollowUpTimeout` is set to `ETH-GLOB-AL-TIME-MANAGED-COUPLING-PORT/PDELAY-RESP-AND-RESP-FOL-LOW-UP-TIMEOUT`.
`EthTSynGlobal-TimeMaster`	This container is only configured if the parent `EthTSynPortConfig` container has been created for the `ETHERNET-COMMUNICATION-CONNECTOR` which connects the imported `ECU-INSTANCE` to the Ethernet `GLOBAL-TIME-DOMAIN` and the `GLOBAL-TIME-ETH-MASTER` of the `GLOBAL-TIME-DOMAIN` references that `ETHERNET-COMMUNICATION-CONNECTOR`. If the release version of `EthTSyn` is `4.4.0` or newer, the `EthTSynGlobal-TimeMaster` container is added as `EthTSynGlobalTimeDomain` child container. If the release version is older than `4.4.0`, the container is added as `EthTSynGlobalTimeDomain/EthTSynPortConfig` child container. `EthTSynGlobalTimeTxPeriod` is set to `SYNC-PERIOD` of the `GLOB-AL-TIME-ETH-MASTER`. `EthTSynCyclicMsgResumeTime` is set to `IMMEDIATE-RESUME-TIME` of the `GLOBAL-TIME-ETH-MASTER`. `EthTSynGlobalTimeTxCrcSecured` is set to `CRC-SECURED` of the `GLOB-AL-TIME-ETH-MASTER`.

Configuration parameters	Mapping description
	`EthTSynTLVFollowUpOFSSubTLV` is set to `GLOBAL-TIME-ETH-MASTER/ SUB-TLV-CONFIG/OFS-SUB-TLV`. `EthTSynTLVFollowUpStatusSubTLV` is set to `GLOBAL-TIME-ETH- MASTER/SUB-TLV-CONFIG/STATUS-SUB-TLV`. `EthTSynTLVFollowUpTimeSubTLV` is set to `GLOBAL-TIME-ETH-MASTER/ SUB-TLV-CONFIG/TIME-SUB-TLV`. `EthTSynTLVFollowUpUserDataSubTLV` is set to `GLOBAL-TIME-ETH- MASTER/SUB-TLV-CONFIG/USER-DATA-SUB-TLV`.
`EthTSynGlobal- TimeMaster/EthTSyn- CrcTimeFlagsTxSe- cured`	If the `GLOBAL-TIME-DOMAIN` contains an `ETH-GLOBAL-TIME-DO- MAIN-PROPS` sub element which in turn contains an `CRC-FLAGS` element, that sub element is used to configure parameters in the `EthTSynCrcTime- FlagsTxSecured` sub container. The container name is the same as the container name of the `EthTSynGlobalTimeDomain`. `EthTSynCrcCorrectionField` is set to `CRC-FLAGS/CRC-CORREC- TION-FIELD`. `EthTSynCrcDomainNumber` is set to `CRC-FLAGS/CRC-DOMAIN-NUMBER`. `EthTSynCrcMessageLength` is set to `CRC-FLAGS/CRC-MESSAGE-LENGTH`. `EthTSynCrcPreciseOriginTimestamp` is set to `CRC-FLAGS/CRC- PRECISE-ORIGIN-TIMESTAMP`. `EthTSynCrcSequenceId` is set to `CRC-FLAGS/CRC-SEQUENCE-ID`. `EthTSynCrcSourcePortIdentity` is set to `CRC-FLAGS/CRC-SOURCE- PORT-IDENTITY`.
`EthTSynGlobal- TimeSlave`	This container is only configured if the parent `EthTSynPortConfig` container has been created for the `ETHERNET-COMMUNICATION-CONNECTOR` which connects the imported `ECU-INSTANCE` to the Ethernet `GLOBAL-TIME-DOMAIN` and one of the `GLOBAL-TIME-ETH-SLAVE` elements of the `GLOBAL-TIME-DOMAIN` references that `ETHERNET-COMMUNICATION-CONNECTOR`. If the release version of `EthTSyn` is `4.4.0` or newer, the `EthTSynGlobal- TimeSlave` container is added as `EthTSynGlobalTimeDomain` child container. If the release version is older than `4.4.0`, the container is added as `EthTSynGlobalTimeDomain/EthTSynPortConfig` child container.

Configuration parameters	Mapping description
	`EthTSynRxCrcValidated` is set to `CRC-VALIDATED` of the `GLOBAL-TIME-ETH-SLAVE`. `EthTSynGlobalTimeFollowUpTimeout` is set to `FOLLOW-UP-TIME-OUT-VALUE` of the `GLOBAL-TIME-ETH-SLAVE`.
`EthTSynGlobal-TimeSlave/EthTSyn-CrcFlagsRxValidated`	If the `GLOBAL-TIME-DOMAIN` contains an `ETH-GLOBAL-TIME-DO-MAIN-PROPS` sub element which in turn contains an `CRC-FLAGS` element, that sub element is used to configure parameters in the `EthTSynCrcFlagsRxValidated` sub container. The container name is the same as the container name of the `EthTSynGlobalTimeDomain`. The same parameters are configured as described for `EthTSynGlobal-TimeMaster/EthTSynCrcTimeFlagsTxSecured`.

3.4.21. FiM

Configuration parameters	Mapping description
`FiM/FiMConfigSet/FiMFID`	A `FiMFID` is created for each `DiagnosticFunctionIdentifier`.
`FiMConfigSet/FiMSummaryEvent`	Based on the attributes `ActualEvent` or `AliasEvent` of the `DiagnosticFimAliasEventMapping` a `FiMSummaryEvent` container is created for every mapping, as follows: 1. If the `ActualEvent` attribute exists then the `FiMSummaryEvent` is created based on it. 2. If the `AliasEvent` attribute exists then the `FiMSummaryEvent` is created based on it. After going through all the `DiagnosticFimAliasEventMapping`, each `FimAliasEventGroupMapping` is taken into account, and based on the `DiagnosticFimEventGroup` and the `DiagnosticFimAliasEventGroup` attributes a `FiMSummaryEvent` is created for every mapping as follows: 1. If the `DiagnosticFimEventGroup` attribute exists then the `FiMSummaryEvent` is created based on the actual events that it contains, and which were not covered by the `DiagnosticFimAliasEventMapping`. 2. If the `DiagnosticFimAliasEventGroup` attribute exists then the `FiMSummaryEvent` is created based on the alias events that it contains,

Configuration parameters	Mapping description
	and which were not covered by the `DiagnosticFimAliasEventMapping`.
`FiMConfigSet/FiMInhibitionConfiguration`	For every `DiagnosticFunctionIdentifierInhibit` a container of type `FiMInhibitionConfiguration` is created. The name of the container is the short name of `DiagnosticFunctionIdentifierInhibit`.
`FiMConfigSet/FiMInhibitionConfiguration/DiagnosticFunctionIdentifierInhibit/FiMInhEventId`	For every `FiMInhibitionConfiguration` container a `FiMInhEventId` subcontainer is created with the short name `FiMInhEventId`. The `FiMInhEventId` has a `FiMInhRefChoice` container of type choice.
`FiMConfigSet/FiMEventSummary/FiMInputSumEventRef`	For every `FiMEventSummary` container a mandatory `FiMInputSumEventRef` reference is set.
`FiMConfigSet/FiMEventSummary/FiMOutputSumEventRef`	For every `FiMEventSummary` container a mandatory `FiMOutputSumEventRef` reference is set.
`FiMConfigSet/FiMSummaryEventId`	For each `FimAliasEventGroupMapping` a `FiMSummaryEventId` container is created based on the attributes `DiagnosticFimEventGroup` and `DiagnosticFimAliasEventGroup` as follows: 1. If the `DiagnosticFimEventGroup` attribute exists then the `FiMSummaryEventId` is created based on the actual eventGroup. 2. If the `DiagnosticFimAliasEventGroup` attribute exists then the `FiMSummaryEventId` is created based on the alias eventGroup.

3.4.22. Fr

Configuration parameters	Mapping description
`FrMultipleConfiguration/FrController`	For every `FLEXRAY-COMMUNICATION-CONTROLLER` which is connected to a `FLEXRAY-CLUSTER` and which belongs to the imported `ECU-INSTANCE`, a `FrController` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `FLEXRAY-COMMUNICATION-CONTROLLER`.

Configuration parameters	Mapping description
	`FrPAllowHaltDueToClock` is set to `ALLOW-HALT-DUE-TO-CLOCK`. `FrPAllowPassiveToActive` is set to `ALLOW-PASSIVE-TO-ACTIVE`. `FrPChannels` is set depending on which `FLEXRAY-PHYSICAL-CHANNEL` elements the `FLEXRAY-COMMUNICATION-CONTROLLER` is connected to: ► `FR_CHANNEL_A` if connected to `CHANNEL-A`. ► `FR_CHANNEL_B` if connected to `CHANNEL-B`. ► `FR_CHANNEL_AB` if connected to `CHANNEL-A` and `CHANNEL-B`. `FrPClusterDriftDamping` is set to `CLUSTER-DRIFT-DAMPING`. `FrPdAcceptedStartupRange` is set to `ACCEPTED-STARTUP-RANGE`. `FrPDecodingCorrection` is set to `DECODING-CORRECTION`. `FrPDelayCompensationA` is set to `DELAY-COMPENSATION-A`. `FrPDelayCompensationB` is set to `DELAY-COMPENSATION-B`. `FrPdListenTimeout` is set to `LISTEN-TIMEOUT`. `FrPdMicrotick` is set depending on the value of `MICROTICK-DURATION` [ns]: ► `T200NS` for `200`. ► `T100NS` for `100`. ► `T50NS` for `50`. ► `T25NS` for `25`. ► `T12_5NS` for `12.5`. ► For any other values `FrPdMicrotick` is not configured and a warning is reported. `FrPExternalSync` is set to `EXTERNAL-SYNC`. `FrPFallBackInternal` is set to `FALL-BACK-INTERNAL`. `FrPKeySlotId` is set to `KEY-SLOT-ID`. If `KEY-SLOT-ONLY-ENABLED`, `KEY-SLOT-USED-FOR-START-UP`, or `KEY-SLOT-USED-FOR-SYNC` are set to `true`, but `KEY-SLOT-ID` is undefined, a warning is reported. `FrPKeySlotOnlyEnabled` is set to `KEY-SLOT-ONLY-ENABLED`.

Configuration parameters	Mapping description

	`FrPKeySlotUsedForStartup` is set to `KEY-SLOT-USED-FOR-START-UP`.
	
	`FrPKeySlotUsedForSync` is set to `KEY-SLOT-USED-FOR-SYNC`.
	
	`FrPLatestTx` is set to `LATEST-TX`.
	
	`FrPMacroInitialOffsetA` is set to `MACRO-INITIAL-OFFSET-A`.
	
	`FrPMacroInitialOffsetB` is set to `MACRO-INITIAL-OFFSET-B`.
	
	`FrPMicroInitialOffsetA` is set to `MICRO-INITIAL-OFFSET-A`.
	
	`FrPMicroInitialOffsetB` is set to `MICRO-INITIAL-OFFSET-B`.
	
	`FrPMicroPerCycle` is set to `MICRO-PER-CYCLE`.
	
	`FrPNmVectorEarlyUpdate` is set to `NM-VECTOR-EARLY-UPDATE`.
	
	`FrPOffsetCorrectionOut` is set to `OFFSET-CORRECTION-OUT`.
	
	`FrPOffsetCorrectionStart` is set to `OFFSET-CORRECTION-START` of the `FLEXRAY-CLUSTER` to which the `FLEXRAY-COMMUNICATION-CONTROLLER` is connected.
	
	`FrPPayloadLengthDynMax` is set to `MAXIMUM-DYNAMIC-PAYLOAD-LENGTH`.
	
	`FrPRateCorrectionOut` is set to `RATE-CORRECTION-OUT`.
	
	`FrPSamplesPerMicrotick` is set depending on the value of `SAMPLES-PER-MICROTICK`:
	
	► `N1SAMPLES` for 1.
	► `N2SAMPLES` for 2.
	► `N4SAMPLES` for 4.
	► For all other values of `SAMPLES-PER-MICROTICK` `FrPSamplesPerMicrotick` is set to `N1SAMPLES` and a warning is reported.
	
	`FrPSecondKeySlotId` is set to `SECOND-KEY-SLOT-ID`. If `TWO-KEY-SLOT-MODE` is set to `true`, but `SECOND-KEY-SLOT-ID` is undefined, a warning is reported.
	
	`FrPTwoKeySlotMode` is set to `TWO-KEY-SLOT-MODE`
	
	`FrPWakeupChannel` is set depending on the `COMMUNICATION-CONNECTOR` elements of the `FLEXRAY-COMMUNICATION-CONTROLLER`. The `PHYSI-`

Configuration parameters	Mapping description
	CAL-CHANNEL which is connected to the COMMUNICATION-CONNECTOR which has WAKE-UP-CHANNEL set to true determines the value of FrPWakeupChannel: ▶ FR_CHANNEL_A if the CHANNEL-NAME is CHANNEL-A. ▶ FR_CHANNEL_B if the CHANNEL-NAME is CHANNEL-B. If no COMMUNICATION-CONNECTOR has its WAKE-UP-CHANNEL set to true a warning is issued and FrPWakeupChannel is set to FR_CHANNEL_A. FrPWakeupPattern is set to WAKE-UP-PATTERN.
FrMultipleConfiguration/FrController/FrAbsoluteTimer	If no FrAbsoluteTimer container exists yet, a new one named FrAbsoluteTimer is created.

3.4.23. FrArTp

Configuration parameters	Mapping description
FrArTpMultipleConfig/FrArTpChannel	For each FLEXRAY-AR-TP-CHANNEL that contains at least one FLEXRAY-AR-TP-CONNECTION which is processed by the imported ECU-INSTANCE, one FrArTpChannel container is created. The name of the container is FrArTpChannel_<idx>, where <idx> is a zero-based index. FrArTpAckType is configured depending on the value of ACK-TYPE: ▶ FRARTP_ACK_WITH_RT for ACK-WITH-RT. ▶ FRARTP_ACK_WITHOUT_RT for ACK-WITHOUT-RT. ▶ FRARTP_NO for NO-ACK. FrArTpAdrType is configured depending on the value of EXTENDED-ADDRESSING: ▶ FRARTP_TB for true. ▶ FRARTP_OB for false. FrArTpGrpSeg is set to MULTICAST-SEGMENTATION.

Configuration parameters	Mapping description
	`FrArTpLm` is configured depending on the value of `MAXIMUM-MESSAGE-LENGTH`: ► `FRARTP_ISO6` for `ISO-6`. ► `FRARTP_ISO` for `ISO`. ► `FRARTP_L4G` for `I4G`. `FrArTpMaxAr` is set to `MAX-AR`. `FrArTpMaxAs` is set to `MAX-AS`. `FrArTpMaxBs` is set to `MAX-BS`. `FrArTpMaxRn` is set to `MAX-RETRIES`. `FrArTpMaxWft` is set to `MAX-FC-WAIT`. `FrArTpStMin` is set to `MINIMUM-SEPARATION-TIME`. `FrArTpStMinGrpSeg` is set to `MINIMUM-MULTICAST-SEPERATION-TIME`. `FrArTpTc` is set to `CANCELLATION`. `FrArTpTimeBr` is set to `TIME-BR`. `FrArTpTimeCs` is set to `TIME-CS`. `FrArTpTimeoutAr` is set to `TIMEOUT-AR`. `FrArTpTimeoutAs` is set to `TIMEOUT-AS`. `FrArTpTimeoutBs` is set to `TIMEOUT-BS`. `FrArTpTimeoutCr` is set to `TIMEOUT-CR`.
`FrTpMultipleConfig/FrArTpChannel/FrArTpConnection`	The imported `ECU-INSTANCE` processes a `FLEXRAY-AR-TP-CONNECTION` if the `FLEXRAY-AR-TP-CONNECTION` references via `SOURCE-REF` or `TARGET-REFS/TARGET-REF` any `FLEXRAY-AR-TP-NODE` that belongs to the imported `ECU-INSTANCE`. A `FLEXRAY-AR-TP-NODE` belongs to the imported `ECU-INSTANCE` if it references any `COMMUNICATION-CONNECTOR` of the imported `ECU-INSTANCE` via `CONNECTOR-REF`.

Configuration parameters	Mapping description
	The sender `FLEXRAY-AR-TP-NODE` of a `FLEXRAY-AR-TP-CONNECTION` is the `FLEXRAY-AR-TP-NODE` which the `FLEXRAY-AR-TP-CONNECTION` references via `SOURCE-REF` The receiver `FLEXRAY-AR-TP-NODE` elements of a `FLEXRAY-AR-TP-CONNECTION` are the `FLEXRAY-AR-TP-NODE` elements which the `FLEXRAY-AR-TP-CONNECTION` references via `TARGET-REFS/TARGET-REF`. The `TP-ADDRESS` of a `FLEXRAY-AR-TP-NODE` is the `TP-ADDRESS` which it references via `TP-ADDRESS-REF`. The sender `TP-ADDRESS` of a `FLEXRAY-AR-TP-CONNECTION` is the `TP-ADDRESS` of the sender node of the `FLEXRAY-AR-TP-CONNECTION`. The receiver `TP-ADDRESS` of a `FLEXRAY-AR-TP-CONNECTION` is either the `TP-ADDRESS` which the `FLEXRAY-AR-TP-CONNECTION` references via `MULTICAST-REF` or the `TP-ADDRESS` of the first receiver `FLEXRAY-AR-TP-NODE` if `MULTICAST-REF` is not defined. If the sender `FLEXRAY-AR-TP-NODE` belongs to the imported `ECU-INSTANCE`, its `TP-ADDRESS` is considered the *local* address of the `FLEXRAY-AR-TP-CONNECTION`, and the receiver `TP-ADDRESS` is considered as its *remote* `TP-ADDRESS`. If the sender `FLEXRAY-AR-TP-NODE` does not belong to the imported `ECU-INSTANCE`, its `TP-ADDRESS` is considered as *remote* and the receiver `TP-ADDRESS` is considered as *local*. All `FLEXRAY-AR-TP-CONNECTION` elements of a `FLEXRAY-AR-TP-CHANNEL` that are processed by the imported `ECU-INSTANCE` are collected according to their *local* and *remote* address pair. For each unique *local/remote* address pair, one `FlexrayArTpConnection` container is created. `FrArTpLa` is set to `TP-ADDRESS/TP-ADDRESS` of the *local* `TP-ADDRESS`. `FrArTpMultRec` is set to `true` if the `FLEXRAY-AR-TP-CONNECTION` references a `TP-ADDRESS` via `MULTICAST-REF`, otherwise to `false`. `FrArTpRa` is set to `TP-ADDRESS/TP-ADDRESS` of the *remote* `TP-ADDRESS`.
`FrTpMultipleConfig/FrArTpChannel/FrArTpConnection/FrArTpRxSdu`	If the imported `ECU-INSTANCE` receives any PDU which the `FLEXRAY-AR-TP-CONNECTION` references either via `DIRECT-TP-SDU-REF` or via `REVERSED-TP-SDU-REF`, an `FrArTpRxSdu` sub container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU.

Configuration parameters	Mapping description
	`FrArTpRxSduRef` is configured to reference the corresponding container of the PDU in the `EcuC` module.
`FrTpMultipleConfig/FrArTpChannel/FrArTpConnection/FrArTpTxSdu`	If the imported `ECU-INSTANCE` sends any PDU which the `FLEXRAY-AR-TP-CONNECTION` references either via `DIRECT-TP-SDU-REF` or via `REVERSED-TP-SDU-REF`, an `FrArTpTxSdu` sub container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU. `FrArTpTxSduRef` is configured to reference the corresponding container of the PDU in the `EcuC` module.
`FrTpMultipleConfig/FrArTpChannel/FrArTpPdu`	For each `N-PDU` which the `FLEXRAY-AR-TP-CHANNEL` references via `N-PDU-REFS/N-PDU-REF` and which the imported `ECU-INSTANCE` either receives or sends, one `FrArTpPdu` container is created. Section 3.3.5, "Collection of N-PDU elements" describes the cases in which an `FrArTpPdu` is considered as received or sent by the imported `ECU-INSTANCE`. The name of the container is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `N-PDU`. `FrArTpPduRef` is configured to reference the corresponding container of the `N-PDU` in the `EcuC` module. `FrArTpPduDirection` is set to `FRARTP_RX` if the imported `ECU-INSTANCE` receives the `N-PDU`. If the imported `ECU-INSTANCE` sends the `N-PDU`, `FrArTpPduDirection` is set to `FRARTP_TX`.

3.4.24. FrIf

Configuration parameters	Mapping description
`FrIfConfig/FrIfCluster`	For every `FLEXRAY-CLUSTER` to which the imported `ECU-INSTANCE` is connected a `FrIfCluster` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `FLEXRAY-CLUSTER`. `FrIfGdWakeupRxIdle` is set to `WAKEUP-RX-IDLE`. `FrIfGdWakeupRxLow` is set to `WAKEUP-RX-LOW`. `FrIfGdWakeupRxWindow` is set to `WAKEUP-RX-WINDOW`.

Configuration parameters	Mapping description
	`FrIfGdWakeupTxIdle` is set to `WAKEUP-TX-IDLE`.

`FrIfGPayloadLengthStatic` is set to `PAYLOAD-LENGTH-STATIC`.

`FrIfGNetworkManagementVectorLength` is set to `NETWORK-MANAGE-MENT-VECTOR-LENGTH`.

`FrIfGdCycle` is set to `CYCLE`.

`FrIfGdCasRxLowMax` is set to `CAS-RX-LOW-MAX`.

`FrIfGNumberOfStaticSlots` is set to `NUMBER-OF-STATIC-SLOTS`.

`FrIfGColdStartAttempts` is set to `COLD-START-ATTEMPTS`.

`FrIfGdActionPointOffset` is set to `ACTION-POINT-OFFSET`.

`FrIfGdDynamicSlotIdlePhase` is set to `DYNAMIC-SLOT-IDLE-PHASE`.

`FrIfGdMinislot` is set to `MINISLOT-DURATION`.

`FrIfGdMiniSlotActionPointOffset` is set to `MINISLOT-AC-TION-POINT-OFFSET`.

`FrIfGdNit` is set to `NETWORK-IDLE-TIME`.

`FrIfGdStaticSlot` is set to `STATIC-SLOT-DURATION`.

`FrIfGdSymbolWindow` is set to `SYMBOL-WINDOW`.

`FrIfGdTSSTransmitter` is set to `TRANSMISSION-START-SEQUENCE-DU-RATION`.

`FrIfGListenNoise` is set to `LISTEN-NOISE`.

`FrIfGMacroPerCycle` is set to `MACRO-PER-CYCLE`.

`FrIfGdMacrotick` is set to `MACROTICK-DURATION`.

`FrIfGMaxWithoutClockCorrectPassive` is set to `MAX-WITHOUT-CLOCK-CORRECTION-PASSIVE`.

`FrIfGMaxWithoutClockCorrectFatal` is set to `MAX-WITHOUT-CLOCK-CORRECTION-FATAL`.

`FrIfGNumberOfMinislots` is set to `NUMBER-OF-MINISLOTS`. |

Configuration parameters	Mapping description
	`FrIfGChannels` is set depending on the `FLEXRAY-CLUSTER`'s `PHYSICAL-CHANNEL` elements to which the imported `ECU-INSTANCE` is connected: ▶ `FR_CHANNEL_A` if connected to `CHANNEL-A`. ▶ `FR_CHANNEL_B` if connected to `CHANNEL-B`. ▶ `FR_CHANNEL_AB` if connected to both `CHANNEL-A` and `CHANNEL-B`. `FrIfGdSampleClockPeriod` is set depending on the value of `SAMPLE-CLOCK-PERIOD`[s]: ▶ `T50NS` for values >= 0.000000050. ▶ `T25S` for values >= 0.000000025 and < 0.000000050. ▶ `T12_5NS` for values >= 0.0000000125 and < 0.000000025. ▶ For values < 0.0000000125 `FrIfGdSampleClockPeriod` is set to `T12_5NS` and a warning is reported. `FrIfGdBit` is set depending on the value of `BIT` [s], `BAUDRATE` [bit/s] or `SPEED` [kbit/s] according to the following formulas: ▶ $BIT * 10^9$. ▶ 10^9 / `BAUDRATE`[bit/s] if `BIT` is not available. ▶ 10^6 / `SPEED`[kbit/s] if neither `BIT` nor `BAUDRATE` are available. `FrIfGdBit` is then configured depending on the calculation result: ▶ `T100NS` for 100. ▶ `T200NS` for 200. ▶ `T400NS` for 400. ▶ For all other values `FrIfGdBit` is set to `T100NS` and a warning is reported. `FrIfDetectNITError` is set to `DETECT-NIT-ERROR`. `FrIfGCycleCountMax` is set to `CYCLE-COUNT-MAX`. `FrIfGdIgnoreAfterTx` is set to `IGNORE-AFTER-TX`. `FrIfGdSymbolWindowActionPointOffset` is set to `SYMBOL-WINDOW-ACTION-POINT-OFFSET`. `FrIfGdWakeupTxActive` is set to `WAKEUP-TX-ACTIVE`.

Configuration parameters	Mapping description
	`FrIfGSyncFrameIDCountMax` is set to `SYNC-FRAME-ID-COUNT-MAX`. `FrIfSafetyMargin` is set to `SAFETY-MARGIN`.
`FrIfConfig/FrIfCluster/FrIfController`	For every `FLEXRAY-COMMUNICATION-CONTROLLER` connected to the `FLEXRAY-CLUSTER` and the imported `ECU-INSTANCE` a `FrIfController` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `FLEXRAY-COMMUNICATION-CONTROLLER`. `FrIfFrCtrlRef` references the corresponding container in the `Fr` module configuration.

`FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering`	For every `FLEXRAY-FRAME-TRIGGERING` which contains via the referenced `FLEXRAY-FRAME` and `I-PDU-TO-FRAME-MAPPING` elements PDUs sent or received by the imported `ECU-INSTANCE` a `FrIfFrameTriggering` container is created. The container name is `t<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `FLEXRAY-FRAME`.
	
	`FrIfChannel` is set depending on which `PHYSICAL-CHANNEL` the `FLEXRAY-FRAME-TRIGGERING` is sent/received:
	
	▶ `FRIF_CHANNEL_A` if sent/received on `CHANNEL-A`.
	
	▶ `FRIF_CHANNEL_B` if sent/received on `CHANNEL-B`.
	
	▶ `FRIF_CHANNEL_AB` if two `FLEXRAY-FRAME-TRIGGERING` elements exist, one on `CHANNEL-A`, the other on `CHANNEL-B`, which reference the same `FLEXRAY-FRAME`. In this case the following conditions must hold as well:
	
	   ▶ `BASE-CYCLE`, `CYCLE-REPETITION`, and `SLOT-ID` must be identical.
	
	   ▶ Both `FLEXRAY-FRAME-TRIGGERING` must have the same transmission direction (sent/received).
	
	   ▶ The `SLOT-ID` must be `<=` `NUMBER-OF-STATIC-SLOTS` as defined in the `FLEXRAY-CLUSTER`.
	
	   ▶ `ALLOW-DYNAMIC-L-SDU-LENGTH`, `PAYLOAD-PREAMBLE-INDICATOR`, and `MESSAGE-ID` must be identical or undefined for both `FLEXRAY-FRAME-TRIGGERING` elements.
	If the conditions hold, only a single `FrIfFrameTriggering` container is created.
	
	`FrIfFrameStructureRef` references the container created for the `FLEXRAY-FRAME` referenced by the `FLEXRAY-FRAME-TRIGGERING` (see `FrIfConfig/FrIfFrameStructure`).

Configuration parameters	Mapping description
	`FrIfBaseCyle`, `FrIfCycleRepetition` are set to `BASE-CYCLE` and `CYCLE-REPETITION` of the `FLEXRAY-FRAME-TRIGGERING` elements `FLEXRAY-ABSOLUTELY-SCHEDULED-TIMING` respectively, where values of multiple `FLEXRAY-ABSOLUTELY-SCHEDULED-TIMING` elements are normalized (i.e. `BASE-CYCLE-0/CYCLE-REPETITION-2` and `BASE-CY-CLE-1/CYCLE-REPETITION-2` are normalized to `BASE-CYCLE-0/CY-CLE-REPETITION-1`). `FrIfLSduLength` is set to `FRAME-LENGTH` of the `FLEXRAY-FRAME` `FrIfSlotId` is set to `SLOT-ID`. `FrIfPayloadPreamble` is set to `PAYLOAD-PREAMBLE-INDICATOR`. `FrIfAllowDynamicLSduLength` is set to `ALLOW-DYNAMIC-L-SDU-LENGTH`. `FrIfMessageId` is set to `MESSAGE-ID`.
`FrIfConfig/FrIf-FrameStructure`	For every `FRAME` instance sent/received by the imported `ECU-INSTANCE` a `FrIfFrameStructure` container is created. The container name is `f<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `FLEXRAY-FRAME`.
`FrIfConfig/FrIf-FrameStructure/FrIfPdusInFrame`	For every PDU sent/received by the imported `ECU-INSTANCE` and contained by the `FLEXRAY-FRAME` for which the `FrIfFrameStructure` container has been created, a `FrIfPdusInFrame` container is created. The container name is `p<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU. `FrIfPduOffset` is set to `START-POSITION` of the `PDU-TO-FRAME-MAPPING`, divided by 8. `FrIfPduUpdateBitOffset` is set to `UPDATE-INDICATION-BIT-POSITION` of the `PDU-TO-FRAME-MAPPING`, converted to `monotone` representation. `FrIfPduRef` references the container created for this PDU under `FrIfConfig/FrIfPdu`.
`FrIfConfig/FrIfPdu`	For every PDU instance (see Section 3.3.1.3, "Instance handling") sent, received or routed (see Section 3.3.6, "PDU routing") by the imported `ECU-INSTANCE`, a `FrIfPdu` container is created. The container name is `p<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU belonging to the PDU instance.

Configuration parameters	Mapping description
	`FrIfPduDirection` is set to `FrIfRxPdu` if received by the imported `ECU-INSTANCE`, or to `FrIfTxPdu` if sent by the imported `ECU-INSTANCE`. The name of `FrIfRxPdu`/`FrIfTxPdu` is set to `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU belonging to the PDU instance. `FrIfRxPduRef`/`FrIfTxPduRef` references the corresponding container in the `EcuC` module configuration. `FrIfUserTxUL`/`FrIfUserRxIndicationUL` is set depending on the type of PDU: ▶ `FR_NM` for `NM-PDU` elements. ▶ `FR_TP` for `N-PDU` elements that are not routed via gateway. ▶ `PDUR` for `I-SIGNAL-PDU` elements, `MULTIPLEXED-I-PDU` elements, `CONTAINER-I-PDU` elements, `GENERAL-PURPOSE-I-PDU` elements, `DCM-I-PDU` elements, for `USER-DEFINED-I-PDU` elements, which either have a `CATEGORY` other than `XCP` or are routed, and for `N-PDU` elements that are routed (see [Section 3.3.6, "PDU routing"](#)). ▶ `XCP` for `XCP-PDU` elements. ▶ `CDD` for `USER-DEFINED-I-PDU` elements that have their `CATEGORY` set to `XCP` and for all other PDU types.

3.4.25. FrNm

Configuration parameters	Mapping description
`FrNmGlobalConfig/FrNmGlobalProperties`	The following parameters are set using the first `FLEXRAY-NM-ECU` of all `FLEXRAY-NM-CLUSTER` elements belonging to the imported `ECU-INSTANCE`: `FrNmMainAcrossFrCycle` is set to `NM-MAIN-FUNCTION-ACROSS-FR-CYCLE`.
`FrNmGlobalConfig/FrNmGlobalFeatures`	`FrNmComUserDataSupport` is set to `true` if `FrNmRxUserDataPduRef` or `FrNmTxUserDataPduRef` is set for any `FrNmChannelConfig/FrNmChannelIdentifiers`, or if any `FLEXRAY-NM-CLUSTER` linked to the imported `ECU-INSTANCE` has its `NM-PNC-PARTICIPATION` either not defined or set to `true`. Otherwise `FrNmComUserDataSupport` is set to `false`.

Configuration parameters	Mapping description
	`FrNmPnResetTime` is set to `PN-RESET-TIME` of the configured `ECU-INSTANCE`. The following parameters are set using the first `NM-ECU` of all `FLEXRAY-NM-CLUSTER` elements belonging to the imported `ECU-INSTANCE`: `FrNmBusSynchronizationEnabled` is set to `NM-BUS-SYNCHRONIZA-TION-ENABLED`. `FrNmDualChannelPduEnable` is set to `NM-MULTIPLE-CHANNELS-ENABLED`. `FrNmPassiveModeEnabled`: see [Section 3.4.4, "CanNm"](#), `CanNmPassiveModeEnabled`. `FrNmPduRxIndicationEnabled` is set to `NM-PDU-RX-INDICATION-EN-ABLED`. `FrNmRemoteSleepIndicationEnabled` is set to `NM-REMOTE-SLEEP-IND-ENABLED`. `FrNmStateChangeIndicationEnabled` is set to `NM-STATE-CHANGE-IND-ENABLED`. `FrNmUserDataEnabled` is set to `NM-USER-DATA-ENABLED`. `FrNmRepeatMessageBitEnabled` is set to `NM-REPEAT-MESSAGE-BIT-EN-ABLE`. If inconsistencies are detected among parameters of multiple `NM-ECU` elements, a warning is reported. `FrNmHwVoteEnable` is set to `NM-HW-VOTE-ENABLED` of the `FLEXRAY-NM-ECU` belonging to the `NM-ECU`. `FrNmControlBitVectorEnabled` is set to `NM-CONTROL-BIT-VECTOR-EN-ABLED` of the first `FLEXRAY-NM-CLUSTER-COUPLING` belonging to the imported `ECU-INSTANCE`.
`FrNmChannelConfig/FrNmChannel`	For every `FLEXRAY-NM-CLUSTER` which belongs to the imported `ECU-INSTANCE`, a `FrNmChannel` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `FLEXRAY-NM-CLUSTER`.

Configuration parameters	Mapping description
	A `FLEXRAY-NM-CLUSTER` belongs to the imported `ECU-INSTANCE` if at least one of the `FLEXRAY-NM-CLUSTER`'s `FLEXRAY-NM-NODE` elements references a `FLEXRAY-COMMUNICATION-CONTROLLER` of this `ECU-INSTANCE`. The `NM-ECU` used for configuring some of the `FrNmChannel` parameters is the `NM-ECU` which the first `FLEXRAY-NM-NODE` references via `NM-IF-ECU-REF`.
`FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers`	`FrNmNodeDetectionEnabled` is set to `NM-CLUSTER/NM-NODE-DETECTION-ENABLED`, or to `NM-ECU/NM-NODE-DETECTION-ENABLED` if `NM-CLUSTER/NM-NODE-DETECTION-ENABLED` is not available. `FrNmSourceNodeIdentifierEnabled` is set to `NM-CLUSTER/NM-NODE-ID-ENABLED`, or to `NM-ECU/NM-NODE-ID-ENABLED` if `NM-CLUSTER/NM-NODE-ID-ENABLED` is not available. `FrNmControlBitVectorActive` is set to `NM-CONTROL-BIT-VECTOR-ACTIVE`. `FrNmRepeatMessageBitActive` is set to `NM-REPEAT-MESSAGE-BIT-ACTIVE`. `FrNmSynchronizationPointEnabled` is set to `NM-SYNCHRONIZING-NETWORK`. `FrNmChannelHandle` references the `FrIfCluster` container in the FrIf module configuration, which has been created for the `FLEXRAY-CLUSTER` referenced via `COMMUNICATION-CLUSTER-REF`. `FrNmNodeId` is set to `NM-NODE-ID` of the first `FLEXRAY-NM-NODE` which belongs to the `FLEXRAY-NM-CLUSTER` and at the same time belongs to the `ECU-INSTANCE`. A `FLEXRAY-NM-NODE` belongs to an `ECU-INSTANCE` if it references a `FLEXRAY-COMMUNICATION-CONTROLLER` of this `ECU-INSTANCE`. `FrNmComMNetworkHandleRef` references the `ComMChannel` container that is created for the `COMMUNICATION-CLUSTER` referenced in `COMMUNICATION-CLUSTER-REF`. `FrNmPduScheduleVariant` is set depending on the value of `NM-SCHEDULE-VARIANT` of the first `FLEXRAY-NM-CLUSTER-COUPLING` belonging to the `FLEXRAY-NM-CLUSTER`: ▶ `FRNM_PDU_SCHEDULE_VARIANT_<x>` is set to `SCHEDULE-VARIANT-<x>` where x lies within [1..7].

Configuration parameters	Mapping description
`FrNmChannelConfig/FrNmChannel/FrNmChannelTiming`	`FrNmDataCycle` is set to `FRNM_CYCLE_VALUE_<value of NM-DATA-CYCLE>`. If `NM-DATA-CYCLE` is not one of [1,2,4,8,18,32,64], a warning is reported and `FrNmDataCycle` is not configured. `FrNmRepetitionCycle` is set depending on the value of `NM-REPETITION-CYCLE`, see `FrNmDataCycle` for how values are mapped. `FrNmVotingCycle` is set depending on the value of `NM-VOTING-CYCLE`, see `FrNmDataCycle` for how values are mapped. `FrNmMainFunctionPeriod` is set to `NM-MAIN-FUNCTION-PERIOD`. `FrNmMsgTimeoutTime` is set to `NM-MESSAGE-TIMEOUT-TIME`. `FrNmNodeDetectionLock` is set to `NM-DETECTION-LOCK`. `FrNmReadySleepCnt` is calculated by the formula `((NM-READY-SLEEP-TIME/CYCLE)/NM-REPETITION-CYCLE) - 1`. `NM-READY-SLEEP-TIME` is taken from the first `FLEXRAY-COMMUNICATION-CONNECTOR` connecting the `COMMUNICATION-CONTROLLER` of the first `FLEXRAY-NM-NODE` to a `PHYSICAL-CHANNEL` of the `FLEXRAY-NM-CLUSTER`'s `FLEXRAY-COMMUNICATION-CLUSTER`. A warning is reported if multiple different `NM-READY-SLEEP-TIME` values are found. If the calculation yields a non-integral value, a warning is reported and `FrNmReadySleepCnt` is not set. If `NM-READY-SLEEP-TIME` is not available, `FrNmReadySleepCnt` is set to `NM-READY-SLEEP-COUNT`. `FrNmRemoteSleepIndTime` is set to `NM-REMOTE-SLEEP-INDICATION-TIME`. `FrNmRepeatMessageTime` is set to `NM-REPEAT-MESSAGE-TIME`.
`FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmRxPdu`	For every `NM-PDU` referenced via `RX-NM-PDU-REF` by the first `FLEXRAY-NM-NODE` belonging to the `FLEXRAY-NM-CLUSTER` and the imported `ECU-INSTANCE`, a `FrNmRxPdu` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `NM-PDU`. If either none or more than two `NM-PDU` elements are referenced via `RX-NM-PDU-REF` a warning is reported. `FrNmRxPduContainsData` is set to `true` if the `NM-PDU` contains `I-SIGNAL-TO-I-PDU-MAPPING` elements referencing an `I-SIGNAL` or an `I-SIG-`

Configuration parameters	Mapping description
	NAL-GROUP, or if NM-DATA-INFORMATION is set to true. In all other cases FrNmRxPduContainsData is set to false. FrNmRxPduPduContainsVote is set to NM-VOTE-INFORMATION. FrNmRxPduRef references the corresponding container in the EcuC module configuration.
FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmUserDataRxPdu	The list of all NM-PDU elements that are referenced via RX-NM-PDU-REF by the first FLEXRAY-NM-NODE that belongs to the FLEXRAY-NM-CLUSTER and the imported ECU-INSTANCE constitutes the received NM-PDU elements of the FLEXRAY-NM-CLUSTER. The FrNmUserDataRxPdu container is created if an NmUserDataPdu container has been created in the EcuC module configuration for at least one of these NM-PDU elements, see Section 3.4.14, "EcuC". The name of FrNmUserDataRxPdu is set to <PREFIX><name><INSTSUFFIX>_NmComUserData, where <name> is the SHORT-NAME of the NM-PDU. FrNmRxUserDataPduRef references the NmUserDataPdu container in the EcuC module which has been created for the NM-PDU. If more than one NM-PDU contains I-SIGNAL-TO-I-PDU-MAPPING elements referencing an I-SIGNAL or an I-SIGNAL-GROUP, the first NM-PDU is used and a warning is reported.
FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmTxPdu	For every NM-PDU referenced via TX-NM-PDU-REF by the first FLEXRAY-NM-NODE belonging to the FLEXRAY-NM-CLUSTER and the imported ECU-INSTANCE, a FrNmTxPdu container is created. The container name is <PREFIX><name><INSTSUFFIX>, where <name> is the SHORT-NAME of the NM-PDU. If more than two NM-PDU elements are referenced via TX-NM-PDU-REF a warning is reported. FrNmTxPduContainsData is set to true if the NM-PDU contains an I-SIGNAL-TO-I-PDU-MAPPING referencing an I-SIGNAL or an I-SIGNAL-GROUP, or if NM-DATA-INFORMATION is set to true. In all other cases FrNmTxPduContainsData is set to false. FrNmTxPduPduContainsVote is set to NM-VOTE-INFORMATION. FrNmTxPduRef references the corresponding container in the EcuC module configuration.

Configuration parameters	Mapping description
`FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmUserDataTxPdu`	See `FrNmUserDataRxPdu` (`NM-PDU` elements referenced via `TX-NM-PDU-REF` are used for `FrNmUserDataTxPdu`).

For the configuration of PNC-related parameters see [Section 3.4.4, "CanNm"](#). The parameters and configuration containers obtain the prefix `FrNmPn`. The `PNC-FILTER-DATA-MASK` values of all `FLEXRAY-COMMUNICATION-CONNECTOR` elements that belong to the imported `ECU-INSTANCE` are taken as input for the calculation of `FrNmPnFilterMaskByte`.

3.4.26. FrSM

Configuration parameters	Mapping description
`FrSMConfig/FrSMCluster`	For every `FLEXRAY-CLUSTER` which belongs to the imported `ECU-INSTANCE`, a `FrSMCluster` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `FLEXRAY-CLUSTER`. `FrSMComMNetworkHandleRef` references the `ComMChannel` container in the `ComM` module configuration, which has been created for the `FLEXRAY-CLUSTER`. `FrSMFrIfClusterRef` references the `FrIfCluster` container in the `FrIf` module configuration, which has been created for the `FLEXRAY-CLUSTER`.

3.4.27. FrTp

Configuration parameters	Mapping description
`FrTpGeneral`	The following parameters are set using the first `FLEXRAY-TP-ECU` belonging to the imported `ECU-INSTANCE`: `FrTpMainFuncCycle` is set to `CYCLE-TIME-MAIN-FUNCTION`. `FrTpFullDuplexEnable` is set to `FULL-DUPLEX-ENABLED`. `FrTpTransmitCancellation` is set to `CANCELLATION`.

Configuration parameters	Mapping description
	If inconsistencies are detected among parameters of multiple `FLEXRAY-TP-ECU` elements of the imported `ECU-INSTANCE`, a warning is reported.
`FrTpMultipleConfig/FrTpConnection`	For each `FLEXRAY-TP-CONNECTION` which belongs to the imported `ECU-INSTANCE`, one `FrTpConnection` container is created, named `FrTpConnection_<suffix>`, where `<suffix>` is a zero-based index. A `FLEXRAY-TP-CONNECTION` belongs to the imported `ECU-INSTANCE` if at least one transmitting or receiving `FLEXRAY-TP-NODE` of this `FLEXRAY-TP-CONNECTION` references a `COMMUNICATION-CONNECTOR` which is also referenced by the imported `ECU-INSTANCE`. Two `FLEXRAY-TP-CONNECTION` elements `<FrTp1>` and `<FrTp2>` are configured as one single `FrTpConnection` container if they both belong to the imported `ECU-INSTANCE` and the following conditions hold: ▶ `<FrTp1>` and `<FrTp2>` both refer to exactly one `FLEXRAY-TP-NODE` via `RECEIVER-REFS` ▶ The single `FLEXRAY-TP-NODE` in `RECEIVER-REFS` of `<FrTp1>` is the same as the one referenced in `TRANSMITTER-REF` of `<FrTp2>` and vice versa. ▶ Both `<FrTp1>` and `<FrTp2>` refer to the exactly one PDU, either via `DIRECT-TP-SDU-REF`, or via `REVERSED-TP-SDU-REF` ▶ Neither `<FrTp1>` nor `<FrTp2>` refers to a `TP-ADDRESS` via `MULTICAST-REF` ▶ Both `<FrTp1>` and `<FrTp2>` refer to the same `FLEXRAY-TP-CONNECTION-CONTROL` via `TP-CONNECTION-CONTROL-REF` ▶ Both `<FrTp1>` and `<FrTp2>` either have the same `BANDWIDTH-LIMITATION` value configured, or none of `<FrTp1>` and `<FrTp2>` has that value configured In that case, the configuration values of `<FrTp1>` are used to configure the `FrTpConnection` container. Additionally, the PDU of `<FrTp2>` is configured in the container as well, either in `FrTpRxSdu` or in `FrTpTxSdu`. `FrTpConCtrlRef` references the container created for the `FLEXRAY-TP-CONNECTION-CONTROL` referenced (see `FrTpMultipleConfig/FrTpConnectionControl`). `FrTpRxPduPoolRef` references the container created for the Rx `N-PDU` elements referenced via `FLEXRAY-TP-PDU-POOL` elements (see `FrTpMultipleConfig/FrTpRxPduPool`).

Configuration parameters	Mapping description
	`FrTpTxPduPoolRef` references the container created for the Tx `N-PDU` elements referenced via `FLEXRAY-TP-PDU-POOL` elements (see `FrTpMultiple-Config/FrTpTxPduPool`). `FrTpRxSdu`: this container is created if the PDU referenced by the `FLEXRAY-TP-CONNECTION` via `DIRECT-TP-SDU-REF` is received or routed (see [Section 3.3.6, "PDU routing"](#)) by the imported `ECU-INSTANCE`, or if this PDU is referenced via `REVERSED-TP-SDU-REF` and is sent or routed by the imported `ECU-INSTANCE`. The name of `FrTpRxSdu` is set to `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU. `FrTpRxSdu/FrTpRxSduRef` references the corresponding container in the `EcuC` module configuration. `FrTpTxSdu`: this container is created if the PDU referenced by the `FLEXRAY-TP-CONNECTION` via `DIRECT-TP-SDU-REF` is sent or routed (see [Section 3.3.6, "PDU routing"](#)) by the imported `ECU-INSTANCE`, or if this PDU is referenced via `REVERSED-TP-SDU-REF` and is received or routed by the imported `ECU-INSTANCE`. The name of `FrTpTxSdu` is set to `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU. `FrTpTxSdu/FrTpTxSduRef` references the corresponding container in the `EcuC` module configuration. `FrTpBandwidthLimitation` is set to `BANDWIDTH-LIMITATION`. `FrTpMultipleReceiverCon` is set to `true` if a valid `TP-ADDRESS` is referenced by the `FLEXRAY-TP-CONNECTION` via `MULTICAST-REF`. Otherwise `FrTpMultipleReceiverCon` is set to `false`. `FrTpLa`: if the `FLEXRAY-TP-NODE` referenced via `TRANSMITTER-REF` belongs to the imported `ECU-INSTANCE`, `FrTpLa` is set to the `TP-ADDRESS` referenced by this `FLEXRAY-TP-NODE`. Otherwise `FrTpLa` is set to the `TP-ADDRESS` referenced by the `FLEXRAY-TP-CONNECTION` via `MULTICAST-REF`, or, if this is not available, to the `TP-ADDRESS` of the first `FLEXRAY-TP-NODE` referenced via `RECEIVER-REF`. `FrTpRa`: if the `FLEXRAY-TP-NODE` referenced via `TRANSMITTER-REF` does not belong to the imported `ECU-INSTANCE`, `FrTpRa` is set to the `TP-ADDRESS` referenced by this `FLEXRAY-TP-NODE`. Otherwise `FrTpRa` is set to the `TP-ADDRESS` referenced by the `FLEXRAY-TP-CONNECTION` via `MULTICAST-REF`, or,

Configuration parameters	Mapping description
	if this is not available, to the `TP-ADDRESS` of the first `FLEXRAY-TP-NODE` referenced via `RECEIVER-REF`.
`FrTpMultipleConfig/FrTpConnectionControl`	For each `FLEXRAY-TP-CONNECTION-CONTROL` referenced by an imported `FLEXRAY-TP-CONNECTION` a `FrTpConnectionControl` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `FLEXRAY-TP-CONNECTION-CONTROL`. `FrTpAckType` is set depending on the value of `ACK-TYPE`: ► `FRTP_ACK_WITH_RT` for `ACK-WITH-RT`. ► `FRTP_NO` for `NO-ACK`. `FrTpMaxAr` is set to `MAX-AR`. `FrTpMaxAs` is set to `MAX-AS`. `FrTpMaxBufferSize` is set to `MAX-BUFFER-SIZE`. `FrTpMaxFCWait` is set to `MAX-FC-WAIT`. `FrTpMaxFrIf` is set to `MAX-FR-IF`. `FrTpMaxRn` is set to `MAX-RETRIES`. `FrTpTimeBr` is set to `TIME-BR`. `FrTpTimeBuffer` is set to `TIME-BUFFER`. `FrTpTimeFrIf` is set to `TIME-FR-IF`. `FrTpTimeoutAr` is set to `TIMEOUT-AR`. `FrTpTimeoutAs` is set to `TIMEOUT-AS`. `FrTpTimeoutBs` is set to `TIMEOUT-BS`. `FrTpTimeoutCr` is set to `TIMEOUT-CR`. `FrTpSCexp` is set to `SEPARATION-CYCLE-EXPONENT`. `FrTpMaxNbrOfNPduPerCycle` is set to `MAX-NUMBER-OF-NPDU-PER-CYCLE`.
`FrTpMultipleConfig/FrTpRxPduPool`	For each `FLEXRAY-TP-PDU-POOL` that is referenced by a sent `FLEXRAY-TP-CONNECTION` via `RX-PDU-POOL-REF` or that is referenced by a received `FLEXRAY-TP-CONNECTION` via `TX-PDU-POOL-REF`, and which contains only references to `N-PDU` elements received by the imported `ECU-INSTANCE`, an

Configuration parameters	Mapping description
	`FrTpRxPduPool` container is created. A `FLEXRAY-TP-CONNECTION` is considered sent/received if it belongs to the imported `ECU-INSTANCE` and if it references a PDU via `DIRECT-TP-SDU-REF` which is sent/received by this `ECU-INSTANCE`. The container name is `<name>_Rx`, where `<name>` is the `SHORT-NAME` of the `FLEXRAY-TP-PDU-POOL`. If a `FLEXRAY-TP-PDU-POOL` that is referenced by a sent `FLEXRAY-TP-CONNECTION` via `RX-PDU-POOL-REF` or that is referenced by a received `FLEXRAY-TP-CONNECTION` via `TX-PDU-POOL-REF`, contains any references to PDUs sent by the imported `ECU-INSTANCE`, the following algorithm is used to create `FrTpRxPduPool` containers: For each unique set of `N-PDU` elements received by the imported `ECU-INSTANCE` and referenced by `FLEXRAY-TP-PDU-POOL` elements of a `FLEXRAY-TP-CONNECTION` (referenced via both `RX-PDU-POOL-REF` and `TX-PDU-POOL-REF`), a `FrTpRxPduPool` container named `PduPool_<suffix>`) is created, where `<suffix>` is a zero-based index. For all referenced `N-PDU` elements the parameter `FrTpRxPduRef` is set (using a reference to corresponding container in the `EcuC` module).
`FrTpMultipleConfig/FrTpTxPduPool`	For each `FLEXRAY-TP-PDU-POOL` that is referenced by a sent `FLEXRAY-TP-CONNECTION` via `TX-PDU-POOL-REF` or that is referenced by a received `FLEXRAY-TP-CONNECTION` via `RX-PDU-POOL-REF`, and which contains only references to `N-PDU` elements sent by the imported `ECU-INSTANCE`, an `FrTpTxPduPool` container is created. The container name is `<name>_Tx`, where `<name>` is the `SHORT-NAME` of the `FLEXRAY-TP-PDU-POOL`. If a `FLEXRAY-TP-PDU-POOL` that is referenced by a sent `FLEXRAY-TP-CONNECTION` via `TX-PDU-POOL-REF` or that is referenced by a received `FLEXRAY-TP-CONNECTION` via `RX-PDU-POOL-REF`, contains any references to PDUs received by the imported `ECU-INSTANCE`, the following algorithm is used to create `FrTpRxPduPool` containers: For each unique set of `N-PDU` elements sent by the imported `ECU-INSTANCE` and referenced by `FLEXRAY-TP-PDU-POOL` elements of a `FLEXRAY-TP-CONNECTION` (referenced via both `RX-PDU-POOL-REF` and `TX-PDU-POOL-REF`), a `FrTpTxPduPool` container named `PduPool_<suffix>`) is created, where `<suffix>` is a zero-based index. For all referenced `N-PDU` elements the parameter `FrTpTxPduRef` is set (using a reference to corresponding container in the `EcuC` module).

3.4.28. FrTSyn

Configuration parameters	Mapping description
`FrTSynGlobalTimeDomain`	One `FrTSynGlobalTimeDomain` container is created for each `GLOBAL-TIME-DOMAIN` that contains a `GLOBAL-TIME-FR-MASTER` or a `GLOBAL-TIME-FR-SLAVE` element that references a `FLEXRAY-COMMUNICATION-CONNECTOR` of the imported `ECU-INSTANCE`. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `GLOBAL-TIME-DOMAIN`. `FrTSynGlobalTimeDomainId` is set to `DOMAIN-ID`. `FrTSynGlobalTimeNetworkSegmentId` is set to `NETWORK-SEGMENT-ID/NETWORK-SEGMENT-ID`. `FrTSynSynchronizedTimeBaseRef` references the corresponding `StbMSynchronizedTimeBase` container in the `StbM` module. If the `GLOBAL-TIME-DOMAIN` is a subdomain of a parent `GLOBAL-TIME-DOMAIN`, `FrTSynSynchronizedTimeBaseRef` references the `StbMSynchronizedTimeBase` container that has been created for the parent `GLOBAL-TIME-DOMAIN`. For further information about the configuration of the `StbM` module, see [Section 3.4.45, "StbM"]().
`FrTSynGlobalTimeDomain/FrTSynGlobalTimeOfsDataIDList`, `FrTSynGlobalTimeDomain/FrTSynGlobalTimeSyncDataIDList`	The `FR-GLOBAL-TIME-DOMAIN-PROPS` element of the FlexRay `GLOBAL-TIME-DOMAIN` entity is used to retrieve the following subelements to configure container lists: ▶ `OFS-DATA-ID-LIST` ▶ `SYNC-DATA-ID-LIST` The configured container lists are the following: ▶ `FrTSynGlobalTimeDomain/FrTSynGlobalTimeOfsDataIDList/FrTSynGlobalTimeOfsDataIDListElement` ▶ `FrTSynGlobalTimeDomain/FrTSynGlobalTimeSyncDataIDList/FrTSynGlobalTimeSyncDataIDListElement` In each container list, one subcontainer is created per element named `Element_<idx>`, where `<idx>` is the zero-based index of the element within the list. In each of these lists, one index parameter and one value parameter are configured. The index parameter represents the zero-based index of the element in the list. The value parameter represents the actual value.

Configuration parameters	Mapping description		
	ID list	**Index parameter**	**Value parameter**
	`FrTSynGlobalTime-OfsDataIDList`	`FrTSynGlobalTime-OfsDataIDListIndex`	`FrTSynGlobalTime-OfsDataIDListValue`
	`FrTSynGlobal-TimeSyncDataIDList`	`FrTSynGlob-alTimeSync-DataIDListIndex`	`FrTSynGlobal-TimeSyncDataIDList-Value`
`FrTSynGlobalTimeDo-main/FrTSynGlobal-TimeMaster`	If the `GLOBAL-TIME-FR-MASTER` of the `GLOBAL-TIME-DOMAIN` references a `FLEXRAY-COMMUNICATION-CONNECTOR` of the imported `ECU-INSTANCE`, an `FrTSynGlobalTimeMaster` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `GLOBAL-TIME-FR-MASTER`. `FrTSynGlobalTimeTxCrcSecured` is set depending on the value of `GLOB-AL-TIME-FR-MASTER/CRC-SECURED`: `CRC_SUPPORTED` for `CRC-SUPPORTED` and `CRC_NOT_SUPPORTED` for `CRC-NOT-SUPPORTED`. `FrTSynGlobalTimeTxPeriod` is set to `GLOBAL-TIME-FR-MASTER/SYNC-PERIOD`. `FrTSynCyclicMsgResumeTime` is set to `GLOBAL-TIME-FR-MASTER/IMMEDIATE-RESUME-TIME`. `FrTSynGlobalTimeDebounceTime` is set to `GLOBAL-TIME-DOMAIN/DE-BOUNCE-TIME`.		
`FrTSynGlobalTimeDo-main/FrTSynGlobal-TimeMaster/FrTSyn-GlobalTimeMasterPdu`	A `GLOBAL-TIME-DOMAIN` is associated with a PDU if that PDU is a `GENER-AL-PURPOSE-PDU` that has its `CATEGORY` set to `GLOBAL_TIME` and one of the conditions holds: ► The `GLOBAL-TIME-DOMAIN` references the PDU in `GLOBAL-TIME-PDU-REF` ► The `GLOBAL-TIME-DOMAIN` references a `PDU-TRIGGERING` either via `GLOBAL-TIME-PDU-TRIGGERING-REF` or via `PDU-TRIGGERING-REF`, and that `PDU-TRIGGERING` refers to the PDU If the `GLOBAL-TIME-DOMAIN` is associated with a PDU, and if this is the PDU that the configured `ECU-INSTANCE` sends on the `FLEXRAY-CLUSTER` which the `GLOBAL-TIME-DOMAIN` references via `COMMUNICATION-CLUSTER-REF`, one `FrTSynGlobalTimeMasterPdu` container is created. Its name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU.		

Configuration parameters	Mapping description
	`FrTSynGlobalTimePduRef` references the `EcuC` container created for the PDU that is associated with the `GLOBAL-TIME-DOMAIN`.
`FrTSynGlobalTimeDomain/FrTSynGlobalTimeSlave`	If one of the `GLOBAL-TIME-FR-SLAVE` elements of the `GLOBAL-TIME-DOMAIN` references a `FLEXRAY-COMMUNICATION-CONNECTOR` of the imported `ECU-INSTANCE`, an `FrTSynGlobalTimeSlave` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `GLOBAL-TIME-FR-SLAVE`. `FrTSynRxCrcValidated` is set depending on the value of `GLOBAL-TIME-FR-SLAVE/CRC-VALIDATED`: `CRC_VALIDATED` for `CRC-VALIDATED`, `CRC_NOT_-VALIDATED` for `CRC-NOT-VALIDATED`, and `CRC_IGNORED` for `CRC-IGNORED`. `FrTSynGlobalTimeSequenceCounterJumpWidth` is set to `GLOBAL-TIME-FR-SLAVE/SEQUENCE-COUNTER-JUMP-WIDTH`.
`FrTSynGlobalTimeDomain/FrTSynGlobalTimeSlave/FrTSynGlobalTimeSlavePdu`	If the `GLOBAL-TIME-DOMAIN` is associated with a PDU, and if this is the PDU that the configured `ECU-INSTANCE` receives on the `FLEXRAY-CLUSTER` which the `GLOBAL-TIME-DOMAIN` references via `COMMUNICATION-CLUSTER-REF`, one `FrTSynGlobalTimeSlavePdu` container is created. Its name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU. `FrTSynGlobalTimePduRef` references the `EcuC` container created for the PDU that is associated with the `GLOBAL-TIME-DOMAIN`.

3.4.29. IpduM

Configuration parameters	Mapping description
`IpduMGeneral/IpduMHeaderByteOrder`	If the imported `ECU-INSTANCE` sends or receives at least one `CONTAINER-I-PDU`, `IpduMHeaderByteOrder` is configured depending on the `CONTAINER-I-PDU-HEADER-BYTE-ORDER` parameter of the `SYSTEM`: ▶ `IPDUM_LITTLE_ENDIAN` for `MOST-SIGNIFICANT-BYTE-LAST` ▶ `IPDUM_BIG_ENDIAN` for `MOST-SIGNIFICANT-BYTE-FIRST`
`IpduMConfig/IpduMRxPathway`	For every `MULTIPLEXED-I-PDU` which references at least one PDU in its `DYNAMIC-PART-ALTERNATIVE` elements or in its `STATIC-PART`, and which is also received by the imported `ECU-INSTANCE`, an `IpduMRxPathway` container is

Configuration parameters	Mapping description
	created. The container name is `RXP_<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `MULTIPLEXED-I-PDU`.
`IpduMConfig/IpduM-RxPathway/IpduM-RxIndication`	The name of `IpduMRxIndication` is set to `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `MULTIPLEXED-I-PDU`. `IpduMRxIndicationPduRef` references the container created for the `MULTIPLEXED-I-PDU` in the `EcuC` module configuration. `IpduMByteOrder` it set to `LITTLE_ENDIAN`. If `SELECTOR-FIELD-BYTE-ORDER` is set to `MOST-SIGNIFICANT-BYTE-FIRST`, the bit range defined by `SELECTOR-FIELD-START-POSITION` and `SELECTOR-FIELD-LENGTH` is converted to `MOST-SIGNIFICANT-BYTE-LAST`, which may result in multiple bit ranges. Multiple bit ranges in turn result in an error and the termination of the import. In case the bit range crosses byte boundaries, a warning is reported. `IpduMSelectorFieldPosition/IpduMSelectorFieldPosition` is set to the start position of the bit range. `IpduMSelectorFieldPosition/IpduMSelectorFieldLength` is set to the length of the bit range.
`IpduMConfig/IpduM-RxPathway/IpduM-RxIndication/IpduM-RxDynamicPart`	For every `DYNAMIC-PART-ALTERNATIVE` of the `MULTIPLEXED-I-PDU` which references an `I-SIGNAL-I-PDU` that is routed (see Section 3.3.6, "PDU routing") or contains at least one `I-SIGNAL` received by the imported `ECU-INSTANCE`, an `IpduMRxDynamicPart` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `I-SIGNAL-I-PDU`. `IpduMOutgoingDynamicPduRef` references the container created for the `I-SIGNAL-I-PDU` in the `EcuC` module configuration. `IpduMRxSelectorValue` is set to `SELECTOR-FIELD-CODE`.
`IpduMConfig/IpduM-RxPathway/IpduM-RxIndication/IpduM-RxDynamicPart/IpduMSegment`	For every bit range defined by `SEGMENT-POSITION` elements of the `DYNAMIC-PART` an `IpduMSegment` container is created. The container name is `IpduMSegment_<auto incremented number>`. If `SEGMENT-BYTE-ORDER` is set to `MOST-SIGNIFICANT-BYTE-FIRST`, the bit range defined by `SEGMENT-POSITION` and `SEGMENT-LENGTH` is converted to `MOST-SIGNIFICANT-BYTE-LAST`, which may result in multiple bit ranges.

Configuration parameters	Mapping description
	`IpduMSegmentPosition` and `IpduMDestinationBit` are set to the start position of the bit range. `IpduMSegmentLength` is set to the length of the bit range.
`IpduMConfig/IpduM-RxPathway/IpduM-RxIndication/IpduM-RxStaticPart`	If a `STATIC-PART` of the `MULTIPLEXED-I-PDU` exists, which references an `I-SIGNAL-I-PDU` that is routed (see [Section 3.3.6, "PDU routing"](#)) or contains at least one `I-SIGNAL` received by the imported `ECU-INSTANCE`, an `IpduM-RxStaticPart` container is created. The name of `IpduMRxStaticPart` is set to `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `I-SIGNAL-I-PDU`. `IpduMOutgoingStaticPduRef` references the container created for the `I-SIGNAL-I-PDU` in the `EcuC` module configuration. For the configuration of `IpduMSegment` refer to `IpduMRxDynamicPart/IpduMSegment`.
`IpduMConfig/IpduMTxPathway`	For every `MULTIPLEXED-I-PDU` sent by the imported `ECU-INSTANCE`, an `IpduMTxPathway` container is created. The container name is `TXP_-<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `MULTIPLEXED-I-PDU`.
`IpduMConfig/IpduMTxPathway/IpduMTxRequest`	The name of `IpduMTxRequest` is set to `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `MULTIPLEXED-I-PDU`. `IpduMOutgoingPduRef` references the container created for the `MULTIPLEXED-I-PDU` in the `EcuC` module configuration. `IpduMByteOrder` it set to `LITTLE_ENDIAN`. For the configuration of `IpduMSelectorFieldPosition` see `IpduMRxIndication/IpduMSelectorFieldPosition`. `IpduMIPduUnusedAreasDefault` is set to `UNUSED-BIT-PATTERN`. If `UNUSED-BIT-PATTERN` is less than zero or greater than 255, a warning is reported and `IpduMIPduUnusedAreasDefault` remains undefined. `IpduMTxTriggerMode` is set depending on the value of `TRIGGER-MODE`: ▶ `STATIC_PART_TRIGGER` for `STATIC-PART-TRIGGER`. ▶ `DYNAMIC_PART_TRIGGER` for `DYNAMIC-PART-TRIGGER`. ▶ `STATIC_OR_DYNAMIC_PART_TRIGGER` for `STATIC-OR-DYNAMIC-PART-TRIGGER`.

Configuration parameters	Mapping description
	► NONE for NONE. `IpduMInitialDynamicPart` references the `IpduMTxDynamicPart` container which has been created for the first `DYNAMIC-PART-ALTERNATIVE` which has its `INITIAL-DYNAMIC-PART` set to `true`. If multiple `DYNAMIC-PART-ALTERNATIVE` elements have its `INITIAL-DYNAMIC-PART` set to `true`, a warning is issued.
`IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart`	For every `DYNAMIC-PART-ALTERNATIVE` of the `MULTIPLEXED-I-PDU` which references an `I-SIGNAL-I-PDU` that is routed (see Section 3.3.6, "PDU routing") or contains at least one `I-SIGNAL` sent by the imported `ECU-INSTANCE`, an `IpduMTxDynamicPart` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `I-SIGNAL-I-PDU`. `IpduMTxDynamicPduRef` references the container created for the `I-SIGNAL-I-PDU` in the `EcuC` module configuration. `IpduMTxSelectorValue` is set to `SELECTOR-FIELD-CODE`. `IpduMSegment`: see `IpduMRxDynamicPart/IpduMSegment`.
`IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart`	If a `STATIC-PART` of the `MULTIPLEXED-I-PDU` exists, which references an `I-SIGNAL-I-PDU` that is routed (see Section 3.3.6, "PDU routing") or contains at least one `I-SIGNAL` sent by the imported `ECU-INSTANCE`, an `IpduMTxStaticPart` container is created. The name of `IpduMTxStaticPart` is set to `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `I-SIGNAL-I-PDU`. `IpduMTxStaticPduRef` references the container created for the `I-SIGNAL-I-PDU` in the `EcuC` module configuration. For the configuration of `IpduMSegment` see `IpduMRxDynamicPart/IpduMSegment`.
`IpduMConfig/IpduMContainerRxPdu`	For every `CONTAINER-I-PDU` which is received by the imported `ECU-INSTANCE`, an `IpduMContainerRxPdu` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `CONTAINER-I-PDU`. `IpduMContainerHeaderSize` is configured depending on `HEADER-TYPE`: ► `IPDUM_HEADERTYPE_SHORT` for `SHORT-HEADER` ► `IPDUM_HEADERTYPE_LONG` for `LONG-HEADER`

Configuration parameters	Mapping description
	► `IPDUM_HEADERTYPE_NONE` for `NO-HEADER` `IpduMContainerRxAcceptContainedPdu` is configured depending on `RX-ACCEPT-CONTAINED-I-PDU`: ► `IPDUM_ACCEPT_ALL` for `ACCEPT-ALL` ► `IPDUM_ACCEPT_CONFIGURED` for `ACCEPT-CONFIGURED` `IpduMContainerRxPduRef` references the container created for the `CONTAINER-I-PDU` in the `EcuC` module configuration. `IpduMContainerQueueSize` is set to `CONTAINER-I-PDU/MINIMUM-RX-CONTAINER-QUEUE-SIZE` if it exists and if it is greater than the pre-existing `IpduMContainerQueueSize` value.
`IpduMConfig/IpduM-ContainerTxPdu`	For every `CONTAINER-I-PDU` which is sent by the imported `ECU-INSTANCE`, an `IpduMContainerTxPdu` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `CONTAINER-I-PDU`. `IpduMContainerHeaderSize` is configured in the same way as in `IpduMConfig/IpduMContainerRxPdu`. `IpduMContainerTxFirstContainedPduTrigger` is configured depending on `CONTAINER-TRIGGER`: ► `true` for `FIRST-CONTAINED-TRIGGER` ► `false` for `DEFAULT-TRIGGER` `IpduMContainerTxSendTimeout` is set to `CONTAINER-TIMEOUT`. `IpduMContainerTxSizeThreshold` is set to `THRESHOLD-SIZE`. `IpduMUnusedAreasDefault` is set to `UNUSED-BIT-PATTERN`. `IpduMContainerTxPduRef` references the container created for the `CONTAINER-I-PDU` in the `EcuC` module configuration. `IpduMContainerTxTriggerMode` is set to `IPDUM_DIRECT` if the `CONTAINER-I-PDU` is contained in a `SECURED-I-PDU`. Otherwise, `IpduMContainerTxTriggerMode` is configured depending on the type of network that sends the `CONTAINER-I-PDU`: ► `IPDUM_DIRECT` for Ethernet and CAN

Configuration parameters	Mapping description
	▶ `IPDUM_TRIGGERTRANSMIT` for FlexRay and LIN `IpduMContainerQueueSize` is set to `CONTAINER-I-PDU/MINIMUM-TX-CONTAINER-QUEUE-SIZE` or, as a fall-back, to `CONTAINER-I-PDU/SDGS/SDG [@GID='IpduMContainerTxPduAttributes']/SD[@-GID='IpduMContainerQueueSize']` if the value is greater than the pre-existing `IpduMContainerQueueSize` value.
`IpduMConfig/IpduM-ContainedRxPdu`	For every PDU that is configured to be received within a `CONTAINER-I-PDU`, an `IpduMContainedRxPdu` container is created. A PDU is configured to be received within a `CONTAINER-I-PDU` if it meets the following conditions: ▶ The type of the PDU is a subclass of `I-PDU`. ▶ The PDU contains a `CONTAINED-I-PDU-PROPS` subelement. ▶ The PDU is referenced by a `PDU-TRIGGERING`. ▶ The imported `ECU-INSTANCE` owns an `I-PDU-PORT` that is referenced by the `PDU-TRIGGERING` and which has its `COMMUNICATION-DIRECTION` set to `IN`. ▶ The `PDU-TRIGGERING` is referenced by at least one received `CONTAINER-I-PDU`. The name of the `IpduMContainedRxPdu` is set to `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the contained PDU. `IpduMContainedPduHeaderId` is configured with either `CONTAINED-I-PDU-PROPS/HEADER-ID-LONG-HEADER` or `CONTAINED-I-PDU-PROPS/HEADER-ID-SHORT-HEADER` or not configured at all, depending on the `HEADER-TYPE` value of the `CONTAINER-I-PDU`: ▶ `CONTAINED-I-PDU-PROPS/HEADER-ID-LONG-HEADER` is taken if `HEADER-TYPE` is set to `LONG-HEADER`. ▶ `CONTAINED-I-PDU-PROPS/HEADER-ID-SHORT-HEADER` is taken if `HEADER-TYPE` is set to `SHORT-HEADER`. `IpduMContainedPduHeaderId` is not configured, if `HEADER-TYPE` is set to `NO-HEADER`. If the PDU is not associated with a `CONTAINER-I-PDU` or the associated `CONTAINER-I-PDU` does not provide a `HEADER-TYPE`, `CONTAINED-I-PDU-PROPS/HEADER-ID-LONG-HEADER` is taken. If the contained PDU does not

Configuration parameters	Mapping description
	provide `CONTAINED-I-PDU-PROPS/HEADER-ID-LONG-HEADER`, `CONTAINED-I-PDU-PROPS/HEADER-ID-SHORT-HEADER` is taken instead. `IpduMContainedRxInContainerPduRef` references the container created for the associated `CONTAINER-I-PDU`. The reference is configured if `RX-ACCEPT-CONTAINED-I-PDU` of the `CONTAINER-I-PDU` is set to a value other than `ACCEPT-ALL` or not set at all. `IpduMContainedRxPduRef` references the container created for the PDU in the `EcuC` module configuration. `IpduMContainedPduOffset` is set to `CONTAINED-I-PDU-PROPS/OFFSET` if the `HEADER-TYPE` is set to `NO-HEADER`. `IpduMPduUpdateBitPosition` is set to `CONTAINED-I-PDU-PROPS/UPDATE-INDICATION-BIT-POSITION` if the `HEADER-TYPE` is set to `NO-HEADER`.
`IpduMConfig/IpduMContainedTxPdu`	For every PDU that is sent within a `CONTAINER-I-PDU`, an `IpduMContainedTxPdu` container is created. A PDU is sent within a `CONTAINER-I-PDU` if it meets the conditions as follows: ▶ The type of the PDU is a subclass of `I-PDU`. ▶ The PDU contains a `CONTAINED-I-PDU-PROPS` subelement. ▶ The PDU is referenced by a `PDU-TRIGGERING`. ▶ The imported `ECU-INSTANCE` owns an `I-PDU-PORT` that is referenced by the `PDU-TRIGGERING` and which has its `COMMUNICATION-DIRECTION` set to `OUT`. ▶ The `PDU-TRIGGERING` is referenced from the `CONTAINER-I-PDU`. The name of the `IpduMContainedTxPdu` is set to `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the contained PDU. `IpduMContainedPduHeaderId` is configured in the same way as the corresponding parameter in `IpduMConfig/IpduMContainedRxPdu`. `IpduMContainedTxPduCollectionSemantics` is configured depending on `CONTAINED-I-PDU-PROPS/COLLECTION-SEMANTICS`: ▶ `IPDUM_COLLECT_LAST_IS_BEST` for `LAST-IS-BEST` ▶ `IPDUM_COLLECT_QUEUED` for `QUEUED`

Configuration parameters	Mapping description
	`IpduMContainedTxPduSendTimeout` is set to `CONTAINED-I-PDU-PROPS/TIMEOUT`. `IpduMContainedTxPduTrigger` is configured depending on `CONTAINED-I-PDU-PROPS/TRIGGER`: ► `IPDUM_TRIGGER_ALWAYS` for `ALWAYS` ► `IPDUM_TRIGGER_NEVER` for `NEVER` `IpduMContainedTxInContainerPduRef` references the container created for the associated `CONTAINER-I-PDU`. `IpduMContainedTxPduRef` is configured in the same way as `IpduMContainedRxPduRef` in `IpduMConfig/IpduMContainedRxPdu`. `IpduMContainedPduOffset` is configured in the same way as the corresponding parameter in `IpduMConfig/IpduMContainedRxPdu`. `IpduMPduUpdateBitPosition` is configured in the same way as the corresponding parameter in `IpduMConfig/IpduMContainedRxPdu`. `IpduMContainedTxPduPriority` is set to `CONTAINED-I-PDU-PROPS/PRIORITY`.

3.4.30. J1939Dcm

Configuration parameters	Mapping description
`J1939DcmConfigSet/` `J1939DcmChannel`	For each `J-1939-CLUSTER` the imported `ECU-INSTANCE` is attached to, one `J1939DcmChannel` container is created. Its name is `J1939DcmChannel_<name>` where `<name>` the short name of the `J1939Cluster`. `J1939DcmComMChannelRef` is set up to refer to the `ComMChannel` container that represents the `J-1939-CLUSTER` in the `ComM` BSW module configuration.
`J1939DcmConfigSet/` `J1939DcmNode`	For each `J-1939-CLUSTER` the imported `ECU-INSTANCE` is attached to, one `J1939DcmNode` container is created. Its name is `J1939DcmNode_-<name>` where `<name>` the short name of the `J1939Cluster`. If there is a `J-1939-NM-NODE` which refers to one of the `COMMUNICATION-CON-TROLLER` elements of the imported `ECU-INSTANCE` and which is contained in a `J-1939-NM-CLUSTER` that refers to the `J-1939-CLUSTER`, then

Configuration parameters	Mapping description
	`J1939DcmNmNodeRef` is configured to refer to the container representing the `J-1939-NM-NODE` in the `J1939Nm` configuration. `J1939DcmNodeRmUserRef` is configured to refer to the container in the `J1939Rm` BSW module that represents the `J1939RM_USER_J1939DCM` user of the `J-1939-CLUSTER`, as described in Section 3.4.32, "J1939Rm".

`J1939RmConfigSet/`
`J1939DcmNode/`
`J1939DcmDiagnostic-`
`MessageSupport`
	For each PDU of the `J1939Rm` user `J1939Dcm_USER_J1939DCM`, see Section 3.4.32, "J1939Rm", one `J1939DcmDiagnosticMessageSupport` container is created. Its name is the short name of the PDU.
	
	`J1939DcmDiagnosticMessageSupportChannelRef` is set up to refer to the `J1939DcmChannel` container that represents the `J-1939-CLUSTER` in which the PDU is sent or received.
	
	If the PDU has a `PGN` associated which the `J1939Dcm` configuration is supporting, i.e. `[J1939DCM_DM01_SUPPORT..J1939DCM_DM57_SUPPORT]`, then `J1939DcmDmxSupport` is configured to that value.
	
	If the imported `ECU-INSTANCE` sends the PDU, one `J1939DcmTxPdu` sub container is created. The container name is the short name of the PDU, `J1939DcmTxPduRef` is configured to refer to the `EcuC` container that has been created for the PDU.

3.4.31. J1939Nm

Configuration parameters	Mapping description
`J1939NmConfigSet/` `J1939NmChannel`	For each `J-1939-NM-CLUSTER` that contains a `J-1939-NM-NODE` which in turn refers via `J-1939-NM-NODE/CONTROLLER-REF` to a `CAN-COMMUNICATION-CONTROLLER` of the imported `ECU-INSTANCE`, one `J1939NmChannel` container is created. Its name is the short name of the `J-1939-NM-CLUSTER`. The following parameters are configured: If `J-1939-NM-CLUSTER/COMMUNICATION-CLUSTER-REF` refers to a valid `J-1939-CLUSTER`, `J-1939-CLUSTER/USES-ADDRESS-ARBITRATION` is used to configure `J1939NmChannelUsesAddressArbitration`. If `J-1939-NM-CLUSTER/COMMUNICATION-CLUSTER-REF` refers to a valid `J-1939-CLUSTER`, `J1939NmComMNetworkHandleRef` is set up to reference the `ComMChannel` container that represents this `J-1939-CLUSTER` in the `ComM` BSW module configuration.

Configuration parameters	Mapping description
	If the `J-1939-NM-NODE` that refers to a `CAN-COMMUNICATION-CON-TROLLER` of the imported `ECU-INSTANCE` references `NM-PDU` elements via `J-1939-NM-NODE/TX-NM-PDU-REFS` and via `J-1939-NM-NODE/RX-NM-PDU-REFS`, and the imported `ECU-INSTANCE` actually sends/receives these PDUs, `J1939NmTxPdu/J1939NmTxPduRef` and `J1939NmRxPdu/J1939NmRxPduRef` are configured to reference the containers that represent these PDUs in the `EcuC` BSW module configuration.
`J1939NmConfigSet/` `J1939NmNode`	For each `J-1939-NM-NODE` entity that refers via `J-1939-NM-NODE/CON-TROLLER-REF` to a `CAN-COMMUNICATION-CONTROLLER` of the imported `ECU-INSTANCE`, one `J1939NmNode` container is created. Its name is the short name of the `J-1939-NM-NODE` entity. The following parameters are derived from the `J-1939-NM-NODE` and its `NODE-NAME` child entity: `J1939NmNodeNameArbitraryAddressCapable` is set to `NODE-NAME/ARBITRARY-ADDRESS-CAPABLE`. `J1939NmNodeNameECUInstance` is set to `NODE-NAME/ECU-INSTANCE`. `J1939NmNodeNameFunction` is set to `NODE-NAME/FUNCTION`. `J1939NmNodeNameFunctionInstance` is set to `NODE-NAME/FUNC-TION-INSTANCE`. `J1939NmNodeNameIdentityNumber` is set to `NODE-NAME/IDEN-TITIY-NUMBER`. `J1939NmNodeNameIndustryGroup` is set to `NODE-NAME/IN-DUSTRY-GROUP`. `J1939NmNodeNameManufacturerCode` is set to `NODE-NAME/MANUFAC-TURER-CODE`. `J1939NmNodeNameVehicleSystem` is set to `NODE-NAME/VEHI-CLE-SYSTEM`. `J1939NmNodeNameVehicleSystemInstance` is set to `NODE-NAME/VE-HICLE-SYSTEM-INSTANCE`. `J1939NmNodePreferredAddress` is set to `J-1939-NM-NODE/NM-NODE-ID`. In `J1939NmNodeChannelRef`, one reference is configured which refers to the `J1939NmChannel` container that was created for the `J-1939-NM-CLUSTER` entity that contains the `J-1939-NM-NODE`.

3.4.32. J1939Rm

Configuration parameters	Mapping description
`J1939RmConfigSet/` `J1939RmChannel`	For each `J-1939-CLUSTER` the imported Ecu Instance is attached to, one `J1939RmChannel` container is created. Its name is `J1939RmChannel_-` `<name>` where `<name>` the short name of the `J-1939-CLUSTER`. For the configuration of its `J1939RmRqstRxPdu, J1939RmRqstTxPdu,` `J1939RmAckmRxPdu`, and `J1939RmAckmTxPdu` sub containers, every PDU that fulfills the following conditions is retrieved:

For the `J1939RmConfigSet/J1939RmChannel` row, the right-hand cell also contains:

► The PDU is sent or received by the imported `ECU-INSTANCE`

► The PDU is directly contained in a `CAN-FRAME`

► The `CAN-FRAME-TRIGGERING` of the `CAN-FRAME` has a valid `CAN-FRAME-TRIGGERING/IDENTIFIER` field assigned, `CAN-FRAME-TRIGGERING/CAN-ADDRESSING-MODE` is set to `EXTENDED`, and `CAN-FRAME-TRIGGERING/J-1939-REQUESTABLE` is either set to `false` or not defined.

► The nine most significant bits of the *PGN* (Parameter Group Number) value, i.e. the bits `[16..24]` of `CAN-FRAME-TRIGGERING/IDEN-TIFIER` contain the value `0xEA`, or `0xE8`.

► The PDU is not of type `N-PDU`

Depending on the send/receive direction of the collected PDUs, and the most significant bits of the *PGN* value, the following sub containers are configured along with a reference parameter that is set up to refer to the `EcuC` container representing the collected PDUs.

Direc-tion/PGN	Container	Reference Parameter
`Tx/0xEA`	`J1939RmRqstTxPdu`	`J1939RmRqstTxPduRef`
`Rx/0xEA`	`J1939RmRqstRxPdu`	`J1939RmRqstRxPduRef`
`Tx/0xE8`	`J1939RmAckmTxPdu`	`J1939RmAckmTxPduRef`
`Rx/0xE8`	`J1939RmAckmRxPdu`	`J1939RmAckmRxPduRef`

| `J1939RmConfigSet/` `J1939RmNode` | For each `J-1939-CLUSTER` the imported `ECU-INSTANCE` is attached to, one `J1939RmNode` container is created. Its name is `J1939RmNode_-` `<name>` where `<name>` the short name of the `J-1939-CLUSTER`.

One `J1939RmNodeChannelRef` container is configured to reference the corresponding `J1939RmChannel` container. If there is a `J-1939-NM-NODE` which refers to one of the `COMMUNICATION-CONTROLLER` elements of |

Configuration parameters	Mapping description
	the imported `ECU-INSTANCE` and which is contained in a `J-1939-NM-CLUSTER` that refers to the `J-1939-CLUSTER`, then `J1939RmNmNodeRef` is configured to refer to the container representing the `J-1939-NM-NODE` in the `J1939Nm` configuration. For the configuration of the `J1939RmUser` sub containers, all PDUs of the `J-1939-CLUSTER` that can be requested by the `J1939Rm` BSW module are collected. They must satisfy the following conditions: ▶ The PDU is sent or received by the imported `ECU-INSTANCE`. ▶ The PDU is directly contained in a `CAN-FRAME`. ▶ The `CAN-FRAME-TRIGGERING` of the `CAN-FRAME` has a valid `CAN-FRAME-TRIGGERING/IDENTIFIER` field assigned, `CAN-FRAME-TRIGGERING/CAN-ADDRESSING-MODE` is set to `EXTENDED`, and `CAN-FRAME-TRIGGERING/J-1939-REQUESTABLE` is set to `true`. ▶ The PDU is not of type `N-PDU`. Moreover, all PDUs of the `J-1939-CLUSTER` that are processed by the `J1939Tp` BSW module and which can be requested by the `J1939Rm` BSW module as well are collected. They fulfill the following conditions: ▶ The PDU is sent or received by the imported `ECU-INSTANCE`. ▶ There is a `J-1939-TP-CONNECTION` which contains a `J-1939-TP-PG` element which in turn references the PDU via `J-1939-TP-PG/SDU-REFS/SDU-REF`. ▶ The `J-1939-TP-CONNECTION` is either sent or received by the imported Ecu Instance and it belongs to a `J-1939-TP-CONFIG` that references the `J-1939-CLUSTER`. ▶ The `J-1939-TP-PG` has its `J-1939-TP-PG/REQUESTABLE` field set to `true`. The two collections of PDUs are split into subgroups each of which belongs to a given `J1939RmUser`. ▶ The PDUs for `J1939RM_USER_COM` are of type `I-SIGNAL-I-PDU`. ▶ The PDUs for `J1939RM_USER_J1939NM` are of type `NM-PDU`, they are sent directly in a `CAN-FRAME`, and they must have assigned the *PGN* `0xee00`. ▶ The PDUs for `J1939RM_USER_J1939DCM` are of type `J-1939-DCM-I-PDU`.

Configuration parameters	Mapping description

`J1939RmConfigSet/`
`J1939RmNode/`
`J1939RmUser/`
`J1939RmUser_Com`
	If PDUs for `J1939RM_USER_COM` exist, one `J1939RmUser_Com` container is created, `J1939RmUserType` is set to `J1939RM_USER_COM`. For each of the PDUs, one `J1939RmComIPdu` sub container is created, its name is the short name of the PDU. `ComIPduType` is set to `ComIPdu_TP` for PDUs that either have a dynamic length or a length that exceeds eight bytes. For all other PDUs, `ComIPduType` is set to `ComIPdu_IF`. An `I-SIGNAL-I-PDU` has a dynamic length if at least one of its `I-SIGNAL` elements refers to a `SYSTEM-SIGNAL` that has its `DYNAMIC-LENGTH` field set to `true`.
	
	For PDUs processed by `J1939Tp`, the parameters `J1939RmComIPduDA`, `J1939RmComIPduSA` are configured according to the `J1939TpTxDa/` `J1939TpRxDa` and `J1939TpTxSa/J1939TpRxSa` parameters of the `J1939TpTxChannel` or `J1939TpRxChannel` the PDU is processed in, see [Section 3.4.33, "J1939Tp"](#). `J1939RmComIPduPGN` is configured in the same way as the `J1939TpTxPgPGN/J1939TpRxPgPGN` parameter values of the `J1939TpTxPg/J1939TpRxPg` containers that were set up for the PDUs in the `J1939Tp` module configuration.
	
	For PDUs that are directly sent/received in a `CAN-FRAME`, the `J1939RmComIPduDA`, `J1939RmComIPduSA`, and `J1939RmComIPduPGN` are extracted from `CAN-FRAME-TRIGGERING/IDENTIFIER`:
	
	▶ If the bit range `[16..23]` yields a value that is less than 240, `J1939RmComIPduDA` is set to zero. Otherwise `J1939RmComIPduDA` is set to the value of the bit range `[8..15]`.
	
	▶ `J1939RmComIPduSA` is set to the value of the bit range `[0..7]`.
	
	▶ If the bit range `[16..23]` yields a value that is less than 240, `J1939RmComIPduPGN` is set to the value of the bit range `[16..24]` multiplied by 256. Otherwise `J1939RmComIPduPGN` is set to the value of the bit range `[8..24]`.
	
	One `J1939RmLowerComIPdu` and one `J1939RmUpperComIPdu` container are created, their name is the short name of the PDU, prefixed by `LO` for the `J1939RmLowerComIPdu` and prefixed by `UP` for the `J1939RmUpperComIPdu` container. The reference parameters `J1939RmLowerComIPduId/J1939RmLowerComIPduRef` and `J1939RmUpperComIPdu/J1939RmUpperComIPduRef` are configured to refer to the representation of the PDU in the `EcuC` module.

`J1939RmConfigSet/` `J1939RmNode/`	If PDUs for `J1939RM_USER_J1939NM` exist, one `J1939RmUser_J1939Nm` container is created, `J1939RmUserType` is set to `J1939RM_USER_-`

Configuration parameters	Mapping description
`J1939RmUser/` `J1939RmUser_J1939Nm`	`J1939NM`. One `J1939RmUserPGN` entry is created, its value is set to `0xee00`.

`J1939RmConfigSet/`
`J1939RmNode/`
`J1939RmUser/`
`J1939RmUser_J1939Dcm`
	If PDUs for `J1939RM_USER_J1939DCM` exist, one `J1939RmUser_-`
	`J1939Dcm` container is created, `J1939RmUserType` is set to `J1939RM_-`
	`USER_J1939DCM`. For every PDU, its PGN is determined in the same way
	as for the `J1939RmComIPduPGN` value of `J1939RM_USER_COM` PDUs. For
	each valid *PGN* value, one `J1939RmUserPGN` entry is created, its value is
	set to the *PGN* value.

3.4.33. J1939Tp

Configuration parameters	Mapping description

`J1939TpConfiguration/`
`J1939TpTxChannel,`
`J1939TpConfiguration/`
`J1939TpRxChannel`
	For each `J-1939-TP-CONNECTION` which the imported Ecu Instance
	either receives or sends, and which belongs to a `J-1939-TP-CONFIG`
	that references a `J-1939-CLUSTER`, one `J1939TpRxChannel` or one
	`J1939TpTxChannel` is created.
	
	The imported `ECU-INSTANCE` receives a `J-1939-TP-CONNECTION` if
	it references a `J-1939-TP-NODE` via `J-1939-TP-CONNECTION/RE-`
	`CEIVER-REF` and if the `J-1939-TP-NODE` refers via `J-1939-TP-NODE/`
	`CONNECTOR-REF` to a `CAN-COMMUNICATION-CONNECTOR` of the imported
	`ECU-INSTANCE`.
	
	The imported `ECU-INSTANCE` sends a `J-1939-TP-CONNECTION` if it ref-
	erences a `J-1939-TP-NODE` via `J-1939-TP-CONNECTION/TRANSMIT-`
	`TER-REF` and if the `J-1939-TP-NODE` refers via `J-1939-TP-NODE/CON-`
	`NECTOR-REF` to a `CAN-COMMUNICATION-CONNECTOR` of the imported
	`ECU-INSTANCE`.
	
	The container name is the short name of the `J-1939-TP-CONNECTION`.
	The following parameters are set for `J1939TpTxChannel` containers:
	
	`J1939TpTxCancellationSupport` is set to `J-1939-TP-CONNEC-`
	`TION/CANCELLATION`.
	
	If `J-1939-TP-CONNECTION/RECEIVER-REF` refers to a valid `J-1939-`
	`TP-NODE`, and `J-1939-TP-NODE/TP-ADDRESS-REF` refers to a valid `TP-`
	`ADDRESS`, `TP-ADDRESS/TP-ADDRESS` is used to configure `J1939TpTxDa`.
	
	`J1939TpTxDynamicBlockCalculation` is set to `J-1939-TP-CONNEC-`
	`TION/DYNAMIC-BS`.

Configuration parameters	Mapping description
	`J1939TpTxMaxPacketsPerBlock` is set to `J-1939-TP-CONNECTION/MAX-EXP-BS`. `J1939TpTxProtocolType` is set to `J1939TP_PROTOCOL_BAM` if `J-1939-TP-CONNECTION/BROADCAS` is `true`, it is set to `J1939TP_PROTOCOL_CMDT` if `J-1939-TP-CONNECTION/BROADCAST` is `false` `J1939TpTxRetrySupport` is set to `J-1939-TP-CONNECTION/RETRY`. If `J-1939-TP-CONNECTION/TRANSMITTER-REF` refers to a valid `J-1939-TP-NODE`, and `J-1939-TP-NODE/TP-ADDRESS-REF` refers to a valid `TP-ADDRESS`, `TP-ADDRESS/TP-ADDRESS` is used to configure `J1939TpTxSa`. The following parameters are set for `J1939TpRxChannel` containers: `J1939TpRxCancellationSupport` is configured in the same way as `J1939TpTxCancellationSupport`. `J1939TpRxDa` is configured in the same way as `J1939TpTxDa`. `J1939TpRxDynamicBlockCalculation` is configured in the same way as `J1939TpTxDynamicBlockCalculation`. `J1939TpRxDynamicBufferRatio` is set to `J-1939-TP-CONNECTION/BUFFER-RATIO`. `J1939TpRxPacketsPerBlock` is set to `J-1939-TP-CONNECTION/MAX-BS`. `J1939TpRxProtocolType` is configured in the same way as `J1939TpTxProtocolType`. `J1939TpRxRetrySupport` is configured in the same way as `J1939TpTxRetrySupport`. `J1939TpRxSa` is configured in the same way as `J1939TpTxSa`.
`N-PDU containers of J1939TpTxChannels`	If the imported `ECU-INSTANCE` sends one of the PDUs that are referenced by `J-1939-TP-CONNECTION/FLOW-CONTROL-PDU-REFS`, one `J1939TpTxCmNPdu` container is created, its name is the short name of the PDU. `J1939TpTxCmNPduRef` is configured to reference the representation of the PDU in the `EcuC` module. If the imported `ECU-INSTANCE` receives one of the PDUs that are referenced by `J-1939-TP-CONNECTION/FLOW-CONTROL-PDU-REFS`, one

Configuration parameters	Mapping description
	`J1939TpRxFcNPdu` container is created, its name is the short name of the PDU. `J1939TpRxFcNPduRef` is configured to reference the representation of the PDU in the `EcuC` module. If the imported `ECU-INSTANCE` sends the PDU that is referenced by `J-1939-TP-CONNECTION/DATA-PDU-REF`, one `J1939TpTxDtNPdu` container is created, its name is the short name of the PDU. `J1939TpTxDtNPduRef` is configured to reference the representation of the PDU in the `EcuC` module.
`N-PDU containers of J1939TpRxChannels`	If the imported `ECU-INSTANCE` receives one of the PDUs that are referenced by `J-1939-TP-CONNECTION/FLOW-CONTROL-PDU-REFS`, one `J1939TpRxCmNPdu` container is created, its name is the short name of the PDU. `J1939TpRxCmNPduRef` is configured to reference the representation of the PDU in the `EcuC` module. If the imported `ECU-INSTANCE` sends one of the PDUs that are referenced by `J-1939-TP-CONNECTION/FLOW-CONTROL-PDU-REFS`, one `J1939TpTxFcNPdu` container is created, its name is the short name of the PDU. `J1939TpTxFcNPduRef` is configured to reference the representation of the PDU in the `EcuC` module. If the imported `ECU-INSTANCE` receives the PDU that is referenced by `J-1939-TP-CONNECTION/DATA-PDU-REF`, one `J1939TpRxDtNPdu` container is created, its name is the short name of the PDU. `J1939TpRxDtNPduRef` is configured to reference the representation of the PDU in the `EcuC` module.
`J1939TpTxChannel/ J1939TpTxPg`	For every `J-1939-TP-PG` of the `J-1939-TP-CONNECTION` which references a PDU which the imported Ecu instance is sending via `J-1939-TP-PG/SDU-REFS` or `J-1939-TP-PG/DIRECT-PDU-REF`, one `J1939TpTxPg` container is created. Its name is either `Pg_<pgn>` if `J-1939-TP-PG/PGN` is available, or `Pg_Idx_<idx>` if `J-1939-TP-PG/PGN` is not available, where `<idx>` is a zero based index. The following parameters are configured: If there is at least one PDU that is referenced via `J-1939-TP-PG/SDU-REFS` and that the imported Ecu Instance sends is an `I-SIGNAL-I-PDU` containing at least one `I-SIGNAL`, `J1939TpTxPgDynLength` is configured. The parameter is set to `true` if at least one of the `I-SIGNAL` elements refers to a `SYSTEM-SIGNAL` that has its `DYNAMIC-LENGTH` field set to `true`. Otherwise the parameter is set to `false`. `J1939TpTxPgPGN` is set to `J-1939-TP-PG/PGN`.

Configuration parameters	Mapping description
	If the imported `ECU-INSTANCE` sends the PDU that is referenced by `J-1939-TP-PG/DIRECT-PDU-REF`, one `J1939TpTxDirectNPdu` container is created, its name is the short name of the PDU. `J1939TpTxDirectNPduRef` is configured to reference the representation of the PDU in the `EcuC` module. For each PDU that is referenced via `J-1939-TP-PG/SDU-REFS` and which the imported `ECU-INSTANCE` is sending, one `J1939TpTxNSdu` container is created, its name is the short name of the PDU. `J1939TpTxNSduRef` is configured to reference the representation of the PDU in the `EcuC` module.
`J1939TpRxChannel/` `J1939TpRxPg`	For every `J-1939-TP-PG` of the `J-1939-TP-CONNECTION` which references a PDU which the imported Ecu instance is receiving via `J-1939-TP-PG/SDU-REFS` or `J-1939-TP-PG/DIRECT-PDU-REF`, one `J1939TpRxPg` container is created. Its name is created in the same way as it is for `J1939TpTxPg` containers. The following parameters are configured: `J1939TpRxPgDynLength` is configured in the same way as `J1939TpTxPgDynLength`. `J1939TpRxPgPGN` is configured in the same way as `J1939TpTxPgPGN`. If the imported `ECU-INSTANCE` receives the PDU that is referenced by `J-1939-TP-PG/DIRECT-PDU-REF`, one `J1939TpRxDirectNPdu` container is created, its name is the short name of the PDU. `J1939TpRxDirectNPduRef` is configured to reference the representation of the PDU in the `EcuC` module. For each PDU that is referenced via `J-1939-TP-PG/SDU-REFS` and which the imported `ECU-INSTANCE` is receiving, one `J1939TpRxNSdu` container is created, its name is the short name of the PDU. `J1939TpRxNSduRef` is configured to reference the representation of the PDU in the `EcuC` module.

3.4.34. LdCom

Configuration parameters	Mapping description
`LdComIPdu`	For every PDU instance (see Section 3.3.1.3, "Instance handling") that represents an `I-SIGNAL-I-PDU`, which is sent or received by the imported `ECU-INSTANCE` and for which the following conditions apply, an `LdComIPdu` container is created.

Configuration parameters	Mapping description
	▶ The PDU instance contains exactly one `I-SIGNAL-TO-I-PDU-MAPPING` referencing an `I-SIGNAL`. ▶ `ComSignalType` of the `I-SIGNAL` is `UINT8_N` or `UINT8_DYN`. ▶ `PACKING-BYTE-ORDER` of the `I-SIGNAL-TO-I-PDU-MAPPING` is set to `OPAQUE`. ▶ `START-POSITION` of the `I-SIGNAL-TO-I-PDU-MAPPING` is set to `0`. ▶ `UPDATE-INDICATION-BIT-POSITION` of the `I-SIGNAL-TO-I-PDU-MAPPING` is not set. ▶ The `I-SIGNAL-PORT` referenced by the `I-SIGNAL`'s `I-SIGNAL-TRIGGERING` has no `TIMEOUT` defined. ▶ The `I-SIGNAL-PORT` referenced by the `I-SIGNAL`'s `I-SIGNAL-TRIGGERING` has no `FIRST-TIMEOUT` defined. ▶ The `I-SIGNAL-PORT` referenced by the `I-SIGNAL`'s `I-SIGNAL-TRIGGERING` either has no `DATA-FILTER` defined or has `DATA-FILTER/DATA-FILTER-TYPE` set to `ALWAYS`. ▶ There is no `I-SIGNAL-I-PDU-GROUP` which references the `I-SIGNAL-I-PDU`. ▶ There is no `I-SIGNAL-MAPPING` which references the `I-SIGNAL`'s `I-SIGNAL-TRIGGERING`. The following conditions only apply for `I-SIGNAL-I-PDU`s, whose `LdComIPduDirection` is set to `LDCOM_SEND`. ▶ `I-PDU-TIMING-SPECIFICATIONS/I-PDU-TIMING/TRANSMISSION-MODE-DECLARATION` is defined. ▶ `TRANSFER-PROPERTY` of the `I-SIGNAL-TO-I-PDU-MAPPING` is set to `TRIGGERED` or `TRIGGERED-WITHOUT-REPETITION`. ▶ `MINIMUM-DELAY` of the `I-PDU-TIMING` is not set if the `LdComIPduDirection` is set to `LDCOM_SEND`. ▶ The `I-PDU-TIMING` has a `TRANSMISSION-MODE-TRUE-TIMING` but no `TRANSMISSION-MODE-FALSE-TIMING` defined. ▶ The `TRANSMISSION-MODE-DECLARATION` either has no `TRANSMISSION-MODE-CONDITIONS` defined or one single `TRANSMISSION-MODE-CONDITION` which references the `I-SIGNAL-TO-I-PDU-MAPPING` and which has its `DATA-FILTER/DATA-FILTER-TYPE` set to `ALWAYS`.

Configuration parameters	Mapping description
	▶ The `TRANSMISSION-MODE-DECLARATION` has no `MODE-DRIVEN-FALSE-CONDITIONS` and no `MODE-DRIVEN-TRUE-CONDITIONS` defined. ▶ The `TRANSMISSION-MODE-TRUE-TIMING` has only an `EVENT-CONTROLLED-TIMING` defined and its `NUMBER-OF-REPETITIONS` must be set to `0`. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `I-SIGNAL-I-PDU`. `LdComPduRef` references the corresponding container in the `EcuC` module configuration. `LdComSystemTemplateSignalRef` is set to the AUTOSAR path of the `I-SIGNAL-TO-I-PDU-MAPPING`. `LdComApiType` is configured in the same way as `ComIPduType` in [Section 3.4.8, "Com"](#), with the exception that `LDCOM_TP` and `LDCOM_IF` are used as configured parameter values instead of `TP` and `NORMAL`. `LdComIPduDirection` is set to `LDCOM_SEND` if the PDU instance is sent by the imported `ECU-INSTANCE`. Otherwise it is set to `LDCOM_RECEIVE`.

3.4.35. Lin

Configuration parameters	Mapping description
`LinChannel`	For every `LIN-CLUSTER` which belongs to the imported `ECU-INSTANCE`, a `LinChannel` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `LIN-CLUSTER`'s `PHYSICAL-CHANNEL`. A `LIN-CLUSTER` belongs to the imported `ECU-INSTANCE` if a `LIN-MASTER` connects the `LIN-CLUSTER` to the imported `ECU-INSTANCE` via its `COMMUNICATION-CONNECTOR`. If a `LIN-CLUSTER` is not connected to the imported `ECU-INSTANCE` by a `LIN-MASTER`, a warning is reported and no `LinChannel` container is created for this `LIN-CLUSTER`. If the `LIN-CLUSTER` comprises more than one `PHYSICAL-CHANNEL`, an error is reported.

Configuration parameters	Mapping description
	`LinChannelBaudRate` is set to `BAUDRATE`. If `BAUDRATE` is not available, `LinChannelBaudRate` is set to `SPEED` * 1000.

3.4.36. LinIf

Configuration parameters	Mapping description
`LinIfChannel`	For every `LIN-CLUSTER` which belongs to the imported `ECU-INSTANCE`, a `LinIfChannel` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `LIN-CLUSTER`'s `PHYSICAL-CHANNEL`. A `LIN-CLUSTER` belongs to the imported `ECU-INSTANCE` if a `LIN-MASTER` connects the `LIN-CLUSTER` to the imported `ECU-INSTANCE` via its `COMMUNICATION-CONNECTOR`. If a `LIN-CLUSTER` is not connected to the imported `ECU-INSTANCE` by a `LIN-MASTER`, a warning is reported and no `LinIfChannel` container is created for this `LIN-CLUSTER`. If the `LIN-CLUSTER` comprises more than one `PHYSICAL-CHANNEL`, an error is reported. If the single `PHYSICAL-CHANNEL` is not a `LIN-PHYSICAL-CHANNEL`, an error is reported. `LinIfChannelRef` references the corresponding container in the `Lin` module configuration. `LinIfComMNetworkHandleRef` references the corresponding container in the `ComM` module configuration.
`LinIfChannel/LinIfMaster`	The following parameters are set using the `LIN-MASTER` connecting the imported `ECU-INSTANCE` to the `LIN-CLUSTER`: `LinIfClusterTimeBase` is set to `TIME-BASE`. `LinIfJitter` is set to `TIME-BASE-JITTER`.
`LinIfChannel/LinIfSlave`	If the `LIN-MASTER` of a given `LIN-CLUSTER` contains one or more `LIN-SLAVE-CONFIG` elements, these are used to configure the `LinIfSlave` configuration containers. If the `LIN-MASTER` does not contain any `LIN-SLAVE-CONFIG`, all `LIN-SLAVE` elements that are connected to the `LIN-CLUSTER` are used to configure the `LinIfSlave` configuration containers.

Configuration parameters	Mapping description
	For every `LIN-SLAVE-CONFIG` or `LIN-SLAVE` element one `LinIfSlave` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of either `LIN-SLAVE-CONFIG/IDENT` or `LIN-SLAVE`. `LinIfProtocolVersion` is set to `PROTOCOL-VERSION`. `LinIfConfiguredNad` is set to `CONFIGURED-NAD`. `LinIfFunctionId` is set to `FUNCTION-ID`. `LinIfSupplierId` is set to `SUPPLIER-ID`. `LinIfVariant` is set to `VARIANT-ID`.
`LinIfChannel/LinIf-Frame`	In the following, `APPLICATION-ENTRY` elements referencing `FRAME-TRIGGERING` elements are referred to as `ApplicationFrame` elements; all other types of `ENTRY` elements do not reference `FRAME-TRIGGERING` elements and are referred to as `ConfigurationFrame` elements. `SlaveToSlaveFrame` elements are a special kind of `ApplicationFrame` elements which are neither sent nor received by the `LIN-MASTER`. For every kind of frame, a `LinIfFrame` container is created. The container name for `ApplicationFrame` elements is `<PREFIX><name><INSTSUFFIX>`, for `SlaveToSlaveFrame` elements it is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `FRAME` referenced by the `LIN-FRAME-TRIGGERING`. The container name for `ConfigurationFrame` elements is `<PREFIX><name><auto incremented number>`, where `<name>` is the `SHORT-NAME` of the parent `LIN-SCHEDULE-TABLE`. `LinIfPid` is set to the protected identifier of the `LIN-FRAME-TRIGGERING`. For details on how to calculate the protected identifier refer to [2]. If the `LinIf-Frame` has been created for a `ConfigurationFrame`, `LinIfPid` is set to `60`, which is the ID of the MasterRequestFrame (MRF). If the `LIN-FRAME-TRIGGERING` references a `LIN-SPORADIC-FRAME`, `LinIfPid` is not configured. `LinIfLength` is set to `FRAME-LENGTH` of the `LIN-UNCONDITIONAL-FRAME` referenced by the `LIN-FRAME-TRIGGERING`. If the `LIN-FRAME-TRIGGERING` references a `LIN-SPORADIC-FRAME` or `LIN-EVENT-TRIGGERED-FRAME`, the `FRAME-LENGTH` of the first `LIN-UNCONDITIONAL-FRAME` referenced by the `LIN-SPORADIC-FRAME` or `LIN-EVENT-TRIGGERED-FRAME` is used. For `ConfigurationFrame` elements `LinIfLength` is always set to `8`. `LinIfFrameType` is set depending on the `IDENTIFIER` of the `LIN-FRAME-TRIGGERING`:

Configuration parameters	Mapping description
	▶ `MRF` for `60` ▶ `SRF` for `61` For other values of `IDENTIFIER LinIfFrameType` is set depending on the type of the referenced `FRAME`: ▶ `UNCONDITIONAL` for `LIN-UNCONDITIONAL-FRAME` elements ▶ `SPORADIC` for `LIN-SPORADIC-FRAME` elements ▶ `EVENT_TRIGGERED` for `LIN-EVENT-TRIGGERED-FRAME` elements For `ConfigurationFrame` elements `LinIfFrameType` is set depending on the type of the `TABLE-ENTRY`: ▶ `FREE` for `FREE-FORMAT-ENTRY` and `DATA-DUMP-ENTRY` ▶ `SAVE_CONFIGURATION` for `SAVE-CONFIGURATION-ENTRY` ▶ `ASSIGN_NAD` for `ASSIGN-NAD` ▶ `ASSIGN` for `ASSIGN-FRAME-ID` ▶ `UNASSIGN` for `UNASSIGN-FRAME-ID` ▶ `ASSIGN_FRAME_ID_RANGE` for `ASSIGN-FRAME-ID-RANGE` ▶ `CONDITIONAL` for `CONDITIONAL-CHANGE-NAD` For `ConfigurationFrame` elements `LinIfChecksumType` is set to `CLASSIC`. For `LIN-FRAME-TRIGGERING` elements referencing a `LIN-UNCONDITIONAL-FRAME`, `LinIfChecksumType` is set to `LIN-CHECKSUM`. For `LIN-SPORADIC-FRAME` elements and `LIN-EVENT-TRIGGERED-FRAME` elements `LIN-CHECKSUM` is taken from the `LIN-FRAME-TRIGGERING` elements referencing the substituted `LIN-UNCONDITIONAL-FRAME` elements. If inconsistencies are detected among the `LIN-CHECKSUM` values of these `LIN-FRAME-TRIGGERING` elements, a warning is reported.
`LinIfChannel/LinIfFrame/LinIfFixedFrameSduByte`	For `ConfigurationFrame` elements the byte array used to configure `LinIfFixedFrameSduByte` is calculated depending on the type of the `TABLE-ENTRY`. In the following, array elements are listed starting from byte zero, and the parameters `CONFIGURED-NAD`, `SUPPLIER-ID`, and `FUNCTION-ID` are either taken from the `LIN-SLAVE-CONFIG` referenced via `ASSIGNED-LIN-SLAVE-CONFIG-REF` or, if this reference does not exist, from the `LIN-SLAVE` referenced via `ASSIGNED-CONTROLLER-REF`. ▶ `FREE-FORMAT`: the byte array is taken directly from `BYTE-VALUES`.

Configuration parameters	Mapping description
	▶ ASSIGN-NAD: INITIAL-NAD of LIN-COMMUNICATION-CONNECTOR or INITIAL-NAD of LIN-SLAVE-CONFIG, 0x06, 0xb0, LSB of SUPPLIER-ID, MSB of SUPPLIER-ID, LSB of MESSAGE-ID, MSB of MESSAGE-ID, NEW-NAD .
	▶ SAVE-CONFIGURATION-ENTRY: CONFIGURED-NAD, 0x01, 0xb6, 0xff, 0xff, 0xff, 0xff, 0xff .
	▶ ASSIGN-FRAME-ID/UNASSIGN-FRAME-ID : CONFIGURED-NAD, 0x06, 0xb1, LSB of SUPPLIER-ID, MSB of SUPPLIER-ID, LSB of MESSAGE-ID, MSB of MESSAGE-ID, protected identifier of LIN-FRAME-TRIGGERING.
	The LIN-FRAME-TRIGGERING is referenced by ASSIGN-FRAME-ID/UNASSIGN-FRAME-ID via ASSIGNED-FRAME-TRIGGERING-REF/UNASSIGNED-FRAME-TRIGGERING-REF.
	If the ASSIGN-FRAME-ID/UNASSIGN-FRAME-ID entity references a LIN-SLAVE-CONFIG entity, the MESSAGE-ID is directly taken from ASSIGN-FRAME-ID/MESSAGE-ID, respectively from UNASSIGN-FRAME-ID/MESSAGE-ID.
	If the ASSIGN-FRAME-ID/UNASSIGN-FRAME-ID entity references a LIN-SLAVE entity, the MESSAGE-ID is taken from the LIN-CONFIGURABLE-FRAME referencing the same LIN-FRAME as the LIN-FRAME-TRIGGERING.
	▶ ASSIGN-FRAME-ID-RANGE: CONFIGURED-NAD, 0x06, 0xb7, START-INDEX, FRAME-PID[INDEX = 0]/PID, FRAME-PID[INDEX = 1]/PID, FRAME-PID[INDEX = 2]/PID, FRAME-PID[INDEX = 3]/PID.
	If ASSIGN-FRAME-ID-RANGE does not contain any FRAME-PID elements and the ASSIGN-FRAME-ID-RANGE refers to a LIN-SLAVE entity, the PID values are either retrieved from the LIN-ORDERED-CONFIGURABLE-FRAMES or, if these are not available, from the LIN-CONFIGURABLE-FRAMES of the LIN-SLAVE that is referenced via ASSIGNED-CONTROLLER-REF. The LIN-ORDERED-CONFIGURABLE-FRAMES and LIN-CONFIGURABLE-FRAMES of a slave are contained in the LIN-COMMUNICATION-CONNECTOR belonging to the LIN-SLAVE.
	The index of a CONFIGURABLE-FRAME element in LIN-ORDERED-CONFIGURABLE-FRAMES is determined by the INDEX parameter of the element. The index of a CONFIGURABLE-FRAME element in LIN-CONFIGURABLE-FRAMES is determined by its position, starting with zero as the index of the first element.

Configuration parameters	Mapping description
	Each of the `CONFIGURABLE-FRAME` elements in the list owns an `IDENTIFIER`, which is stored in the `FRAME-TRIGGERING` referencing the same `FRAME` that is referenced by the `CONFIGURABLE-FRAME`. The `PID` of a `CONFIGURABLE-FRAME` is its `IDENTIFIER` plus two parity bits. For details on the parity bits refer to [2]. The bytes `[4 .. 7]` are filled with the `PID` of the `CONFIGURABLE-FRAME` at index `[START-INDEX .. START-INDEX + 3]`. If no valid `FRAME-PID` or `CONFIGURABLE-FRAME` is found at a given `INDEX`, `0xff` is configured. ► `CONDITIONAL-CHANGE-NAD`: `CONFIGURED-NAD, 0x06, 0xb3, ID, BYTE, MASK, INVERT, NEW-NAD`. ► `DATA-DUMP-ENTRY`: `CONFIGURED-NAD, 0x06, 0xb4, BYTE-VALUES/BYTE-VALUE[0], BYTE-VALUES/BYTE-VALUE[1], BYTE-VALUES/BYTE-VALUE[2], BYTE-VALUES/BYTE-VALUE[3], BYTE-VALUES/BYTE-VALUE[4]`. If the number of `BYTE-VALUE` elements is not exactly five, an error is reported. If any of the values required to fill the byte array cannot be retrieved, an error is reported. For each byte of the array, a `LinIfFixedFrameSduByte` is created. The container name is `LinIfFixedFrameSduByte_<byte position within array>`. `LinIfFixedFrameSduBytePos` is set to the byte position within the array. `LinIfFixedFrameSduByteVal` is set to `<byte value>`, where only the eight least significant bits are considered.
`LinIfChannel/LinIfFrame/LinIfPduDirection`	For `LIN-FRAME-TRIGGERING` elements referencing a `LIN-SPORADIC-FRAME` and which have their `IDENTIFIER` set to values other than 60 or 61, the choice container is set to `LinIfRxPdu` (received by `LIN-MASTER`) or `LinIfTxPdu` (sent by `LIN-MASTER`). The name of `LinIfRxPdu`/`LinIfTxPdu` is set to `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU belonging to the PDU instance (see Section 3.3.1.3, "Instance handling") of the `LIN-FRAME-TRIGGERING`.

Configuration parameters	Mapping description
	`LinIfRxPduRef`/`LinIfTxPduRef` references the container in the `EcuC` module configuration, which has been created for the PDU instance of the `LIN-FRAME-TRIGGERING`. If the number of PDU instances for the `LIN-FRAME-TRIGGERING` is not exactly one, no properties are exported for `LinIfRxPdu`/`LinIfTxPdu` and a warning is reported. `LinIfUserRxIndicationUL`/`LinIfUserTxUL` is set to `PDUR` for `I-SIGNAL-PDU` elements, `MULTIPLEXED-I-PDU` elements, `CONTAINER-I-PDU` elements, `GENERAL-PURPOSE-I-PDU` elements, `DCM-I-PDU` elements, and for `USER-DEFINED-I-PDU` elements, which either have a `CATEGORY` other than `XCP` or are routed (see <u>Section 3.3.6, "PDU routing"</u>), or to `CDD` for received or sent `USER-DEFINED-I-PDU` elements that have their `CATEGORY` set to `XCP` and for all other PDU types. For `LIN-FRAME-TRIGGERING` elements not sent/received by the `LIN-MASTER`, but referenced by a `TABLE-ENTRY`, the `LinIfPduDirection` choice container is set to `LinIfSlaveToSlavePdu`. For all other `TABLE-ENTRY` types, the choice container is set to `LinIfInternalPdu`.
`LinIfChannel/LinIfFrame/LinIfSubstitutionFrame`	For every `LIN-UNCONDITIONAL-FRAME` referenced by a `LIN-SPORADIC-FRAME` via `SUBSTITUTED-FRAME-REF`, or by a `LIN-EVENT-TRIGGERED-FRAME` via `LIN-UNCONDITIONAL-FRAME-REF`, a `LinIfSubstitutionFrame` container is created. The container name is the same as the `LinIfFrame` created for the referenced `LIN-UNCONDITIONAL-FRAME`. `LinIfSubstitutionFrameRef` references the container created for the referenced `LIN-UNCONDITIONAL-FRAME`. `LinIfFramePriority` is set to the position of the referring element (`SUBSTITUTED-FRAME-REF`/`LIN-UNCONDITIONAL-FRAME-REF`) within its parent element.
`LinIfChannel/LinIfScheduleTable`	For each `LIN-SCHEDULE-TABLE` a `LinIfScheduleTable` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `LIN-SCHEDULE-TABLE`. `LinIfScheduleTableName` is set to `SHORT-NAME`. `LinIfRunMode` is set depending on `RUN-MODE`: ▶ `RUN_CONTINUOUS` for `RUN-CONTINUOUS` ▶ `RUN_ONCE` for `RUN-ONCE` `LinIfResumePosition` is set depending on `RESUME-POSITION`:

Configuration parameters	Mapping description
	▶ CONTINUE_AT_IT_POINT for CONTINUE-AT-IT-POSITION ▶ START_FROM_BEGINNING for START-FROM-BEGINNING
LinIfChannel/LinIfScheduleTable/LinIfEntry	For each TABLE-ENTRY a LinIfEntry container is created. The container name is Entry<auto incremented number>. LinIfDelay is set to DELAY. LinIfEntryIndex is set to POSITION-IN-TABLE. LinIfFrameRef references the corresponding LinIfFrame container. LinIfCollisionResolvingRef is only set for TABLE-ENTRY elements which reference a LIN-FRAME-TRIGGERING referencing a LIN-EVENT-TRIGGERED-FRAME. It references the LinIfScheduleTable created for the LIN-SCHEDULE-TABLE referenced via COLLISION-RESOLVING-SCHEDULE-REF.

3.4.37. LinSM

Configuration parameters	Mapping description
LinSMConfigSet/LinSMChannel	For every LIN-CLUSTER which belongs to the imported ECU-INSTANCE, a LinSMChannel container is created. The container name is <PREFIX><name>, where <name> is the SHORT-NAME of the LIN-CLUSTER. LinSMComMNetworkHandleRef references the ComMChannel container in the ComM module configuration, which has been created for the LIN-CLUSTER.
LinSMConfigSet/LinSMChannel/LinSMSchedule	For every LIN-SCHEDULE-TABLE of the LIN-CLUSTER's first LIN-PHYSICAL-CHANNEL, a LinSMSchedule container is created. The container name is <PREFIX><name>, where <name> is the SHORT-NAME of the LIN-SCHEDULE-TABLE. LinSMScheduleIndexRef references the LinIfScheduleTable container in the LinIf module configuration, which has been created for the LIN-SCHEDULE-TABLE.

3.4.38. LinTp

Configuration parameters	Mapping description
`LinTpGlobalConfig`	The following parameters are set using the first `LIN-TP-NODE` which belongs to the imported `ECU-INSTANCE`. A `LIN-TP-NODE` belongs to the `ECU-INSTANCE` if at least one of the `COMMUNICATION-CONNECTOR` elements it references also belongs to this `ECU-INSTANCE`. `LinTpP2Timing` is set to `P-2-TIMING`. `LinTpP2Max` is set to `P-2-MAX`. `LinTpMaxNumberOfRespPendingFrames` is set to `MAX-NUMBER-OF-RESP-PENDING-FRAMES`. If inconsistencies among multiple `LIN-TP-NODE` elements are detected, a warning is reported.
`LinTpGlobalConfig/LinTpRxNSdu`	For every `PDU` which is referenced by a `LIN-TP-CONNECTION` which belongs to the imported `ECU-INSTANCE`, and which is also received or routed by the `ECU-INSTANCE`, a `LinTpRxNSdu` container is created. For more information, see Section 3.3.6, "PDU routing". The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `PDU`. A `LIN-TP-CONNECTION` belongs to the imported `ECU-INSTANCE` if at least one transmitting or receiving `LIN-TP-NODE` of this `LIN-TP-CONNECTION` references a `COMMUNICATION-CONNECTOR` which is also referenced by the imported `ECU-INSTANCE`. If multiple `LIN-TP-CONNECTION` elements reference the same `PDU`, only one `LinTpRxNSdu` container is created. The parameters `TimeoutAs`, `TimeoutCr`, and `TimeoutCs` of these `LIN-TP-CONNECTION` elements are checked for consistency. If inconsistencies are detected, a warning is reported. `LinTpRxNSduPduRef` references the corresponding container in the `EcuC` module configuration. `LinTpRxNSduChannelRef` references the container created for the `LIN-PHYSICAL-CHANNEL` of the `LIN-TP-CONNECTION` in the `LinIf` module configuration. `LinTpRxNSduTpChannelRef` references the `LinTpChannelConfig` container created for the `LIN-PHYSICAL-CHANNEL` of the `LIN-TP-CONNECTION`.

Configuration parameters	Mapping description
	`LinTpRxNSduNad` is set to `TP-ADDRESS` referenced by the `LIN-TP-NODE` referenced via `TRANSMITTER-REF`. `LinTpDl` is set to `LENGTH`. `LinTpNcr` is set to `TIMEOUT-CR`.
`LinTpGlobalConfig/LinTpTxNSdu`	For every `PDU` which is referenced by a `LIN-TP-CONNECTION` which belongs to the imported `ECU-INSTANCE`, and which is also sent or routed by the `ECU-INSTANCE`, a `LinTpTxNSdu` container is created. For more information, see Section 3.3.6, "PDU routing". The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `PDU`. If multiple `LIN-TP-CONNECTION` elements reference the same `PDU`, only one `LinTpTxNSdu` container is created. The parameters `TimeoutAs`, `TimeoutCr`, and `TimeoutCs` of these `LIN-TP-CONNECTION` elements are checked for consistency. If inconsistencies are detected, a warning is reported. `LinTpTxNSduPduRef` references the corresponding container in the `EcuC` module configuration. `LinTpTxNSduChannelRef` references the container created for the `LIN-PHYSICAL-CHANNEL` of the `LIN-TP-CONNECTION` in the `LinIf` module configuration. `LinTpTxNSduTpChannelRef` references the `LinTpChannelConfig` container created for the `LIN-PHYSICAL-CHANNEL` of the `LIN-TP-CONNECTION`. `LinTpTxNSduNad` is set to `TP-ADDRESS` referenced via `MULTICAST-REF`. If no `MULTICAST-REF` exists, the `TP-ADDRESS` of the first `LIN-TP-NODE` referenced via `RECEIVER-REFS` is used. If more than one `MULTICAST-REF` exists, a warning is issued. `LinTpNas` is set to `TIMEOUT-AS`. `LinTpNcs` is set to `TIMEOUT-CS`.
`LinTpGlobalConfig/LinTpChannelConfig`	For the `LIN-PHYSICAL-CHANNEL` which belongs to the `LIN-TP-CONNECTION` a `LinTpChannelConfig` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `LIN-PHYSICAL-CHANNEL`. A `LIN-PHYSICAL-CHANNEL` belongs to a `LIN-TP-CONNECTION` if it is part of the `COMMUNICATION-CLUSTER` referenced by the parent `LIN-TP-CONFIG`.

Configuration parameters	Mapping description
	`LinTpDropNotRequestedNad` is set to `DROP-NOT-REQUESTED-NAD`. If inconsistencies are detected among `DROP-NOT-REQUESTED-NAD` values of multiple `LIN-TP-CONNECTION` elements which belong to the same `LIN-PHYSICAL-CHANNEL`, a warning is reported.

3.4.39. Nm

Configuration parameters	Mapping description
`NmChannelConfig`	For every `CAN-CLUSTER`, `J-1939-CLUSTER`, `FLEXRAY-CLUSTER`, and `ETHERNET-PHYSICAL-CHANNEL` for which an Nm channel container has been created in `CanNm`, `J1939Nm`, `FrNm`, or `UdpNm`, an `NmChannelConfig` container is created. For details on Nm channel container creation in the bus-specific NM modules, see [Section 3.4.4, "CanNm"](#), [Section 3.4.31, "J1939Nm"](#), [Section 3.4.25, "FrNm"](#), and [Section 3.4.48, "UdpNm"](#).
	The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `NM-CLUSTER` or, if the `NmChannelConfig` has been created for an `ETHERNET-PHYSICAL-CHANNEL`, the `ETHERNET-PHYSICAL-CHANNEL`'s `SHORT-NAME`.
	The `NM-ECU` used for configuring some of the `NmChannelConfig` parameters is the `NM-ECU` which the first `NM-NODE` that is associated to the configured `ECU-INSTANCE` and to the `NM-CLUSTER` references via `NM-IF-ECU-REF`.
	`NmNodeDetectionEnabled` is set to `NM-CLUSTER/NM-NODE-DETECTION-ENABLED`, or to `NM-ECU/NM-NODE-DETECTION-ENABLED` if `NM-CLUSTER/NM-NODE-DETECTION-ENABLED` is not available.
	`NmNodeIdEnabled` is set to `NM-CLUSTER/NM-NODE-ID-ENABLED`, or to `NM-ECU/NM-NODE-ID-ENABLED` if `NM-CLUSTER/NM-NODE-ID-ENABLED` is not available.
	`NmRepeatMsgIndEnabled` is set to `NM-CLUSTER/NM-REPEAT-MSG-IND-ENABLED`, or to `NM-ECU/NM-REPEAT-MSG-IND-ENABLED` if `NM-CLUSTER/NM-REPEAT-MSG-IND-ENABLED` is not available.
	`NmSynchronizingNetwork` is set to `NM-SYNCHRONIZING-NETWORK`.
	`NmChannelSleepMaster` is set to `NM-CHANNEL-SLEEP-MASTER`.

Configuration parameters	Mapping description
	`NmComMChannelRef` references the `ComMChannel` container in the `ComM` module that corresponds to this `NmChannelConfig` container. `NmBusType/NmStandardBusNmConfig/NmStandardBusType` is configured for the following sub types of `NM-CLUSTER`: ▶ `NM_BUSNM_CANNM` for `CAN-NM-CLUSTER` elements. ▶ `NM_BUSNM_FRNM` for `FLEXRAY-NM-CLUSTER` elements. ▶ `NM_BUSNM_UDPNM` for `UDP-NM-CLUSTER` elements. If the `NM-CLUSTER` is of type `J-1939-NM-CLUSTER`, `NmBusType/NmGenericBusNmConfig/NmGenericBusNmPrefix` is set to `J1939Nm`. The following parameters are set using the `NM-COORDINATOR` which belongs to the first `NM-NODE` belonging to the `NM-CLUSTER` and the imported `ECU-INSTANCE`. An `NM-COORDINATOR` belongs to an `NM-NODE` if it is aggregated by the `NM-ECU` of the `NM-NODE` and if the `NM-COORDINATOR` references this `NM-NODE`. `NmShutdownDelayTimer` is set to `NM-SHUTDOWN-DELAY-TIMER`. The following parameters are set using the first `NM-NODE` which belongs to the `NM-CLUSTER` and the imported `ECU-INSTANCE`. `NmPassiveModeEnabled` is set to `NM-PASSIVE-MODE-ENABLED`. `NmActiveCoordinator` is set depending on the value of `NM-COORDINATOR-ROLE`: ▶ `true` for `ACTIVE`. ▶ `false` for `PASSIVE`. `NmCoordClusterIndex` is set to `NM-COORD-CLUSTER`. If `NM-COORD-CLUSTER` is not available, `NmCoordClusterIndex` is set to `INDEX` of the `NM-COORDINATOR`.
`NmGlobalConfig`	The following parameters are set using the first `NM-ECU` of all `NM-CLUSTER` elements belonging to the imported `ECU-INSTANCE`.
`NmGlobalConfig/NmGlobalProperties`	`NmCycletimeMainFunction` is set to `NM-CYCLETIME-MAIN-FUNCTION`.
`NmGlobalConfig/NmGlobalFeatures`	`NmUserDataEnabled` is set to `NM-USER-DATA-ENABLED`. `NmComUserDataSupport` is set to true if any `NmUserDataPdu` container has been created in the `EcuC` module configuration as described in Section 3.4.14,

Configuration parameters	Mapping description
	"EcuC", or if any NM-CLUSTER linked to the imported ECU-INSTANCE has its NM-PNC-PARTICIPATION either not defined or set to true. NmPduRxIndicationEnabled is set to NM-PDU-RX-INDICATION-ENABLED. NmStateChangeIndEnabled is set to NM-STATE-CHANGE-IND-ENABLED. NmRemoteSleepIndEnabled is set to NM-REMOTE-SLEEP-IND-ENABLED. NmBusSynchronizationEnabled is set to NM-BUS-SYNCHRONIZA-TION-ENABLED. NmCoordinatorSupportEnabled is set to true if NM-COORDINATOR elements exist for the NM-ECU elements. NmCarWakeUpRxEnabled is set to true, if at least one NM-CLUSTER belonging to the imported ECU-INSTANCE has NM-CAR-WAKE-UP-RX-ENABLED set to true. If all NM-CLUSTER elements have NM-CAR-WAKE-UP-RX-ENABLED set to false, NmCarWakeUpRxEnabled is also set to false. Otherwise NmCarWake-UpRxEnabled is not set. The following parameters are set using the first NM-COORDINATOR of all NM-ECU elements of all NM-CLUSTER elements which belong to the imported ECU-INSTANCE. NmGlobalCoordinatorTime is set to NM-GLOBAL-COORDINATOR-TIME. NmCoordinatorSyncSupport is set to NM-COORD-SYNC-SUPPORT.

3.4.40. PduR

Configuration parameters	Mapping description
PduRRoutingTables/PduRRoutingTable/PduRRoutingPath	For every PDU instance (see Section 3.3.1.3, "Instance handling") sent, received or routed (see Section 3.3.6, "PDU routing") by the imported ECU-INSTANCE, a PduRRoutingPath container is created. For every PDU referenced by a TP-CONNECTION belonging to the imported ECU-INSTANCE, a PduRRoutingPath container is created as well. The container name is <PREFIX><name>, where <name> is the SHORT-NAME of the PDU. For NM-PDUs the container name is <PREFIX><name>_NmComUserData<INSTSUFFIX>.

Configuration parameters	Mapping description
	Also, for all `Pdu` containers created for PNC-enabled `NM-CLUSTER` elements in the `EcuC` module (see [Section 3.4.14, "EcuC"](#)), a `PduRRoutingPath` is created. The container name is the name of the `Pdu` container in the `EcuC` module, and both `PduRSrcPduRef` and `PduRDestPduRef` reference this container. For the following PDU instances no `PduRRoutingPath` is created: ▶ PDU instances representing `N-PDU` elements unless an `I-PDU-MAPPING` exists for them. ▶ PDU instances representing `NM-PDU` elements for which no `NmUserDataPdu` container has been created, see [Section 3.4.14, "EcuC"](#). ▶ PDU instances sent by the imported `ECU-INSTANCE` for which also an `I-PDU-MAPPING` exists which references the PDU's `PDU-TRIGGERING` via `TARGET-I-PDU-REF`. ▶ PDU instances representing `I-SIGNAL-I-PDU` elements which do not contain any signal which the `ECU-INSTANCE` sends or receives and for which no `I-PDU-MAPPING` exists. If such a PDU is detected, a warning is reported. ▶ PDU instances representing `DCM-I-PDU` elements that are referenced via `DO-IP-TP-CONNECTION` elements and which are sent and have their `DIAG-PDU-TYPE` value set to `DIAG-REQUEST`, or which are received and have their `DIAG-PDU-TYPE` value set to `DIAG-RESPONSE`. `PduRSrcPdu/PduRSrcPduRef` references the PDU container in the `EcuC` module configuration. For `NM-PDU` elements it references the `NmUserDataPdu` container in the `EcuC` module which has been created for this `NM-PDU`. The name of `PduRSrcPdu` is set to `<PREFIX><name><INSTSUFFIX>_S`, where `<name>` is the `SHORT-NAME` of the PDU.
`PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRDestPdu`	If the PDU instance is routed locally (i.e. the `ECU-INSTANCE` processes the data contained in the PDU), a `PduRDestPdu` container is created. Received PDUs which are referenced by `TP-CONNECTION` elements and by the `SOURCE-I-PDU-REF` of at least one `I-PDU-MAPPING` additionally require that one of the following conditions is met: ▶ The `TP-CONNECTION` element of the PDU references a multicast `TP-ADDRESS` ▶ The `TP-CONNECTION` element of the PDU is referenced by a `DIAGNOSTIC-CONNECTION` which in turn is referenced by a `DIAGNOSTIC-SERVICE-TABLE` that also references the configured `ECU-INSTANCE` The container name is the name of its `PduRRoutingPath` parent container plus the suffix `_D`. `PduRDestPdu/PduRDestPduRef` references the same `EcuC` PDU container as `PduRSrcPdu/PduRSrcPduRef` of the parent container.

Configuration parameters	Mapping description
	The PDU instance is routed via gateway if two conditions hold: ▶ The `ECU-INSTANCE` receives the PDU instance ▶ `I-PDU-MAPPING` elements exist which reference the PDU instance's `PDU-TRIGGERING` via `SOURCE-I-PDU-REF` If the PDU instance is routed via gateway, one `PduRDestPdu` container is created for each `I-PDU-MAPPING`. The sent PDU instance that corresponds to the `PDU-TRIGGERING` that is referenced via `I-PDU-MAPPING/TARGET-I-PDU/TARGET-I-PDU-REF` is used to configure the `PduRDestPdu` container. If the `I-PDU-MAPPING` contains a valid `PDUR-TP-CHUNK-SIZE` subelement and the `PduRRoutingPath` does not contain additional sibling `PduRDestPdu` containers, the content of `PDUR-TP-CHUNK-SIZE` is used to configure `PduRDestPdu/PduRTpThreshold`. `PduRDestPduRef` references the `EcuC` container created for the sent PDU instance.
`PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRDestPdu/PduRDefaultValue/PduRDefaultValueElement`	For every `DEFAULT-VALUE-ELEMENT` of the `TARGET-I-PDU`, a `PduRDefaultValueElement` container is created. The name of the container is `PduRDefaultValueElement_<ELEMENT-POSITION>`. If `ELEMENT-POSITION` >= PDU's `LENGTH` a warning is reported and no `PduRDefaultValueElement` is created. If multiple `DEFAULT-VALUE-ELEMENT` have the same `ELEMENT-POSITION`, only one `PduRDefaultValueElement` container is created and a warning is reported. If `ELEMENT-BYTE-VALUE` does not lie within the interval [0 .. 255], a warning is reported and no `PduRDefaultValueElement` is created. `PduRDefaultValueElement` is set to `ELEMENT-BYTE-VALUE`. `PduRDefaultValueElementBytePosition` is set to `ELEMENT-POSITION`.
`PduRRoutingTables/PduRRoutingPathGroup`	For every `PDUR-I-PDU-GROUP` referenced by the imported `ECU-INSTANCE` via `ASSOCIATED-PDUR-I-PDU-GROUP-REF`, a `PduRRoutingPathGroup` container is created. The name of the container is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `PDUR-I-PDU-GROUP`. For each `PDU-TRIGGERING-REF` of the `PDUR-I-PDU-GROUP` a `PduRDestPduRef` reference is created. It refers to the `PduRDestPdu` container that has been created for the `PDU-TRIGGERING` referenced by `PDU-TRIGGERING-REF`.

3.4.41. Sd

In its R19/11 release, AUTOSAR introduced a new communication and service model for Ethernet that super-sedes the model that was present in AUTOSAR 4.4 an earlier releases. The AUTOSAR R19-11 schema allows to provide Ethernet communication and service data using either model. Both models are supported for the configuration of the `Sd` module. It is, however, strongly discouraged to mix AUTOSAR 4.4 and R19-11 communication and service model concepts in a single Ecu Extract or System Description.

This chapter describes the configuration for both AUTOSAR 4.4 and R19-11 model, [5] and [6] provide in their "Ethernet specific description" chapters details on the differences of both models.

In AUTOSAR 4.4, a `PROVIDED-SERVICE-INSTANCE`/`CONSUMED-SERVICE-INSTANCE` belong to a `SOCK-ET-ADDRESS` if the `SOCKET-ADDRESS` contains an `APPLICATION-ENDPOINT` that in turn contains the `PROVIDED-SERVICE-INSTANCE`/`CONSUMED-SERVICE-INSTANCE`.

In AUTOSAR R19-11, a `PROVIDED-SERVICE-INSTANCE` belongs to a `SOCKET-ADDRESS` if it references the `APPLICATION-ENDPOINT` of the `SOCKET-ADDRESS` via `PROVIDED-SERVICE-INSTANCE/LOCAL-AD-DRESSS`. `CONSUMED-SERVICE-INSTANCE` belongs to a `SOCKET-ADDRESS` if it references the `APPLI-CATION-ENDPOINT` of the `SOCKET-ADDRESS` via `CONSUMED-SERVICE-INSTANCE/LOCAL-ADDRESSS` or if it contains one or more `CONSUMED-EVENT-GROUP` elements which reference the `APPLICATION-ENDPOINT` of the `SOCKET-ADDRESS` via `CONSUMED-EVENT-GROUP/EVENT-MULTICAST-ADDRESSS`.

Configuration parameters	Mapping description
`SdConfig/SdInstance`	For each `ETHERNET-PHYSICAL-CHANNEL` for which the following conditions hold, one `SdInstance` container is created. ▶ The imported `ECU-INSTANCE` sends and/or receives PDUs on the `ETHERNET-PHYSICAL-CHANNEL`. ▶ The imported `ECU-INSTANCE` owns on the `ETHERNET-PHYSICAL-CHANNEL` at least one *local* `SOCKET-ADDRESS` to which in turn belongs at least one `PROVIDED-SERVICE-INSTANCE` or at least one `CONSUMED-SERVICE-INSTANCE`. For more information about `SOCKET-ADDRESS`, see Section 3.4.43, "SoAd". The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `ETHERNET-PHYSICAL-CHANNEL`. `SdInstanceHostname` is configured by retrieving the `FULLY-QUALIFIED-DOMAIN-NAME` elements of all `NETWORK-ENDPOINT` elements that belong to the `ETHERNET-PHYSICAL-CHANNEL` and to an `ETHERNET-COMMUNI-CATION-CONNECTOR` of the imported `ECU-INSTANCE`.

Configuration parameters	Mapping description
	A `NETWORK-ENDPOINT` belongs to the `ETHERNET-PHYSICAL-CHANNEL` and to an `ETHERNET-COMMUNICATION-CONNECTOR` if there is a `SOCKET-ADDRESS` contained in the `ETHERNET-PHYSICAL-CHANNEL` which references the `NETWORK-ENDPOINT` via `SOCKET-ADDRESS/NETWORK-ENDPOINT-REF` and which contains an `APPLICATION-ENDPOINT` that in turn references the `ETHERNET-COMMUNICATION-CONNECTOR` of the `ECU-INSTANCE` either via `APPLICATION-ENDPOINT/CONNECTOR-REF` or via `APPLICATION-END-POINT/MULTICAST-CONNECTOR-REFS/MULTICAST-CONNECTOR-REF`. If the collection yields exactly one distinct `FULLY-QUALIFIED-DOMAIN-NAME`, it is used to configure `SdInstanceHostname`. If none or more than one distinct `FULLY-QUALIFIED-DOMAIN-NAME` elements are found, `SdInstanceHostname` is not set.
`SdConfig/SdInstance/SdInstanceTxPdu`	If an `Sd` PDU exists which is sent by the imported `ECU-INSTANCE` via a *local* `SOCKET-ADDRESS` of the `ETHERNET-PHYSICAL-CHANNEL`, an `SdInstanceTxPdu` container is created. `SdTxPduRef` references the corresponding container in the `EcuC` module configuration. A PDU is considered an `Sd` PDU if it is a `GENERAL-PURPOSE-PDU` and its `CATEGORY` is set to `SD`.
`SdConfig/SdInstance/SdInstanceUnicastRxPdu`	If an `Sd` PDU exists which is received by the imported `ECU-INSTANCE` via a local unicast *local* `SOCKET-ADDRESS` of the `ETHERNET-PHYSICAL-CHANNEL`, an `SdInstanceUnicastRxPdu` container is created. `SdRxPduRef` references the corresponding container in the `EcuC` module configuration.
`SdConfig/SdInstance/SdInstanceMulticastRxPdu`	If an `Sd` PDU exists which is received by the imported `ECU-INSTANCE` a local multicast *local* `SOCKET-ADDRESS` of the `ETHERNET-PHYSICAL-CHANNEL`, an `SdInstanceMulticastRxPdu` container is created. `SdRxPduRef` references the corresponding container in the `EcuC` module configuration.
`SdConfig/SdInstance/SdSubscribeEventgroupRetryDelay`	In AUTOSAR R19-11 and later models, `SdSubscribeEventgroupRetryDelay` is configured by retrieving all `SOMEIP-SD-CLIENT-EVENT-GROUP-TIMING-CONFIG` elements of all `CONSUMED-EVENT-GROUP` elements that are associated with the imported `ECU-INSTANCE` in the VLAN that corresponds to the `SdInstance` container. If all retrieved `SOMEIP-SD-CLIENT-EVENT-GROUP-TIMING-CONFIG` elements contain the same value in `SUBSCRIBE-EVENTGROUP-RETRY-DELAY`, that value is used to configure `SdSubscribeEventgroupRetryDelay`.
`SdConfig/SdInstance/SdSubscribeEventgroupRetryMax`	In AUTOSAR R19-11 and later models, `SdSubscribeEventgroupRetryMax` is configured by retrieving all `SOMEIP-SD-CLIENT-EVENT-GROUP-TIMING-CONFIG` elements of all `CONSUMED-EVENT-GROUP` elements that are associated with the imported `ECU-INSTANCE` in the VLAN that corresponds to the `SdInstance` container. If all retrieved `SOMEIP-SD-CLIENT-EVENT-GROUP-`

Configuration parameters	Mapping description
	`TIMING-CONFIG` elements contain the same value in `SUBSCRIBE-EVENT-GROUP-RETRY-MAX`, that value is used to configure `SdSubscribeEventgroupRetryMax`.
`SdConfig/SdInstance/SdServerService`	For every local `SOCKET-ADDRESS` of the imported `ECU-INSTANCE` all its `PROVIDED-SERVICE-INSTANCE` elements are retrieved. For each of these `PROVIDED-SERVICE-INSTANCE` elements, an `SdServerService` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `PROVIDED-SERVICE-INSTANCE`. `SdServerServiceInstanceId` is set to `INSTANCE-IDENTIFIER`. `SdServerServiceId` is set to `SERVICE-IDENTIFIER`. `SdServerServiceMajorVersion` is set to `SD-SERVER-CONFIG/SERVER-SERVICE-MAJOR-VERSION` in AUTOSAR 4.4 models and to `MAJOR-VERSION` in AUTOSAR R19-11 models. `SdServerServiceMinorVersion` is set to `SD-SERVER-CONFIG/SERVER-SERVICE-MINOR-VERSION` in AUTOSAR 4.4 models and to `MINOR-VERSION` in AUTOSAR R19-11 models. `SdServerServiceAutoAvailable` is set to `PROVIDED-SERVICE-INSTANCE/AUTO-AVAILABLE` in AUTOSAR R19-11 models. `SdServerServiceTimerRef` references the `SdServerTimer` container created either for the AUTOSAR 4.4 model `SD-SERVER-CONFIG`, or for the AUTOSAR R19-11 `SOMEIP-SD-SERVER-SERVICE-INSTANCE-CONFIG`. If the *local* `SOCKET-ADDRESS` is connected to a `SoAdSocketConnectionGroup` of type `SoAdSocketUdp` and the `SOCKET-ADDRESS` is a unicast address, `SdServerServiceUdpRef` references the `SoAdSocketConnectionGroup`. For more information about `SOCKET-ADDRESS`, see [Section 3.4.43, "SoAd"](#). If the *local* `SOCKET-ADDRESS` is connected to a `SoAdSocketConnectionGroup` of type `SoAdSocketTcp` and the `SOCKET-ADDRESS` is a unicast address, `SdServerServiceTcpRef` references the `SoAdSocketConnectionGroup`.
`SdConfig/SdInstance/SdServerService/SdServerCapabilityRecord`	For each `CAPABILITY-RECORDS/TAG-WITH-OPTIONAL-VALUE`, an `SdServerCapabilityRecord` container is created. The container name is `SdServerCapabilityRecord_<auto incremented number>`.

Configuration parameters	Mapping description
	`SdServerCapabilityRecordKey` is set to `KEY`. `SdServerCapabilityRecordValue` is set to `VALUE`.
`SdConfig/SdInstance/SdServerService/SdEventHandler`	For each `EVENT-HANDLER` of the `PROVIDED-SERVICE-INSTANCE`, an `SdEventHandler` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `EVENT-HANDLER`. `SdEventHandlerEventGroupId` is set to `EVENT-GROUP-IDENTIFIER` of the first `CONSUMED-EVENT-GROUP` that is referenced by the `EVENT-HANDLER` in AUTOSAR 4.4 models, and to the `EVENT-GROUP-IDENTIFIER` of the `EVENT-HANDLER` in AUTOSAR R19-11 models. `SdEventHandlerMulticastThreshold` is set to `EVENT-HANDLER/MULTI-CAST-THRESHOLD`. If any of the AUTOSAR 4.4 `SO-AD-ROUTING-GROUP` elements referenced by the `EVENT-HANDLER` or any of the AUTOSAR 19-11 `PDU-ACTI-VATION-ROUTING-GROUP` elements aggregated in the `EVENT-HANDLER` has its `EVENT-GROUP-CONTROL-TYPE` set to `ACTIVATION-MULTICAST`, `SdEventHandlerMulticast/SdEventActivationRef` references the `SoAdRout-ingGroup` container that has been created for this `SO-AD-ROUTING-GROUP` or `PDU-ACTIVATION-ROUTING-GROUP`. In AUTOSAR 4.4, the `APPLICATION-ENDPOINT` that transmits the multicast PDUs of an `EVENT-HANDLER` is either the `APPLICATION-ENDPOINT` that `EVENT-HANDLER/APPLICATION-ENDPOINT-REF` refers to, or, if `EVENT-HAN-DLER/APPLICATION-ENDPOINT-REF` does not exist, the `APPLICATION-END-POINT` that contains the `EVENT-HANDLER`. In AUTOSAR 19-11 the `APPLI-CATION-ENDPOINT` is the one that the parent `PROVIDED-SERVICE-INSTANCE` of the `EVENT-HANDLER` references via `PROVIDED-SERVICE-INSTANCE/LO-CAL-UNICAST-ADDRESSS` and which represents a Udp address. `SdMulti-castEventSoConRef` references a *multicast* `SoAdSocketConnection` container in the `SoAd` module if the `APPLICATION-ENDPOINT` meets the following conditions: ▶ The `APPLICATION-ENDPOINT` is aggregated by a *local* `SOCKET-ADDRESS` of the imported `ECU-INSTANCE`. ▶ The `SoAd` configuration contains a `SoAdSocketConnectionGroup` container that uses the `SOCKET-ADDRESS` as *local* address. ▶ One of the following conditions is met:

Configuration parameters	Mapping description
	► The `SoAdSocketConnectionGroup` configuration contains a single `SoAdSocketConnection` that is associated with a *remote multicast* `SOCKET-ADDRESS`. ► In AUTOSAR 4.4 models, the `SoAdSocketConnectionGroup` configuration contains two or more `SoAdSocketConnection` containers that are associated with a *remote multicast* `SOCKET-ADDRESS`, but only one of the `SoAdSocketConnection` containers refers to the *remote multicast* `SOCKET-ADDRESS` that all `CONSUMED-EVENT-GROUP` elements of the `EVENT-HANDLER` refer to as well. The *multicast* `SOCKET-ADDRESS` of a `CONSUMED-EVENT-GROUP` is the `SOCKET-ADDRESS` that contains the `APPLICATION-ENDPOINT` that the `CONSUMED-EVENT-GROUP` refers to via `APPLICATION-END-POINT-REF`. `SdEventHandlerTimerRef` is configured in the same way as `SdServerServiceTimerRef` in AUTOSAR 4.4 models. In AUTOSAR R19-11 models, `SdEventHandlerTimerRef` references the `SdServerTimer` container created for the AUTOSAR R19-11 `SOMEIP-SD-SERVER-EVENT-GROUP-TIMING-CONFIG`. `SdServerCapabilityRecord` is configured in the same way as `SdServerService/SdServerCapabilityRecord` in AUTOSAR 4.4 models. In AUTOSAR R19-11 there are no capability records for `EVENT-HANDLER` elements, hence nothing is configured.
`SdConfig/SdInstance/SdServerService/SdEventHandler/SdEventHandlerUdp`	If the *local* `SOCKET-ADDRESS` is connected to a `SoAdSocketConnectionGroup` of type `SoAdSocketUdp` (see [Section 3.4.43, "SoAd"](#)), an `SdEventHandlerUdp` container is created. If any of the AUTOSAR 4.4 `SO-AD-ROUTING-GROUP` elements or any of the AUTOSAR R19-11 `PDU-ACTIVATION-ROUTING-GROUP` elements of the `EVENT-HANDLER` has its `EVENT-GROUP-CONTROL-TYPE` set to `ACTIVATION-UNICAST` or to `ACTIVATION-AND-TRIGGER-UNICAST`, `SdEventActivationRef` references the `SoAdRoutingGroup` container that has been created for this `SO-AD-ROUTING-GROUP` or `PDU-ACTIVATION-ROUTING-GROUP`. If any of the AUTOSAR 4.4 `SO-AD-ROUTING-GROUP` or any of the AUTOSAR R19-11 `PDU-ACTIVATION-ROUTING-GROUP` elements of the `EVENT-HANDLER` has its `EVENT-GROUP-CONTROL-TYPE` set to `TRIGGER-UNICAST` or to `ACTIVATION-AND-TRIGGER-UNICAST`, `SdEventTriggeringRef` refer-

Configuration parameters	Mapping description
	ences the `SoAdRoutingGroup` container that has been created for this `SO-AD-ROUTING-GROUP`.
`SdConfig/SdInstance/SdServerService/SdEventHandler/SdEventHandlerTcp`	If the *local* `SOCKET-ADDRESS` is connected to a `SoAdSocketConnectionGroup` of type `SoAdSocketTcp` (see [Section 3.4.43, "SoAd"](#)), an `SdEventHandlerTcp` container is created. The parameters `SdEventActivationRef` and `SdEventTriggeringRef` are configured in the same way as the corresponding parameters of `SdEventHandlerUdp`.
`SdConfig/SdInstance/SdServerService/SdProvidedMethods`	`SdServerServiceActivationRef` references the `SoAdRoutingGroup` container that has been created for the first AUTOSAR 4.4 `SO-AD-ROUTING-GROUP` referenced by the `PROVIDED-SERVICE-INSTANCE` or the AUTOSAR 19-11 `PDU-ACTIVATION-ROUTING-GROUP` that the `PROVIDED-SERVICE-INSTANCE` contains in `METHOD-ACTIVATION-ROUTING-GROUPS`.
`SdConfig/SdInstance/SdServerTimer`	`SdServerTimer` are created for different AUTOSAR 4.4 and AUTOSAR R19-11 elements. The container name is `SdServerTimer<auto incremented number>`. ► For all AUTOSAR 4.4 `SD-SERVER-CONFIG` elements, the following parameters are configured: `SdServerTimerInitialOfferDelayMax` is set to `INITIAL-OFFER-BEHAVIOR/INITIAL-DELAY-MAX-VALUE`. `SdServerTimerInitialOfferDelayMin` is set to `INITIAL-OFFER-BEHAVIOR/INITIAL-DELAY-MIN-VALUE`. `SdServerTimerInitialOfferRepetitionBaseDelay` is set to `INITIAL-OFFER-BEHAVIOR/INITIAL-REPETITIONS-BASE-DELAY`. `SdServerTimerInitialOfferRepetitionsMax` is set to `INITIAL-OFFER-BEHAVIOR/INITIAL-REPETITIONS-MAX`. `SdServerTimerOfferCyclicDelay` is set to `OFFER-CYCLIC-DELAY`. `SdServerTimerRequestResponseMaxDelay` is set to `REQUEST-RESPONSE-DELAY/MAX-VALUE`. `SdServerTimerRequestResponseMinDelay` is set to `REQUEST-RESPONSE-DELAY/MIN-VALUE`. `SdServerTimerTTL` is set to `TTL`.

Configuration parameters	Mapping description
	▶ For all AUTOSAR R19-11 `SOMEIP-SD-SERVER-SERVICE-INSTANCE-CONFIG` elements, the following parameters are configured: `SdServerTimerInitialOfferDelayMax` is set to `INITIAL-OFFER-BEHAVIOR/INITIAL-DELAY-MAX-VALUE`. `SdServerTimerInitialOfferDelayMin` is set to `INITIAL-OFFER-BEHAVIOR/INITIAL-DELAY-MIN-VALUE`. `SdServerTimerInitialOfferRepetitionBaseDelay` is set to `INITIAL-OFFER-BEHAVIOR/INITIAL-REPETITIONS-BASE-DELAY`. `SdServerTimerInitialOfferRepetitionsMax` is set to `INITIAL-OFFER-BEHAVIOR/INITIAL-REPETITIONS-MAX`. `SdServerTimerOfferCyclicDelay` is set to `OFFER-CYCLIC-DELAY`. `SdServerTimerRequestResponseMaxDelay` is set to `REQUEST-RESPONSE-DELAY/MAX-VALUE`. `SdServerTimerRequestResponseMinDelay` is set to `REQUEST-RESPONSE-DELAY/MIN-VALUE`. `SdServerTimerTTL` is set to `SERVICE-OFFER-TIME-TO-LIVE`. ▶ For all AUTOSAR R19-11 `SOMEIP-SD-SERVER-EVENT-GROUP-TIMING-CONFIG` elements, the following parameters are configured: `SdServerTimerRequestResponseMaxDelay` is set to `REQUEST-RESPONSE-DELAY/MAX-VALUE`. `SdServerTimerRequestResponseMinDelay` is set to `REQUEST-RESPONSE-DELAY/MIN-VALUE`.
`SdConfig/SdInstance/SdClientService`	For every local `SOCKET-ADDRESS` of the imported `ECU-INSTANCE` all its `CONSUMED-SERVICE-INSTANCE` elements are retrieved. For each of these `CONSUMED-SERVICE-INSTANCE` elements, an `SdClientService` container is created. The container name is `<PREFIX><name>` where `<name>` is the `SHORT-NAME` of the `CONSUMED-SERVICE-INSTANCE`. The parameters `SdClientServiceInstanceId`, `SdClientServiceId`, `SdClientServiceMajorVersion`, `SdClientServiceMinorVersion`, `SdClientServiceTimerRef` and `SdClientCapabilityRecord` are configured in the same way as the corresponding parameters of the `SdServerService` container.

Configuration parameters	Mapping description
	`SdClientServiceAutoRequire` is set to `CONSUMED-SERVICE-INS-TANCE/AUTO-REQUIRE` in AUTOSAR R19-11 models. If the *local* `SOCKET-ADDRESS` is connected to a `SoAdSocketConnection-Group` of type `SoAdSocketUdp` and the `SOCKET-ADDRESS` is a unicast address, `SdClientServiceUdpRef` references this `SoAdSocketConnection-Group`. For more information about `SOCKET-ADDRESS`, see [Section 3.4.43, "SoAd"](#). If the *local* `SOCKET-ADDRESS` is connected to a `SoAdSocketConnection-Group` of type `SoAdSocketTcp` and the `SOCKET-ADDRESS` is a unicast address, `SdClientServiceTcpRef` references this `SoAdSocketConnection-Group`. `SdVersionDrivenFindBehavior` is configured depending on `CONSUMED-SERVICE-INSTANCE/VERSION-DRIVEN-FIND-BEHAVIOR` in AUTOSAR R19-11 models. If `VERSION-DRIVEN-FIND-BEHAVIOR` is set to `EXACT-OR-ANY-MINOR-VERSION`, `EXACT_OR_ANY_MINOR_VERSION` is configured. If `VERSION-DRIVEN-FIND-BEHAVIOR` is set to `MINIMUM-MINOR-VERSION`, `MINIMUM_MINOR_VERSION` is configured. In AUTOSAR R19-11 models, one `SdBlacklistedVersions` sub container is created for each `SOMEIP-SERVICE-VERSION` element in `CONSUMED-SERVICE-INSTANCE/BLACKLISTED-VERSIONS` that holds a valid value in `MINOR-VERSION`. The container name is `BlockedVersion_<csi-name>_<minorVersion>`, where `<csiname>` is the `SHORT-NAME` of the `CONSUMED-SERVICE-INSTANCE` and `<minorVersion>` corresponds to `SOMEIP-SERVICE-VERSION/MINOR-VERSION`. In the container, the parameter `SdBlacklistedMinorVersion` is configured using the value of `SOMEIP-SERVICE-VERSION/MINOR-VERSION`.
`SdConfig/SdIn-stance/SdClientSer-vice/SdConsumedE-ventGroup`	For each `CONSUMED-EVENT-GROUP` of the `CONSUMED-SERVICE-INSTANCE`, an `SdConsumedEventGroup` container is created. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `CONSUMED-EVENT-GROUP`. `SdConsumedEventGroupId` is set to `EVENT-GROUP-IDENTIFIER`. `SdConsumedEventGroupTimerRef` references the `SdClientTimer` container created either for the AUTOSAR 4.4 model `SD-CLIENT-CONFIG`, or for the AUTOSAR R19-11 `SOMEIP-SD-CLIENT-SERVICE-INSTANCE-CONFIG`.

Configuration parameters	Mapping description
	`SdClientCapabilityRecord`: see `SdServerService/SdServerCapabilityRecord`. If any of the AUTOSAR 4.4 `SO-AD-ROUTING-GROUP` elements referenced by the `CONSUMED-EVENT-GROUP` or any of the AUTOSAR 19-11 `PDU-ACTIVATION-ROUTING-GROUP` elements aggregated in the `CONSUMED-EVENT-GROUP` has its `EVENT-GROUP-CONTROL-TYPE` set to `ACTIVATION-UNICAST` or to `ACTIVATION-AND-TRIGGER-UNICAST`, a reference is set up to the `SoAdRoutingGroup` container that has been created for this `SO-AD-ROUTING-GROUP` or `PDU-ACTIVATION-ROUTING-GROUP`. If the *local* `SOCKET-ADDRESS` is connected to a `SoAdSocketConnectionGroup` of type `SoAdSocketUdp`, the reference is `SdConsumedEventGroupUdpActivationRef`, if the `SoAdSocketConnectionGroup` is of type `SoAdSocketTcp` the reference is `SdConsumedEventGroupTcpActivationRef`. For more information about `SOCKET-ADDRESS`, see [Section 3.4.43, "SoAd"](#). If any of the AUTOSAR 4.4 `SO-AD-ROUTING-GROUP` elements or any of the AUTOSAR R19-11 `PDU-ACTIVATION-ROUTING-GROUP` elements of the `CONSUMED-EVENT-GROUP` has its `EVENT-GROUP-CONTROL-TYPE` set to `ACTIVATION-MULTICAST`, `SdConsumedEventGroupMulticastActivationRef` references the `SoAdRoutingGroup` container that has been created for this `SO-AD-ROUTING-GROUP`. `SdConsumedEventGroupMulticastGroupRef` is set up to reference a `SoAdSocketConnectionGroup` container in the `SoAd` module if the following conditions are met: ▶ The `CONSUMED-EVENT-GROUP` references an `APPLICATION-ENDPOINT` via `APPLICATION-ENDPOINT-REF` in AUTOSAR 4.4, or via `EVENT-MULTICAST-ADDRESSS` in AUTOSAR R19-11. The `APPLICATION-ENDPOINT` is aggregated by a *local multicast* `SOCKET-ADDRESS` of the imported `ECU-INSTANCE`. ▶ The `SoAd` configuration contains a `SoAdSocketConnectionGroup` container that uses the `SOCKET-ADDRESS` as *local* address.
`SdConfig/SdInstance/SdClientService/SdConsumedMethods`	`SdClientServiceActivationRef` references the `SoAdRoutingGroup` container that has been created for the first AUTOSAR 4.4 `SO-AD-ROUTING-GROUP` referenced by the `CONSUMED-SERVICE-INSTANCE` or for the first AUTOSAR R19-11 `PDU-ACTIVATION-ROUTING-GROUP` contained in the `CONSUMED-SERVICE-INSTANCE`.

Configuration parameters	Mapping description
`SdConfig/SdInstance/SdClientTimer`	`SdClientTimer` are created for different AUTOSAR 4.4 and AUTOSAR R19-11 elements. The container name is `SdClientTimer<auto incremented number>`. ► For all AUTOSAR 4.4 `SD-CLIENT-CONFIG` elements, the following parameters are configured: `SdClientTimerInitialFindDelayMax` is set to `INITIAL-FIND-BEHAVIOR/INITIAL-DELAY-MAX-VALUE`. `SdClientTimerInitialFindDelayMin` is set to `INITIAL-FIND-BEHAVIOR/INITIAL-DELAY-MIN-VALUE`. `SdClientTimerInitialFindRepetitionsBaseDelay` is set to `INITIAL-FIND-BEHAVIOR/INITIAL-REPETITIONS-BASE-DELAY`. `SdClientTimerInitialFindRepetitionsMax` is set to `INITIAL-FIND-BEHAVIOR/INITIAL-REPETITIONS-MAX`. `SdClientTimerRequestResponseMaxDelay` is set to `REQUEST-RESPONSE-DELAY/MAX-VALUE`. `SdClientTimerRequestResponseMinDelay` is set to `REQUEST-RESPONSE-DELAY/MIN-VALUE`. `SdClientTimerTTL` is set to `TTL`. ► For all AUTOSAR R19-11 `SOMEIP-SD-CLIENT-EVENT-GROUP-TIMING-CONFIG` elements, the following parameters are configured: `SdClientTimerRequestResponseMaxDelay` is set to `REQUEST-RESPONSE-DELAY/MAX-VALUE`. `SdClientTimerRequestResponseMinDelay` is set to `REQUEST-RESPONSE-DELAY/MIN-VALUE`. `SdClientTimerTTL` is set to `TIME-TO-LIVE`. ► For all AUTOSAR R19-11 `SOMEIP-SD-CLIENT-SERVICE-INSTANCE-CONFIG` elements, the following parameters are configured: `SdClientTimerInitialFindDelayMax` is set to `INITIAL-FIND-BEHAVIOR/INITIAL-DELAY-MAX-VALUE`.

Configuration parameters	Mapping description
	`SdClientTimerInitialFindDelayMin` is set to `INITIAL-FIND-BE-HAVIOR/INITIAL-DELAY-MIN-VALUE`. `SdClientTimerInitialFindRepetitionsBaseDelay` is set to `INITIAL-FIND-BEHAVIOR/INITIAL-REPETITIONS-BASE-DELAY`. `SdClientTimerInitialFindRepetitionsMax` is set to `INITIAL-FIND-BEHAVIOR/INITIAL-REPETITIONS-MAX`. `SdClientTimerTTL` is set to `SERVICE-FIND-TIME-TO-LIVE`.

3.4.42. SecOC

The `SecOC` secures or authenticates the payload of a PDU by using a `SECURED-I-PDU` and an associated *payload PDU*. A `SECURED-I-PDU` is associated with a payload PDU if `SECURED-I-PDU/PAYLOAD-REF` refers to a `PDU-TRIGGERING` which in turn refers to the payload PDU.

If the `SECURED-I-PDU` has its `USE-AS-CRYPTOGRAPHIC-I-PDU` parameter either set to `false` or not set at all, the `SECURED-I-PDU` contains the associated payload PDU and the meta-data, i.e. *Message Authentication Code* and *Freshness Value*.

If the `SECURED-I-PDU` has its `USE-AS-CRYPTOGRAPHIC-I-PDU` parameter set to `true`, the `SECURED-I-PDU` mostly contains the security data, while the payload is independently transmitted in a second payload PDU. A `SECURED-I-PDU` that has its `USE-AS-CRYPTOGRAPHIC-I-PDU` set to `true` is referred to as *cryptographic* PDU. The PDU that contains the encrypted or authenticated payload is referred to as *payload* PDU.

Configuration parameters	Mapping description
`SecOC/SecOCTx-PduProcessing, SecOC/SecOCRxPduProcessing`	For every `SECURED-I-PDU` that the imported `ECU-INSTANCE` sends, a `SecOC-TxPduProcessing` container is created, for every `SECURED-I-PDU` that is received by the imported `ECU-INSTANCE`, a `SecOCRxPduProcessing` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `SECURED-I-PDU`. `SecOCPduType` is set to `SECOC_TPPDU` in the following cases: ▶ The `SECURED-I-PDU` is referenced by a `CAN-TP-CONNECTION` via `TP-SDU-REF` ▶ The `SECURED-I-PDU` is referenced by a `FLEXRAY-TP-CONNECTION` via `DIRECT-TP-SDU-REF` or via `REVERSED-TP-SDU-REF`

Configuration parameters	Mapping description
	▶ The `SECURED-I-PDU` is referenced by a `FLEXRAY-AR-TP-CONNECTION` via `DIRECT-TP-SDU-REF` or via `REVERSED-TP-SDU-REF` ▶ The `SECURED-I-PDU` is referenced by a `LIN-TP-CONNECTION` via `LIN-TP-N-SDU-REF` ▶ The `SECURED-I-PDU` is referenced by a `PDU-TRIGGERING`, which in turn is referenced by a `SOMEIP-TP-CONNECTION` via `TP-SDU-REF` ▶ An `ETH-TP-CONNECTION`, a `DO-IP-TP-CONNECTION`, or a `SOMEIP-TP-CONNECTION` refers to the `PDU-TRIGGERING` that references the transmitted `SECURED-I-PDU` and that belongs to the `PHYSICAL-CHANNEL` on which the `I-SIGNAL-I-PDU` is transmitted. In all other cases, the value of `SecOCPduType` is set to `SECOC_IFPDU`. If `AUTHENTICATION-PROPS-REF` refers to a valid `SECURE-COMMUNI-CATION-AUTHENTICATION-PROPS` element, `SecOCAuthInfoTxLength` is set to `SECURE-COMMUNICATION-AUTHENTICATION-PROPS/AUTH-INFO-TX-LENGTH`. Otherwise `SecOCAuthInfoTxLength` is set to `SE-CURE-COMMUNICATION-PROPS/AUTH-INFO-TX-LENGTH`. `SecOCAuthenticationBuildAttempts` is set to `SECURE-COMMUNI-CATION-PROPS/AUTHENTICATION-BUILD-ATTEMPTS`. `SecOCDataId` is set to `SECURE-COMMUNICATION-PROPS/DATA-ID`. `SecOCFreshnessValueId` is set to `SECURE-COMMUNICATION-PROPS/FRESHNESS-VALUE-ID`. If `FRESHNESS-PROPS-REF` refers to a valid `SECURE-COMMUNI-CATION-FRESHNESS-PROPS` element, `SecOCFreshnessValueLength` is set to `SECURE-COMMUNICATION-FRESHNESS-PROPS/FRESHNESS-VAL-UE-LENGTH`. Otherwise `SecOCFreshnessValueLength` is set to `SE-CURE-COMMUNICATION-PROPS/FRESHNESS-VALUE-LENGTH`. If `FRESHNESS-PROPS-REF` refers to a valid `SECURE-COMMUNI-CATION-FRESHNESS-PROPS` element, `SecOCFreshnessValueTxLength` is set to `SECURE-COMMUNICATION-FRESHNESS-PROPS/FRESHNESS-VAL-UE-TX-LENGTH`. Otherwise `SecOCFreshnessValueTxLength` is set to `SE-CURE-COMMUNICATION-PROPS/FRESHNESS-VALUE-TX-LENGTH`. `SecOCUseAuthDataFreshness` is set to `USE-AUTH-DATA-FRESHNESS` of the `I-PDU-PORT` via which the `SECURED-I-PDU` is sent or received.

Configuration parameters	Mapping description

`SecOC/SecOCTx-PduProcessing`	For every `SECURED-I-PDU` that the imported `ECU-INSTANCE` sends, the following parameters are configured in `SecOCTxPduProcessing`:
	
	▶ `SecOCTxAuthenticPduLayer/SecOCTxAuthenticLayerPduRef` is set to reference the `EcuC` configuration container of the PDU that the `SE-CURED-I-PDU` references via `PAYLOAD-REF`. That container represents the PDU that is provided by the upper layer module. The following PDU types are supported:
	
	   ▶ `I-SIGNAL-I-PDU`
	   ▶ `CONTAINER-I-PDU`
	   ▶ `MULTIPLEXED-I-PDU`
	   ▶ `USER-DEFINED-I-PDU`
	   ▶ `DCM-I-PDU`
	   ▶ `GENERAL-PURPOSE-I-PDU`
	
	▶ `SecOCTxPduSecuredArea/SecOCSecuredTxPduLength` is set to `SE-CURE-COMMUNICATION-PROPS/SECURED-AREA-LENGTH`.
	
	▶ `SecOCTxPduSecuredArea/SecOCSecuredTxPduOffset` is set to `SE-CURE-COMMUNICATION-PROPS/SECURED-AREA-OFFSET`.
	
	▶ `SecOCTxAuthServiceConfigRef` is set to reference the `CsmJob` container that has been created for the `SEC-OC-CRYPTO-SERVICE-MAPPING` which the `PDU-TRIGGERING` of the `SECURED-I-PDU` is referencing.
	
	The content of the choice container `SecOCTxSecuredPduLayer` depends on whether the `SECURED-I-PDU` is a *cryptographic* PDU or not. If it is a *cryptographic* PDU, the content of the choice container is set to `SecOCTxSecuredPduCollection`, otherwise to `SecOCTxSecuredPdu`.
	
	`SecOCTxSecuredPduCollection/SecOCTxAuthenticPdu/SecOCTxAuthenticPduRef` is set to reference the configuration container of the *payload* PDU in the `EcuC` module configuration.
	
	`SecOCTxSecuredPduCollection/SecOCTxCryptographicPdu/SecOC-TxCryptographicPduRef` is set to reference the configuration container of the *cryptographic* PDU in the `EcuC` module configuration.
	
	`SecOCTxSecuredPduCollection/SecOCUseMessageLink/SecOCMes-sageLinkLen` is set to `SECURE-COMMUNICATION-PROPS/MESSAGE-LINK-LENGTH`, `SecOCTxSecuredPduCollection/SecOCUseMessageLink/Se-`

Configuration parameters	Mapping description
	`cOCMessageLinkPos` is set to `SECURE-COMMUNICATION-PROPS/MESSAGE-LINK-POSITION`. `SecOCTxSecuredPdu/SecOCTxSecuredLayerPduRef` is set to reference the configuration container of the `SECURED-I-PDU` in the `EcuC` module configuration. `SecOCTxSecuredPdu/SecOCAuthPduHeaderLength` and `SecOCTxSecuredPduCollection/SecOCTxAuthenticPdu/SecOCAuthPduHeaderLength` are configured depending on the value of `SECURED-I-PDU/USE-SECURED-PDU-HEADER`: ▶ 0 for `NO-HEADER` ▶ 1 for `SECURED-PDU-HEADER-08-BIT` ▶ 2 for `SECURED-PDU-HEADER-16-BIT` ▶ 4 for `SECURED-PDU-HEADER-32-BIT`

`SecOC/SecOCRx-PduProcessing`	For every `SECURED-I-PDU` that the imported `ECU-INSTANCE` receives, the following parameters are configured in `SecOCRxPduProcessing`:
	
	▶ `SecOCAuthenticationVerifyAttempts` is set to `SECURE-COMMUNICATION-PROPS/AUTHENTICATION-RETRIES`.
	
	▶ `SecOCAuthDataFreshnessLen` is set to `SECURE-COMMUNICATION-PROPS/AUTH-DATA-FRESHNESS-LENGTH`.
	
	▶ `SecOCAuthDataFreshnessStartPosition` is set to `SECURE-COMMUNICATION-PROPS/AUTH-DATA-FRESHNESS-START-POSITION`.
	
	▶ `SecOCRxAuthenticPduLayer/SecOCRxAuthenticLayerPduRef` is set to reference the `EcuC` configuration container of the PDU that the `SECURED-I-PDU` references via `PAYLOAD-REF`. That container represents the PDU that is provided to the upper layer module. The following PDU types are supported:
	
	  ▶ `I-SIGNAL-I-PDU`
	
	  ▶ `CONTAINER-I-PDU`
	
	  ▶ `MULTIPLEXED-I-PDU`
	
	  ▶ `USER-DEFINED-I-PDU`
	
	  ▶ `DCM-I-PDU`
	
	  ▶ `GENERAL-PURPOSE-I-PDU`

Configuration parameters	Mapping description
	▶ `SecOCRxPduSecuredArea/SecOCSecuredRxPduLength` is set to `SE-CURE-COMMUNICATION-PROPS/SECURED-AREA-LENGTH`. ▶ `SecOCRxPduSecuredArea/SecOCSecuredRxPduOffset` is set to `SE-CURE-COMMUNICATION-PROPS/SECURED-AREA-OFFSET`. ▶ `SecOCRxAuthServiceConfigRef` is set to reference the `CsmJob` container that has been created for the `SEC-OC-CRYPTO-SERVICE-MAPPING` which the `PDU-TRIGGERING` of the `SECURED-I-PDU` is referencing. The content of the choice container `SecOCRxSecuredPduLayer` depends on whether the `SECURED-I-PDU` is a *cryptographic* PDU or not. If it is a *cryptographic* PDU, the content of the choice container is set to `SecOCRxSecuredPduCollection`, otherwise to `SecOCRxSecuredPdu`. `SecOCRxSecuredPduCollection/SecOCRxAuthenticPdu/SecOCRxAuthenticPduRef` is set to reference the configuration container of the *payload* PDU in the `EcuC` module configuration. `SecOCRxSecuredPduCollection/SecOCRxCryptographicPdu/SecOCRxCryptographicPduRef` is set to reference the configuration container of the *cryptographic* PDU in the `EcuC` module configuration. `SecOCRxSecuredPduCollection/SecOCUseMessageLink/SecOCMessageLinkLen` is set to `SECURE-COMMUNICATION-PROPS/MESSAGE-LINK-LENGTH`, `SecOCRxSecuredPduCollection/SecOCUseMessageLink/SecOCMessageLinkPos` is set to `SECURE-COMMUNICATION-PROPS/MESSAGE-LINK-POSITION`. `SecOCRxSecuredPduCollection/SecOCSecuredRxPduVerification` is set to `RX-SECURITY-VERIFICATION` of the `I-PDU-PORT` via which the `SECURED-I-PDU` is received. `SecOCRxSecuredPdu/SecOCRxSecuredLayerPduRef` is set to reference the configuration container of the `SECURED-I-PDU` in the `EcuC` module configuration. `SecOCRxSecuredPdu/SecOCSecuredRxPduVerification` is set to `RX-SECURITY-VERIFICATION` of the `I-PDU-PORT` via which the `SECURED-I-PDU` is received. `SecOCRxSecuredPdu/SecOCAuthPduHeaderLength` and `SecOCRxSecuredPduCollection/SecOCRxAuthenticPdu/SecOCAuthPduHeaderLength` are configured in the same way as `SecOCTxSecuredPdu/SecOCAu-`

Configuration parame-ters	Mapping description
	`thPduHeaderLength` and `SecOCTxSecuredPduCollection/SecOCTxAu-thenticPdu/SecOCAuthPduHeaderLength`.

3.4.43. SoAd

In its R19/11 release, AUTOSAR introduced a new communication and service model for Ethernet that super-sedes the model that was present in AUTOSAR 4.4 an earlier releases. The AUTOSAR R19-11 schema allows to provide Ethernet communication and service data using either model. Both models are supported for the configuration of the `SoAd` module. It is, however, strongly discouraged to mix AUTOSAR 4.4 and R19-11 communication and service model concepts in a single Ecu Extract or System Description.

This chapter describes the configuration for both AUTOSAR 4.4 and R19-11 model, [5] and [6] provide in their "Ethernet specific description" chapters details on the differences of both models.

A `SOCKET-ADDRESS` conveys a parameter tuple (IP Address/Port Number/protocol). The protocol may either be `Tcp` or `Udp`. The IP Address and Port Number fields can either be defined statically, i.e. they provide a valid IPV4 or IPV6 address respectively a valid port number, or they can be defined dynamically. Section 3.4.43.3, "Parameters derived from the local SOCKET-ADDRESS" describes how the parameters are extracted out of a `SOCKET-ADDRESS` element.

Two `SOCKET-ADDRESS` elements are considered *distinct* if at least one of both has both IP Address and Port Number defined dynamically, or if both `SOCKET-ADDRESS` elements provide distinct values for the parameter fields they define statically.

In addition to that, two *remote* `SOCKET-ADDRESS` elements are also considered *distinct* if at least one of both has either IP Address or Port Number defined dynamically.

3.4.43.1. SoAdSocketConnectionGroup

In the AUTOSAR R19-11 release, a `SoAdSocketConnectionGroup` container and one or more `SoAdSock-etConnection` child containers are created in the following cases:

▸ One or more `STATIC-SOCKET-CONNECTION`s exists which are contained in an `APPLICATION-END-POINT` which in turn is contained in a `SOCKET-ADDRESS` that refers to one of the `ETHERNET-COMMUNI-CATION-CONNECTOR` elements of the configured Ecu Instance. The name of the container is `CG<Name>`, where `<Name>` is the `SHORT-NAME` of the first `STATIC-SOCKET-CONNECTION`. The `SOCKET-ADDRESS` is used to configure the local IP address and port number as described in Section 3.4.43.3, "Parameters derived from the local SOCKET-ADDRESS". The role of the `SoAdSocketConnectionGroup` is *Server* if `STATIC-SOCKET-CONNECTION/TCP-ROLE` is `LISTEN`, otherwise it is *Client*.

One `SoAdSocketConnection` child container is created for each `STATIC-SOCKET-CONNECTION` that refers to a distinct `SOCKET-ADDRESS` via `STATIC-SOCKET-CONNECTION/REMOTE-ADDRESSS`. Its name is the `SHORT-NAME` of the `STATIC-SOCKET-CONNECTION`. The `SoAdSocketRemoteIpAddress` child container is configured as (partly) static, see [Section 3.4.43.2, "Parameters derived from the remote SOCKET-ADDRESS"](#).

▶ The `APPLICATION-ENDPOINT` of a `SOCKET-ADDRESS` that refers to one of the `ETHERNET-COMMUNICATION-CONNECTOR` elements of the configured Ecu Instance is referenced by one or more `PROVIDED-SERVICE-INSTANCE` and/or `CONSUMED-SERVICE-INSTANCE` elements via `PROVIDED-SERVICE-INSTANCE/LOCAL-UNICAST-ADDRESSS` or `CONSUMED-SERVICE-INSTANCE/LOCAL-UNICAST-ADDRESSS`. The name of the container is `CG<Name>_<suffix>`, where `<Name>` is the `SHORT-NAME` of the `SOCKET-ADDRESS`, and `<suffix>` is either a numerical index, or the `SHORT-NAME` of a remote `SOCKET-ADDRESS`. The referenced `SOCKET-ADDRESS` is used to configure the local IP address and port number as described in [Section 3.4.43.3, "Parameters derived from the local SOCKET-ADDRESS"](#). The role of the `SOAD-SOCKET-CONNECTION-GROUP` is *Server* if there is at least one `PROVIDED-SERVICE-INSTANCE` that references the `APPLICATION-ENDPOINT` of the `SOCKET-ADDRESS`, otherwise it is *Client*.

`APPLICATION-ENDPOINT/MAX-NUMBER-OF-CONNECTIONS` child `SoAdSocketConnection` containers are created if the parameter is defined. The container name is `<Name>_<suffix>`, where `<Name>` is the `SHORT-NAME` of the `SOCKET-ADDRESS`, and `<suffix>` is a zero-based index. The `SoAdSocketRemoteIpAddress` is configured as dynamic, see [Section 3.4.43.2, "Parameters derived from the remote SOCKET-ADDRESS"](#).

Additional `SoAdSocketConnection` containers are created for each distinct `SOCKET-ADDRESS` which any of the `PROVIDED-SERVICE-INSTANCE` and/or `CONSUMED-SERVICE-INSTANCE` references via `PROVIDED-SERVICE-INSTANCE/REMOTE-UNICAST-ADDRESSS` or `CONSUMED-SERVICE-INSTANCE/REMOTE-UNICAST-ADDRESSS` which uses the same protocol (i.e. Tcp or Udp) as the local `SOCKET-ADDRESS`. The container name is `<Name>_<suffix>`, where `<suffix>` is the `SHORT-NAME` of the remote `SOCKET-ADDRESS`. The `SoAdSocketRemoteIpAddress` is configured as (partly) static, see [Section 3.4.43.2, "Parameters derived from the remote SOCKET-ADDRESS"](#).

▶ The `APPLICATION-ENDPOINT` of a `SOCKET-ADDRESS` that refers to one of the `ETHERNET-COMMUNICATION-CONNECTOR` elements of the configured Ecu Instance is referenced by one or more `CONSUMED-EVENT-GROUP` elements via `CONSUMED-EVENT-GROUP/EVENT-MULTICAST-ADDRESSS`. The name of the container is `CG<Name>_<suffix>` where `<Name>` is the `SHORT-NAME` of the `SOCKET-ADDRESS`, and `<suffix>` is either a numerical index, or the `SHORT-NAME` of a remote `SOCKET-ADDRESS`. The `SOCKET-ADDRESS` is used to configure the local IP address and port number as described in [Section 3.4.43.3, "Parameters derived from the local SOCKET-ADDRESS"](#). Since the local address is supposed to be a multicast address which is not supported for Tcp, the role of the `SoAdSocketConnectionGroup` is not relevant in this case.

One `SoAdSocketConnection` container is created if the `CONSUMED-SERVICE-INSTANCE` of the `CONSUMED-EVENT-GROUP` refers to a Udp address via `CONSUMED-SERVICE-INSTANCE/REMOTE-UNICAST-ADDRESSS`. The name of the container is the `SHORT-NAME` of that remote `SOCKET-ADDRESS`. The

`SoAdSocketRemoteIpAddress` is configured as (partly) static, see Section 3.4.43.2, "Parameters derived from the remote SOCKET-ADDRESS". If `APPLICATION-ENDPOINT/MAX-NUMBER-OF-CONNEC-TIONS` is defined, then `MAX-NUMBER-OF-CONNECTIONS` child `SoAdSocketConnection` containers are created, named `<Name>_<suffix>`, where `<Name>` is the `SHORT-NAME` of the `SOCKET-ADDRESS`, and `<suffix>` is a zero-based index. The `SoAdSocketRemoteIpAddress` is configured as dynamic, see Section 3.4.43.2, "Parameters derived from the remote SOCKET-ADDRESS".

If `APPLICATION-ENDPOINT/MAX-NUMBER-OF-CONNECTIONS` is not defined or zero, and the parent `CONSUMED-SERVICE-INSTANCE/REMOTE-UNICAST-ADDRESSS` does not refer to a remote Udp address, one `SoAdSocketConnection` container is created, named `<Name>_0` in order to allow reception of PDUs. The `SoAdSocketRemoteIpAddress` is configured as dynamic, see Section 3.4.43.2, "Parameters derived from the remote SOCKET-ADDRESS".

In the AUTOSAR 4.4 and earlier releases, a `SoAdSocketConnectionGroup` and one or more `SoAdSocketConnection` child containers are created for the following cases:

▶ One or more `SOCKET-CONNECTION-BUNDLE` elements refer via `SOCKET-CONNECTION-BUNDLE/SERVER-PORT-REF` to a `SOCKET-ADDRESS` which in turn refers to one of the `ETHERNET-COMMUNICATION-CONNECTOR` elements of the configured Ecu Instance. In this case, the configured Ecu Instance acts as a server. The container name is `CG<Name>`, where `<Name>` is the `SHORT-NAME` of the `SOCKET-CONNECTION-BUNDLE`. The `SOCKET-ADDRESS` is used to configure the local IP address and port number as described in Section 3.4.43.3, "Parameters derived from the local SOCKET-ADDRESS".

For each distinct `SOCKET-ADDRESS` that is referenced by any `SOCKET-CONNECTION` of the `SOCKET-CONNECTION-BUNDLE`, one `SoAdSocketConnection` container is created. Its name is the `SHORT-LABEL` of the `SOCKET-CONNECTION`. If `SOCKET-CONNECTION/CLIENT-IP-ADDR-FROM-CONNECTION-REQUEST` is `true`, `SoAdSocketRemoteIpAddress/SoAdSocketRemoteIpAddress` is configured as `ANY`, otherwise the `SOCKET-ADDRESS` is taken to configure the remote IP address statically, see Section 3.4.43.2, "Parameters derived from the remote SOCKET-ADDRESS". If `SOCKET-CONNECTION/CLIENT-PORT-FROM-CONNECTION-REQUEST` is `true`, `SoAdSocketRemoteIpAddress/SoAdSocketRemotePort` is configured as `0`, otherwise the `SOCKET-ADDRESS` is taken to configure the remote port statically, see Section 3.4.43.2, "Parameters derived from the remote SOCKET-ADDRESS".

▶ One or more `SOCKET-CONNECTION` elements refer via `SOCKET-CONNECTION/CLIENT-PORT-REF` to a `SOCKET-ADDRESS` which in turn refers to one of the `ETHERNET-COMMUNICATION-CONNECTOR` elements of the configured Ecu Instance. In this case, the configured Ecu Instance acts as a client. The container name is `CG<Name>`, where `<Name>` is the `SHORT-LABEL` of one of the `SOCKET-CONNECTION` elements. The `SOCKET-ADDRESS` is used to configure the local IP address and port number as described in Section 3.4.43.3, "Parameters derived from the local SOCKET-ADDRESS".

For each distinct `SOCKET-ADDRESS` that is referenced by any `SOCKET-CONNECTION-BUNDLE` of the `SOCKET-CONNECTION`, one `SoAdSocketConnection` container is created. Its name is the `SHORT-NAME` of the `SOCKET-CONNECTION-BUNDLE`. If `SOCKET-CONNECTION/RUNTIME-IP-ADDRESS-CONFIGURATION` is `SD`, `SoAdSocketRemoteIpAddress/SoAdSocketRemoteIpAddress` is configured

as `ANY`, otherwise the `SOCKET-ADDRESS` is taken to configure the remote IP address statically, see [Section 3.4.43.2, "Parameters derived from the remote SOCKET-ADDRESS"](#). If `SOCKET-CONNECTION/RUN-TIME-PORT-CONFIGURATION` is `SD`, `SoAdSocketRemoteIpAddress/SoAdSocketRemotePort` is configured as `0`, otherwise the `SOCKET-ADDRESS` is taken to configure the remote port statically, see [Section 3.4.43.2, "Parameters derived from the remote SOCKET-ADDRESS"](#).

If any `SoAdPduRoute` or `SoAdSocketRoute` associated with the `SoAdSocketConnectionGroup` container or one of its `SoAdSocketConnection` child containers has `SoAdTxPduHeaderId` or `SoAdRxPduHeaderId` `PDU-ACTIVATION-ROUTING-GROUP` configured, then `SoAdPduHeaderEnable` is set to `true`, otherwise to `false`.

3.4.43.2. Parameters derived from the remote SOCKET-ADDRESS

The `SoAdSocketRemoteIpAddress` sub container of a `SoAdSocketConnection` can be configured as dynamic, as static, or as partly static, i.e. either IP address or port number is configured as dynamic whereas the other parameter is configured as static.

If the `SoAdSocketRemoteIpAddress` is configured as dynamic, `SoAdSocketRemoteIpAddress/SoAdSocketRemoteIpAddress` is configured as `ANY` and `SoAdSocketRemoteIpAddress/SoAdSocketRemotePort` is configured as `0`. At runtime, the socket can be reached via any IP Address/Port number combination provided the protocol, i.e. Tcp/Udp is matching the protocol configured in the parent `SoAdSocketConnectionGroup`.

If the `SoAdSocketRemoteIpAddress` is configured as static or partly static, either the remote IP Address, or the remote Port, or both are determined at configuration time by the properties of the remote `SOCKET-ADDRESS`. If it contains an `APPLICATION-ENDPOINT` which in turn contains a `TCP-TP` or `UDP-TP` element providing a valid `PORT-NUMBER` value and `DYNAMICALLY-ASSIGNED` is either not defined or set to `false`, then `PORT-NUMBER` is used to configure `SoAdSocketRemoteIpAddress/SoAdSocketRemotePort`. If the `APPLICATION-ENDPOINT` of the remote `SOCKET-ADDRESS` refers to a `NETWORK-ENDPOINT` which either contains an `IPV-4-CONFIGURATION` or an `IPV-6-CONFIGURATION`, which contains a valid IP Address in `IPV-4-CONFIGURATION/IPV-4-ADDRESS` or in `IPV-6-CONFIGURATION/IPV-6-ADDRESS`, then this IP address is used to configure `SoAdSocketRemoteIpAddress/SoAdSocketRemoteIpAddress`.

3.4.43.3. Parameters derived from the local SOCKET-ADDRESS

`SoAdSocketLocalAddressRef` is set up to refer to the `TcpIpLocalAddr` container in the `TcpIp` module that has been created for the local `SOCKET-ADDRESS`.

If the local `SOCKET-ADDRESS` contains an `APPLICATION-ENDPOINT` which in turn contains a `TCP-TP` or `UDP-TP` element providing a valid `PORT-NUMBER` value and `DYNAMICALLY-ASSIGNED` is either not defined or set to `false`, then `PORT-NUMBER` is used to configure `SoAdSocketLocalPort`.

The `SoAdSocketProtocol` sub container is configured depending to the child element of the `TP-CONFIGU-RATION` of the `APPLICATION-ENDPOINT` in the local `SOCKET-ADDRESS`. `SoAdSocketProtocol` is configured as `SoAdSocketUdp` if the child is a `UDP-TP` element and configured as `SoAdSocketTcp` if the child is a `TCP-TP` element.

If the local `SOCKET-ADDRESS` has one or more `PROVIDED-SERVICE-INSTANCE` associated, i.e. the `APPLICATION-ENDPOINT` of the `SOCKET-ADDRESS` contains the `PROVIDED-SERVICE-INSTANCE` elements, or the `PROVIDED-SERVICE-INSTANCE` elements reference that `APPLICATION-ENDPOINT` via `PROVIDED-SERVICE-INSTANCE/LOCAL-UNICAST-ADDRESSS`, and the `PROVIDED-SERVICE-INSTANCE` elements contain a valid and consistent `PRIORITY` value, then that value is used to configure `SoAdSocket-FramePriority`.

If the local `SOCKET-ADDRESS` has one or more `CONSUMED-EVENT-GROUP` associated, i.e. the `APPLI-CATION-ENDPOINT` of the `SOCKET-ADDRESS` contains `CONSUMED-SERVICE-INSTANCE` elements which in turn contain the `CONSUMED-EVENT-GROUP` elements, or the `CONSUMED-EVENT-GROUP`s reference that `AP-PLICATION-ENDPOINT` via `CONSUMED-EVENT-GROUP/EVENT-MULTICAST-ADDRESS`, and the `CONSUMED-EVENT-GROUP` elements contain a valid and consistent `PRIORITY` value, then that value is used to configure `SoAdSocketFramePriority`.

If `SoAdSocketFramePriority` cannot be derived from `PROVIDED-SERVICE-INSTANCE` or `CONSUMED-EVENT-GROUP` elements, the `APPLICATION-ENDPOINT/PRIORITY` is used to configure `SoAdSock-etFramePriority`. If `APPLICATION-ENDPOINT/PRIORITY` is also not present, `NETWORK-END-POINT/PRIORITY` of the `NETWORK-ENDPOINT` referenced via `APPLICATION-ENDPOINT/NETWORK-END-POINT-REF` is used to configure `SoAdSocketFramePriority`.

If the local `SOCKET-ADDRESS` contains an `APPLICATION-ENDPOINT` which in turn contains a `TCP-TP` element, the following parameters are configured in `SoAdSocketTcp`:

▶ `SoAdSocketTcpNoDelay` from the negated value of `NAGLES-ALGORITHM`

▶ `SoAdSocketTcpKeepAliveProbesMax` from `KEEP-ALIVE-PROBE-MAX`

▶ `SoAdSocketTcpKeepAliveInterval` from `KEEP-ALIVE-INTERVAL`

▶ `SoAdSocketTcpKeepAliveTime` from `KEEP-ALIVE-TIME`

If any of `SoAdSocketTcp/SoAdSocketTcpKeepAliveProbesMax`, `SoAdSocketTcp/SoAdSocketTcp-KeepAliveInterval`, or `SoAdSocketTcp/SoAdSocketTcpKeepAliveTime` is configured or if the local `SOCKET-ADDRESS` contains an `APPLICATION-ENDPOINT` which in turn contains a `TCP-TP` element providing a `KEEP-ALIVES` field which is set to `true`, then `SoAdSocketTcp/SoAdSocketTcpKeepAlive` is also configured as `true`.

3.4.43.4. Local SOCKET-ADDRESS elements configured by Sd

A *local* `SOCKET-ADDRESS` of the configured `ECU-INSTANCE` is configured by the `Sd` at runtime if the configured `ECU-INSTANCE` sends or receives one or more PDUs via the `SOCKET-ADDRESS` and one of the following conditions applies:

▶ The `SOCKET-ADDRESS` is referenced by an AUTOSAR 4.4 `SOCKET-CONNECTION` via `CLIENT-PORT-REF` and the `SOCKET-CONNECTION` has its `RUNTIME-IP-ADDRESS-CONFIGURATION` set to `SD`.

▶ The `SOCKET-ADDRESS` contains an `APPLICATION-ENDPOINT`, which in turn an AUTOSAR R19-11 `CONSUMED-EVENT-GROUP` of the configured `ECU-INSTANCE` references via `EVENT-MULTICAST-ADDRESSS`.

3.4.43.5. Parameters derived from other entities

If the local `SOCKET-ADDRESS` contains an `APPLICATION-ENDPOINT` which in turn contains a `TCP-TP` element, then `SoAdSocketTcp/SoAdSocketTcpInitiate` is configured depending on the role of the `SoAdSocketConnectionGroup`, see [Section 3.4.43.1, "SoAdSocketConnectionGroup"](#). If the role is *Client*, then `SoAdSocketTcp/SoAdSocketTcpInitiate` is configured as `true`. `SoAdSocketTcp/SoAdSocketTcpInitiate` is configured as `false` if the `SoAdSocketConnectionGroup` container contains only one `SoAdSocketConnection` child container and its role is *Server*.

If the local `SOCKET-ADDRESS` contains an `APPLICATION-ENDPOINT` which in turn contains a `UDP-TP` element, then `SoAdSocketUdp/SoAdSocketUdpTriggerTimeout` and `SoAdSocketUdp/SoAdSocketnPduUdpTxBufferMin` are configured depending on the entity for which the `SoAdSocketConnectionGroup` was created.

If the `SoAdSocketConnectionGroup` was created for a set of `SOCKET-CONNECTION-BUNDLE` elements, or for a set of `SOCKET-CONNECTION` elements, then all `SOCKET-CONNECTION`s respectively all `SOCKET-CONNECTION` elements contained in all `SOCKET-CONNECTION-BUNDLE` elements are examined. If one of the `SOCKET-CONNECTION` elements has a valid `PDU-COLLECTION-TIMEOUT` parameter defined, then this value is taken to configure `SoAdSocketUdp/SoAdSocketUdpTriggerTimeout`. If one of the `SOCKET-CONNECTION` elements has a valid `PDU-COLLECTION-MAX-BUFFER-SIZE` parameter defined, then this value is taken to configure `SoAdSocketUdp/SoAdSocketnPduUdpTxBufferMin`.

If the `SoAdSocketConnectionGroup` was created for a `STATIC-SOCKET-CONNECTION` or for a `SOCKET-ADDRESS` referenced by one or more `PROVIDED-SERVICE-INSTANCE` and/or `CONSUMED-SERVICE-INSTANCE` elements, then `SoAdSocketUdp/SoAdSocketUdpTriggerTimeout` and `SoAdSocketUdp/SoAdSocketnPduUdpTxBufferMin` are configured depending on the local `SOCKET-ADDRESS`: If the local `SOCKET-ADDRESS` contains a valid `PDU-COLLECTION-TIMEOUT` value, then this value is taken to configure `SoAdSocketUdp/SoAdSocketUdpTriggerTimeout`. If the local `SOCKET-ADDRESS` contains a valid `PDU-COLLECTION-MAX-BUFFER-SIZE` value, then this value is taken to configure `SoAdSocketUdp/SoAdSocketnPduUdpTxBufferMin`.

SoAdSocketUdp/SoAdSocketUdpRetryEnabled is configured depending on whether there is any PDU sent/received via Udp that has a length such that (IP_HEADER_LEN + UDP_HEADER_LEN + PDU_LEN) > MTU. In that case, SoAdSocketUdp/SoAdSocketUdpRetryEnabled is configured as true, otherwise it is configured as false. IP_HEADER_LEN in the above formula is 20 for IPV4 and 40 for IPV6, and UDP_-HEADER_LEN is 8 The length of a PDU is calculated as described in Section 3.3.7, "PDU length calculation".

If either the local or the remote SOCKET-ADDRESS refers to a TLS-CRYPTO-SERVICE-MAPPING which in turn contains a TLS-CRYPTO-CIPHER-SUITE for which a TlsConnection container was created in the configuration of the Tls module, then SoAdSocketConnection/SoAdTlsConnectionRef is configured to refer to that TlsConnection container. Section 3.4.47, "Tls" describes the configuration of TlsConnection containers.

3.4.43.6. SoAdPduRoute and SoAdSocketRoute containers

Since AUTOSAR R19-11 SO-CON-I-PDU-IDENTIFIER replaces SOCKET-CONNECTION-IPDU-IDEN-TIFIER, and PDU-ACTIVATION-ROUTING-GROUP replaces SO-AD-ROUTING-GROUP. Both old and new entities convey roughly the same information, however, the relationship between them and the relationship to the entities for which SoAdSocketConnectionGroup and SoAdSocketConnection containers are created are modeled differently.

A SO-CON-I-PDU-IDENTIFIER belongs to a PDU-ACTIVATION-ROUTING-GROUP if the PDU-AC-TIVATION-ROUTING-GROUP references the SO-CON-I-PDU-IDENTIFIER either via I-PDU-IDEN-TIFIER-TCP-REFS or via I-PDU-IDENTIFIER-UDP-REFS. The role via which the SO-CON-I-PDU-IDEN-TIFIER is referenced also defines the protocol used for sending/receiving.

A SOCKET-CONNECTION-IPDU-IDENTIFIER belongs to a SO-AD-ROUTING-GROUP if the SOCKET-CON-NECTION-IPDU-IDENTIFIER references the SO-AD-ROUTING-GROUP.

A SO-CON-I-PDU-IDENTIFIER belongs to a SoAdSocketConnection container if the SoAdSocketCon-nection was created for a STATIC-SOCKET-CONNECTION and the STATIC-SOCKET-CONNECTION refer-ences the SO-CON-I-PDU-IDENTIFIER.

A SO-CON-I-PDU-IDENTIFIER belongs to a SoAdSocketConnectionGroup or one of its SoAdSocket-Connection child containers if the SoAdSocketConnectionGroup was created for a local SOCKET-AD-DRESS element that contains an APPLICATION-ENDPOINT which in turn is referenced by one or more PROVIDED-SERVICE-INSTANCE and/or CONSUMED-SERVICE-INSTANCE elements and one of the following condition holds:

▶ The PROVIDED-SERVICE-INSTANCE/CONSUMED-SERVICE-INSTANCE contains a PDU-ACTI-VATION-ROUTING-GROUP which in turn references the SO-CON-I-PDU-IDENTIFIER.

▶ The PROVIDED-SERVICE-INSTANCE contains an EVENT-HANDLER, which aggregates a PDU-ACTI-VATION-ROUTING-GROUP which in turn references the SO-CON-I-PDU-IDENTIFIER.

▶ The `CONSUMED-SERVICE-INSTANCE` contains a `CONSUMED-EVENT-GROUP`, which aggregates a `PDU-ACTIVATION-ROUTING-GROUP` which in turn references the `SO-CON-I-PDU-IDENTIFIER`.

In all cases the protocol of the `SOCKET-ADDRESS` and the role of the reference of the `PDU-ACTIVATION-ROUTING-GROUP` must match, i.e. if the `SOCKET-ADDRESS` represents a `Udp` address, the `SO-CON-I-PDU-IDENTIFIER` must be referenced via `PDU-ACTIVATION-ROUTING-GROUP/I-PDU-IDENTIFIER-UDP-REFS`, if the `SOCKET-ADDRESS` represents a `Tcp` address, the `SO-CON-I-PDU-IDENTIFIER` must be referenced via `PDU-ACTIVATION-ROUTING-GROUP/I-PDU-IDENTIFIER-TCP-REFS`. If the `PROVIDED-SERVICE-INSTANCE`/`CONSUMED-SERVICE-INSTANCE` references a remote IP address that matches the protocol via `REMOTE-UNICAST-ADDRESSS`, the `SO-CON-I-PDU-IDENTIFIER` belongs to the `SoAdSocketConnection` that was created for that remote address, otherwise the `SO-CON-I-PDU-IDENTIFIER` belongs to the `SoAdSocketConnectionGroup`.

A `SO-CON-I-PDU-IDENTIFIER` belongs to a `SoAdSocketConnectionGroup` or one of its `SoAdSocketConnection` child containers if the `SoAdSocketConnectionGroup` was created for a local `SOCKET-ADDRESS` element that contains an `APPLICATION-ENDPOINT` which in turn is referenced by one or more `CONSUMED-SERVICE-INSTANCE` elements via `CONSUMED-SERVICE-INSTANCE/EVENT-MULTICAST-ADDRESSS` and the following conditions hold:

▶ The `CONSUMED-SERVICE-INSTANCE` contains a `PDU-ACTIVATION-ROUTING-GROUP` which in turn references the `SO-CON-I-PDU-IDENTIFIER` via `PDU-ACTIVATION-ROUTING-GROUP/I-PDU-IDENTIFIER-UDP-REFS`.

▶ The `SOCKET-ADDRESS` represents a Udp multicast address.

If the `CONSUMED-SERVICE-INSTANCE` references a remote `SOCKET-ADDRESS` that represents a Udp address the `SO-CON-I-PDU-IDENTIFIER` belongs to the `SoAdSocketConnection` that was created for that remote address, otherwise the `SO-CON-I-PDU-IDENTIFIER` belongs to the `SoAdSocketConnectionGroup`.

A `SOCKET-CONNECTION-IPDU-IDENTIFIER` belongs to a `SoAdSocketConnectionGroup` container if the `SoAdSocketConnectionGroup` was created for a `SocketConnectionBundle` and the `SocketConnectionBundle` contains the `SOCKET-CONNECTION-IPDU-IDENTIFIER`.

A `SOCKET-CONNECTION-IPDU-IDENTIFIER` belongs to a `SoAdSocketConnectionGroup` container if the `SoAdSocketConnectionGroup` was created for a list of `SOCKET-CONNECTION` elements and all of the `SOCKET-CONNECTION` elements contain a `SOCKET-CONNECTION-IPDU-IDENTIFIER` with the identical properties.

A `SOCKET-CONNECTION-IPDU-IDENTIFIER` belongs to a `SoAdSocketConnection` container if the `SoAdSocketConnection` container was created for a `SOCKET-CONNECTION` and the `SOCKET-CONNECTION` contains the `SOCKET-CONNECTION-IPDU-IDENTIFIER`.

The following sections refer to `SO-CON-I-PDU-IDENTIFIER` and `SOCKET-CONNECTION-IPDU-IDENTIFIER` as `SCII`, and to `PDU-ACTIVATION-ROUTING-GROUP` and `SO-AD-ROUTING-GROUP` as `ROUTING-GROUP` if statements apply to both types at the same time.

An `SCII` is related to a `PDU-TRIGGERING` if it references the `PDU-TRIGGERING`.

A PDU is related to an `SCII` if the `SCII` references a `PDU-TRIGGERING` which in turn references the PDU, and the imported `ECU-INSTANCE` sends, receives, or routes that PDU. For further information about routing see Section 3.3.6, "PDU routing".

3.4.43.7. SoAdPduRoute

All `SCII` elements associated with sent PDUs are grouped according to their sent PDU. One `SoAdPduRoute` sub container is created for each distinct sent PDU. The container name is `PR<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU.

`SoAdTxPduRef` references the corresponding PDU container in the `EcuC` module.

`SoAdTxUpperLayerType` is set to `TP` in the following cases:

▶ An `ETH-TP-CONNECTION` refers to the `PDU-TRIGGERING` that references the transmitted PDU and that belongs to the `PHYSICAL-CHANNEL` on which the PDU is transmitted.

▶ For `GENERAL-PURPOSE-PDU` elements with `CATEGORY` set to `DoIP`.

▶ For `GENERAL-PURPOSE-PDU` elements with `CATEGORY` set to `Dlt_TP`.

▶ For `DCM-I-PDU` elements unless they are referenced by a `PDU-TRIGGERING` which in turn is referenced via `DIAGNOSTIC-CONNECTION/PERIODIC-RESPONSE-UUDT-REF`.

`SoAdTxUpperLayerType` is set to `IF` in the following cases:

▶ For `GENERAL-PURPOSE-PDU` elements with `CATEGORY` set to `Dlt_IF`.

In all other cases, `SoAdTxUpperLayerType` is set to `IF` as a default.

`SoAdTxPduCollectionSemantics` is configured by collecting the `PDU-COLLECTION-SEMANTICS` values of all `SCII` elements that belong to this `SoAdPduRoute`. If that collection yields exactly one distinct value, that value is used to configure `SoAdTxPduCollectionSemantics`:

▶ `SOAD_COLLECT_LAST_IS_BEST` is configured for `LAST-IS-BEST`

▶ `SOAD_COLLECT_QUEUED` is configured for `QUEUED`

3.4.43.8. SoAdPduRoute/SoAdPduRouteDest

One `SoAdPduRouteDest` child container is created for each `SCII` of a sent PDU for which the parent `SoAd-PduRoute` was created unless the type of the received PDU is listed in Section 3.4.43.12, "PDU types without transmission support on SoAdSocketConnectionGroup level". If the `SCII` is associated with a `SoAdSocket-ConnectionGroup` container, see Section 3.4.43.6, "SoAdPduRoute and SoAdSocketRoute containers", only one `SoAdPduRouteDest` container is created. The container name is `<name>_<headerId>`, where `<name>` is the name of the container that represents the related `SoAdSocketConnectionGroup` or `SoAdSocket-`

Connection container and `<headerId>` is the `HEADER-ID` of the `SCII`. If no `HEADER-ID` is available the container name is `<name>`.

`SoAdTxPduHeaderId` is set to `HEADER-ID` of the `SCII` referencing the sent PDU.

`SoAdTxUdpTriggerMode` is set depending on `PDU-COLLECTION-TRIGGER` of the `SCII` referencing the sent PDU: `TRIGGER_ALWAYS` for `ALWAYS` and `TRIGGER_NEVER` for `NEVER`.

`SoAdTxUdpTriggerTimeout` is set to `PDU-COLLECTION-PDU-TIMEOUT` of the `SCII` which references the sent PDU.

`SoAdTxSocketConnOrSocketConnBundleRef` references the `SoAdSocketConnectionGroup` or `SoAdSocketConnection` as described in Section 3.4.43.13, "SocketConnOrSocketConnBundleRef".

`SoAdTxRoutingGroupRef` references all `ROUTING-GROUP` containers created for the `ROUTING-GROUP` elements of the `SCII`.

3.4.43.9. SoAdSocketRoute

All `SCII` elements associated with received PDUs are grouped according to their `HEADER-ID` field and the `SoAdSocketConnection`/`SoAdSocketConnectionGroup` container they are associated with. One container is created for each group unless the type of the received PDU is listed in Section 3.4.43.12, "PDU types without transmission support on SoAdSocketConnectionGroup level". For more information about received or routed PDUs, see Section 3.3.6, "PDU routing" and Section 3.4.46, "TcpIp". The container name is `SR<PREFIX><name>_<headerId>` where `<name>` is the name of the `SoAdSocketConnectionGroup` or `SoAdSocketConnection` of the group, and `<headerId>` is the value of `HEADER-ID`. If no `HEADER-ID` is available the container name is `SR<PREFIX><name>`.

`SoAdRxPduHeaderId` is set to the `HEADER-ID` of the group.

`SoAdRxSocketConnOrSocketConnBundleRef` references the `SoAdSocketConnectionGroup` or `SoAdSocketConnection` as described in Section 3.4.43.13, "SocketConnOrSocketConnBundleRef".

3.4.43.10. SoAdSocketRoute/SoAdSocketRouteDest

One `SoAdSocketRouteDest` child container is created for each `SCII` that yields the `HEADER-ID` value of the parent `SoAdSocketRoute` container and which is associated with the `SoAdSocketConnection` or `SoAdSocketConnectionGroup` the parent `SoAdSocketRoute` container has been created for. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU.

`SoAdRxPduRef` references the corresponding PDU container in the `EcuC` module.

`SoAdRxUpperLayerType` is configured in the same way as `SoAdTxUpperLayerType`.

If the `SoAdSocketLocalAddressRef` parameter of the associated `SoAdSocketConnectionGroup` refers to a multicast address, `SoAdRxRoutingGroupRef` references all `SoAdRoutingGroup` containers that

have been created for `ROUTING-GROUP` elements that have `EVENT-GROUP-CONTROL-TYPE` set to `ACTI-VATION-MULTICAST` and that are associated with the `SCII`.

If the `SoAdSocketLocalAddressRef` parameter of the associated `SoAdSocketConnectionGroup` refers to a unicast address, `SoAdRxRoutingGroupRef` references all `SoAdRoutingGroup` containers that have been created for `ROUTING-GROUP` elements that have `EVENT-GROUP-CONTROL-TYPE` set to a value other than `ACTIVATION-MULTICAST` and that are associated with the `SCII`.

3.4.43.11. SoAdRoutingGroup

For each unique `ROUTING-GROUP` associated with an `SCII` for which a `SoAdPduRoute` and/or a `SoAdSocketRoute` container has been created, a `SoAdRoutingGroup` container is created. The container name is `RG<PREFIX><name>` where `<name>` is the `SHORT-NAME` of the `ROUTING-GROUP`.

`SoAdRoutingGroupTxTriggerable` is set to `true`, if `EVENT-GROUP-CONTROL-TYPE` is set to `ACTI-VATION-AND-TRIGGER-UNICAST` or `TRIGGER-UNICAST`.

3.4.43.12. PDU types without transmission support on SoAdSocketConnection-Group level

For the following PDU types dedicated `SoAdPduRoute` and `SoAdSocketRoute` containers are created for every `SoAdSocketConnection` child container of a `SoAdSocketConnectionGroup` container, even if the `SCII` is associated with the `SoAdSocketConnectionGroup` container:

PDU type	Category	Protocol
GENERAL-PURPOSE-PDU	DoIP	Tcp
GENERAL-PURPOSE-PDU	DoIP	Udp

3.4.43.13. SocketConnOrSocketConnBundleRef

`SoAdPduRoute` and `SoAdSocketRoute` containers reference `SoAdSocketConnectionGroup` or `SoAd-SocketConnection` containers via the parameters `SoAdTxSocketConnOrSocketConnBundleRef` and `SoAdRxSocketConnOrSocketConnBundleRef`.

If a `SoAdPduRoute/SoAdSocketRoute` references a `SoAdSocketConnectionGroup`, the PDUs related to the `SoAdPduRoute/SoAdSocketRoute` can be sent/received via any of the `SoAdSocketConnection` elements of the `SoAdSocketConnectionGroup` at runtime.

Whether or not a `SoAdPduRoute/SoAdSocketRoute` references a `SoAdSocketConnectionGroup` or a `SoAdSocketConnection` depends on the requirements of the `ROUTING-GROUP` elements associated with the PDUs and on the properties of the `SCII` which specifies `HEADER-ID` and `PDU`.

An `SCII` may be associated with one or more `ROUTING-GROUP` elements. They are also referenced by the following entities that are related to service oriented communication:

▶ `PROVIDED-SERVICE-INSTANCE`

▶ `CONSUMED-SERVICE-INSTANCE`

▶ `CONSUMED-EVENT-GROUP`

▶ `EVENT-HANDLER`

If all `ROUTING-GROUP` elements of a `SOCKET-CONNECTION-IPDU-IDENTIFIER` are referenced from entities of one single type, then that type determines whether the `SoAdPduRoute/SoAdSocketRoute` references a `SoAdSocketConnectionGroup` or a `SoAdSocketConnection`. If that type is `PROVIDED-SERVICE-INS-TANCE`, the `SoAdPduRoute/SoAdSocketRoute` references a `SoAdSocketConnection`. For all other types, the `SoAdPduRoute/SoAdSocketRoute` references a `SoAdSocketConnectionGroup`.

If none or more than one type was found, the `SoAdPduRoute/SoAdSocketRoute` references the `SoAdSocketConnection` or `SoAdSocketConnectionGroup` container of the `SCII`, according to Section 3.4.43.6, "SoAdPduRoute and SoAdSocketRoute containers".

3.4.44. SomeIpTp

Configuration parameters	Mapping description
`SomeIpTp/SomeIpTpChannel`	For every `SOMEIP-TP-CONNECTION` element that refers via `TP-SDU-REF` to a PDU which the imported `ECU-INSTANCE` sends or receives, a `SomeIpTpChannel` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU. If `SOMEIP-TP-CONNECTION` refers to a `SOMEIP-TP-CHANNEL` which contains a valid `SEPARATION-TIME`, that value is used to configure `SomeIpTpNPduSeparationTime`. Otherwise, the `SEPARATION-TIME` value of the `SOMEIP-TP-CONNECTION` is used for that purpose. If `SOMEIP-TP-CONNECTION` refers to a `SOMEIP-TP-CHANNEL` which contains a valid `RX-TIMEOUT-TIME`, that value is used to configure `SomeIpTpRxTimeoutTime`.
`SomeIpTp/SomeIpTpChannel/SomeIpTpRxNSdu`	If the `SOMEIP-TP-CONNECTION` refers to a PDU that the imported `ECU-INS-TANCE` receives, one `SomeIpTpRxNSdu` sub container is created. Its name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU.

Configuration parameters	Mapping description
	`SomeIpTpRxSduRef` is set to reference the configuration container of the PDU in the `EcuC` module configuration.
`SomeIpTp/SomeIpT-pChannel/SomeIpT-pRxNSdu/SomeIpT-pRxNPdu`	One `SomeIpTpRxNPdu` is created if the `SOMEIP-TP-CONNECTION` refers via `TRANSPORT-PDU-REF` to a PDU for which the following conditions hold: ▶ The referenced PDU is of type `GENERAL-PURPOSE-I-PDU` ▶ The `CATEGORY` field of the PDU is set to `SOMEIP_SEGMENTED_IPDU` ▶ The imported `ECU-INSTANCE` receives the PDU `SomeIpTpRxNPduRef` is set to reference the configuration container of the PDU in the `EcuC` module configuration.
`SomeIpTp/SomeIpT-pChannel/SomeIpTp-TxNSdu`	If the `SOMEIP-TP-CONNECTION` refers to a PDU that the imported `ECU-INSTANCE` sends, one `SomeIpTpTxNSdu` sub container is created. Its name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the PDU. `SomeIpTpTxNSduRef` is set to reference the configuration container of the PDU in the `EcuC` module configuration.
`SomeIpTp/SomeIpT-pChannel/SomeIpT-pRxNSdu/SomeIpT-pRxNPdu`	One `SomeIpTpRxNPdu` is created if the `SOMEIP-TP-CONNECTION` refers via `TRANSPORT-PDU-REF` to a PDU for which the following conditions hold: ▶ The referenced PDU is of type `GENERAL-PURPOSE-I-PDU` ▶ The `CATEGORY` field of the PDU is set to `SOMEIP_SEGMENTED_IPDU` ▶ The imported `ECU-INSTANCE` sends the PDU `SomeIpTpTxNPduRef` is set to reference the configuration container of the PDU in the `EcuC` module configuration.

3.4.45. StbM

Configuration parameters	Mapping description
`StbMSynchronized-TimeBase`	One `StbMSynchronizedTimeBase` container is created for each `GLOB-AL-TIME-DOMAIN` that fulfills the following conditions: ▶ The imported `ECU-INSTANCE` belongs to the `GLOBAL-TIME-DOMAIN`. ▶ The `GLOBAL-TIME-DOMAIN` does not belong to a parent `GLOBAL-TIME-DOMAIN` to which the imported `ECU-INSTANCE` belongs as well.

Configuration parameters	Mapping description
	The imported `ECU-INSTANCE` belongs to a `GLOBAL-TIME-DOMAIN` in the following cases: ► The `GLOBAL-TIME-MASTER` of the `GLOBAL-TIME-DOMAIN` references a `COMMUNICATION-CONNECTOR` of the imported `ECU-INSTANCE` ► One of the `GLOBAL-TIME-SLAVE` elements of the `GLOBAL-TIME-DOMAIN` references a `COMMUNICATION-CONNECTOR` of the imported `ECU-INSTANCE` ► The `GLOBAL-TIME-DOMAIN` is considered a parent `GLOBAL-TIME-DOMAIN` of a `GLOBAL-TIME-DOMAIN` of the imported `ECU-INSTANCE` A `GLOBAL-TIME-DOMAIN` gtp is considered to be the parent `GLOBAL-TIME-DOMAIN` of a `GLOBAL-TIME-DOMAIN` gts in the following cases: ► gts contains a `GLOBAL-TIME-GATEWAY` which references a `GLOBAL-TIME-MASTER` contained in gts and a `GLOBAL-TIME-SLAVE` contained in gtp. ► gts either contains a `GLOBAL-TIME-MASTER` or `GLOBAL-TIME-SLAVE`, and gtp neither contains a `GLOBAL-TIME-MASTER` nor any `GLOBAL-TIME-SLAVE`, but references gts in `SUB-DOMAIN-REFS`. The name of an imported `StbMSynchronizedTimeBase` container is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `GLOBAL-TIME-DOMAIN`. `StbMSynchronizedTimeBaseIdentifier` is set to `DOMAIN-ID`. `StbMSyncLossTimeout` is set to `SYNC-LOSS-TIMEOUT`. `StbMTimeLeapFutureThreshold` is set to `TIME-LEAP-FUTURE-THRESHOLD`. `StbMTimeLeapPastThreshold` is set to `TIME-LEAP-PAST-THRESHOLD`. `StbMClearTimeleapCount` is set to `TIME-LEAP-HEALING-COUNTER`. For the configuration of `StbMIsSystemWideGlobalTimeMaster`, all `GLOBAL-TIME-MASTER` elements of the `GLOBAL-TIME-DOMAIN` are retrieved. If the `GLOBAL-TIME-DOMAIN` is *abstract*, i.e. it does not reference a `GLOBAL-TIME-MASTER` and no `GLOBAL-TIME-SLAVE`, the `GLOBAL-TIME-MASTER` elements that reference a `COMMUNICATION-CONNECTOR` of the imported `ECU-INS-`

Configuration parameters	Mapping description
	TANCE of all GLOBAL-TIME-DOMAIN elements that are referenced via SUB-DO-MAIN-REF are collected.
	If the GLOBAL-TIME-DOMAIN is not *abstract*, the GLOBAL-TIME-MASTER in GLOBAL-TIME-DOMAIN/MASTER is collected if it references a COMMUNI-CATION-CONNECTOR of the imported ECU-INSTANCE.
	If no GLOBAL-TIME-MASTER was retrieved in the previous step, false is configured. If all retrieved GLOBAL-TIME-MASTER elements yield the same IS-SYSTEM-WIDE-GLOBAL-TIME-MASTER value, that value is configured. Otherwise, nothing is configured.
	If OFFSET-TIME-DOMAIN-REF refers to a valid GLOBAL-TIME-DOMAIN, StbMOffsetTimeBase is configured to reference the StbMSynchronizedTimeBase container that has been created for that GLOBAL-TIME-DOMAIN.
StbMSynchronized-TimeBase/StbM-TimeValidation	If the imported ECU-INSTANCE is connected to a GLOBAL-TIME-DOMAIN as a GLOBAL-TIME-SLAVE, and GLOBAL-TIME-SLAVE/TIME-LEAP-HEALING-COUNTER contains a valid value, then one StbMTimeValidation sub container is created. Its name is <globalTimeDomain>TimeValidation, where <globalTimeDomain> is the SHORT-NAME of the GlobalTimeDomain. StbMTimeValidationRecordTableBlockCount is configured using the value of GLOBAL-TIME-SLAVE/TIME-LEAP-HEALING-COUNTER.
StbMSynchronized-TimeBase/StbM-TimeCorrection	If the GLOBAL-TIME-DOMAIN contains a GLOBAL-TIME-CORRECTION-PROPS, then a StbMTimeCorrection container is created, and following parameters are set: StbMOffsetCorrectionAdaptionInterval is set to OFFSET-CORREC-TION-ADAPTION-INTERVAL. StbMOffsetCorrectionJumpThreshold is set to OFFSET-CORREC-TION-JUMP-THRESHOLD. StbMRateCorrectionMeasurementDuration is set to RATE-CORREC-TION-MEASUREMENT-DURATION. StbMRateCorrectionsPerMeasurementDuration is set to RATE-COR-RECTIONS-PER-MEASUREMENT-DURATION.

3.4.46. TcpIp

Configuration parameters	Mapping description
`TcpIpConfig/TcpIpCtrl`	For every *virtual* `ETHERNET-COMMUNICATION-CONTROLLER` of the imported `ECU-INSTANCE` (see [Section 3.4.17, "EthIf"](#)), a `TcpIpCtrl` container is created. The container name is `<PREFIX><name_cc>_<name_chn>`, where `<name_cc>` is the `SHORT-NAME` of the `ETHERNET-COMMUNICATION-CONTROLLER` and `<name_chn>` is the `SHORT-NAME` of the `ETHERNET-PHYSICAL-CHANNEL`. `TcpIpEthIfCtrlRef` references the container created for this *virtual* `ETHERNET-COMMUNICATION-CONTROLLER` in the `EthIf` module. `TcpIpIpFramePrioDefault` is set to the value of `DEFAULT-PRIORITY` of the `VLAN-MEMBERSHIP` belonging to the `ETHERNET-PHYSICAL-CHANNEL` of the `ETHERNET-COMMUNICATION-CONTROLLER`. A `VLAN-MEMBERSHIP` belongs to an `ETHERNET-PHYSICAL-CHANNEL` if it references this `ETHERNET-PHYSICAL-CHANNEL` and is aggregated by a `COUPLING-PORT` of the `ETHERNET-COMMUNICATION-CONTROLLER`. If `DEFAULT-PRIORITY` is not available, `TcpIpIpFramePrioDefault` is set to `0` For every `ETHERNET-COMMUNICATION-CONNECTOR` associated with a `NETWORK-ENDPOINT` with valid `IPV-4-CONFIGURATION` or `IPV-6-CONFIGURATION`, one `TcpIpIpVXCtrl` container is created. Depending on the `TcpIpDomainType`, a `TcpIpIpV4Ctrl` or `TcpIpIpV6Ctrl` subcontainer is created. An `ETHERNET-COMMUNICATION-CONNECTOR` is associated with a `NETWORK-ENDPOINT` if there is a `SOCKET-ADDRESS` which references the `NETWORK-ENDPOINT` via `SOCKET-ADDRESS/NETWORK-ENDPOINT-REF` and which contains an `APPLICATION-ENDPOINT` that in turn references an `ETHERNET-COMMUNICATION-CONNECTOR` either via `APPLICATION-ENDPOINT/CONNECTOR-REF` or via `APPLICATION-ENDPOINT/MULTICAST-CONNECTOR-REFS/MULTICAST-CONNECTOR-REF`.
`TcpIpConfig/TcpIpCtrl/TcpIpIpVXCtrl/TcpIpIpV4Ctrl`	▶ `TcpIpIpV4PathMtuEnabled` is set to `PATH-MTU-ENABLED` ▶ `TcpIpIpV4PathMtuTimeout` is set to `PATH-MTU-TIMEOUT`
`TcpIpConfig/TcpIpCtrl/TcpIpIpVXCtrl/TcpIpIpV6Ctrl`	▶ `TcpIpIpV6PathMtuEnabled` is set to `IP-V-6-PATH-MTU-ENABLED` ▶ `TcpIpIpV6PathMtuTimeout` is set to `IP-V-6-PATH-MTU-TIMEOUT`
`TcpIpConfig/TcpIpIpCon-`	If at least one PDU that is sent or received via Udp needs fragmentation, i.e. its payload length plus the length of the Udp header plus the length of the IP head-

Configuration parameters	Mapping description
`fig/TcpIpIpFragmentationConfig`	er exceeds the MTU size defined in `ETHERNET-COMMUNICATION-CONNEC-TOR/MAXIMUM-TRANSMISSION-UNIT`, a container `TcpIpIpConfig` and a subcontainer `TcpIpIpFragmentationConfig` are created in `TcpIpConfig`. If any of the PDUs exceeding the MTU size is received, `TcpIpIpFragmentationRxEnabled` is set to `true`. If any of the PDUs exceeding the MTU size is sent, `TcpIpIpFragmentationTxEnabled` is set to `OUTOFORDER`. The payload length of a PDU is calculated as described in [Section 3.3.7, "PDU length calculation"](#).
`TcpIpConfig/TcpIpIpConfig/TcpIpIpV4Config`	If the imported `ECU-INSTANCE` contains one or more `ETHERNET-COMMUNICATION-CONNECTOR` elements which reference an `ETH-IP-PROPS` element, and all referenced `ETH-IP-PROPS` elements contain consistent parameter values, one `TcpIpIpV4Config` sub container is created if the referenced `ETH-IP-PROPS` elements contain an `IPV-4-PROPS` element. The following tables display how the parameters in the `TcpIpIpV4Config` container are configured using the parameter values of the `IPV-4-PROPS` elements.

TcpIpIpV4Config Parameter	IPV-4-PROPS Parameter
`TcpIpArpConfig/TcpIpArpNumGratuitousARPonStartup`	`ARP-PROPS/TCP-IP-ARP-NUM-GRATUITOUS-ARP-ON-STARTUP`
`TcpIpArpConfig/TcpIpArpPacketQueueEnabled`	`ARP-PROPS/TCP-IP-ARP-PACKET-QUEUE-ENABLED`
`TcpIpArpConfig/TcpIpArpRequestTimeout`	`ARP-PROPS/TCP-IP-ARP-REQUEST-TIMEOUT`
`TcpIpArpConfig/TcpIpArpTableEntryTimeout`	`ARP-PROPS/TCP-IP-ARP-TABLE-ENTRY-TIMEOUT`
`TcpIpAutoIpConfig/TcpIpAutoIpInitTimeout`	`AUTO-IP-PROPS/CP-IP-AUTO-IP-INIT-TIMEOUT`
`TcpIpIpFragmentationConfig/TcpIpIpFragmentationRxEnabled`	`FRAGMENTATION-PROPS/TCP-IP-IP-FRAGMENTATION-RX-ENABLED`
`TcpIpIpFragmentationConfig/TcpIpIpNumFragments`	`FRAGMENTATION-PROPS/TCP-IP-IP-NUM-FRAGMENTS`
`TcpIpIpFragmentationConfig/TcpIpIpNumReassDgrams`	`FRAGMENTATION-PROPS/TCP-IP-IP-NUM-REASS-DGRAMS`
`TcpIpIpFragmentationConfig/TcpIpIpReassTimeout`	`FRAGMENTATION-PROPS/TCP-IP-IP-REASS-TIMEOUT`

Configuration parameters	Mapping description
	If the imported `ECU-INSTANCE` references an `ETH-TCP-IP-ICMP-PROPS` element, one `TcpIpIcmpConfig` sub container is created in `TcpIpIpV4Config` if the referenced `ETH-TCP-IP-ICMP-PROPS` element contains an `ICMP-V-4-PROPS` element, the following table displays how the parameters in the `TcpIpIcmpConfig` container are configured using the parameter values of the `ICMP-V-4-PROPS` elements.

TcpIpIcmpConfig Parameter	ICMP-V-4-PROPS Parameter
`TcpIpIcmpEchoReplyEnabled`	`TCP-IP-ICMP-V-4-ECHO-REPLY-ENABLED`
`TcpIpIcmpTtl`	`TCP-IP-ICMP-V-4-TTL`

Configuration parameters	Mapping description
`TcpIpConfig/TcpIpIpConfig/TcpIpIpV6Config`	If the imported `ECU-INSTANCE` contains one or more `ETHERNET-COMMUNICATION-CONNECTOR` elements which reference an `ETH-IP-PROPS` element, and all referenced `ETH-IP-PROPS` elements contain consistent parameter values, one `TcpIpIpV6Config` sub container is created if the referenced `ETH-IP-PROPS` elements contain an `IP-V6-PROPS` element. The following tables display how the parameters in the `TcpIpIpV6Config` container are configured using the parameter values of the `IP-V6-PROPS` elements.

TcpIpIpV6Config Parameter	IP-V6-PROPS Parameter
`TcpIpIpV6FragmentationConfig/TcpIpIpV6ReassemblyBufferCount`	`FRAGMENTATION-PROPS/TCP-IP-IP-REASSEMBLY-BUFFER-COUNT`
`TcpIpIpV6FragmentationConfig/TcpIpIpV6ReassemblyBufferSize`	`FRAGMENTATION-PROPS/TCP-IP-IP-REASSEMBLY-BUFFER-SIZE`
`TcpIpIpV6FragmentationConfig/TcpIpIpV6ReassemblySegmentCount`	`FRAGMENTATION-PROPS/TCP-IP-IP-REASSEMBLY-SEGMENT-COUNT`
`TcpIpIpV6FragmentationConfig/TcpIpIpV6ReassemblyTimeout`	`FRAGMENTATION-PROPS/TCP-IP-IP-REASSEMBLY-TIMEOUT`
`TcpIpIpV6FragmentationConfig/TcpIpIpV6TxFragmentBufferCount`	`FRAGMENTATION-PROPS/TCP-IP-IP-TX-FRAGMENT-BUFFER-COUNT`
`TcpIpIpV6FragmentationConfig/TcpIpIpV6TxFragmentBufferSize`	`FRAGMENTATION-PROPS/TCP-IP-IP-TX-FRAGMENT-BUFFER-SIZE`

Configuration parameters	Mapping description	
	TcpIpIpV6Config Parameter	**IP-V6-PROPS Parameter**
	`TcpIpDhcpV6Config/ TcpIpDhcpV6CnfDelayMax`	`DHCP-PROPS/TCP-IP-DHCP-V-6- CNF-DELAY-MAX`
	`TcpIpDhcpV6Config/ TcpIpDhcpV6CnfDelayMin`	`DHCP-PROPS/TCP-IP-DHCP-V-6- CNF-DELAY-MIN`
	`TcpIpDhcpV6Config/ TcpIpDhcpV6InfDelayMax`	`DHCP-PROPS/TCP-IP-DHCP-V-6- INF-DELAY-MAX`
	`TcpIpDhcpV6Config/ TcpIpDhcpV6InfDelayMin`	`DHCP-PROPS/TCP-IP-DHCP-V-6- INF-DELAY-MIN`
	`TcpIpDhcpV6Config/ TcpIpDhcpV6SolDelayMax`	`DHCP-PROPS/TCP-IP-DHCP-V-6- SOL-DELAY-MAX`
	`TcpIpDhcpV6Config/ TcpIpDhcpV6SolDelayMin`	`DHCP-PROPS/TCP-IP-DHCP-V-6- SOL-DELAY-MIN`
	`TcpIpNdpConfig/TcpIpNdpAr- NudConfig/TcpIpNdpDefault- ReachableTime`	`NDP-PROPS/TCP-IP-NDP-DE- FAULT-REACHABLE-TIME`
	`TcpIpNdpConfig/TcpIpNdpAr- NudConfig/TcpIpNdpDefaultRe- transTimer`	`NDP-PROPS/TCP-IP-NDP-DE- FAULT-RETRANS-TIMER`
	`TcpIpNdpConfig/TcpIpNdpAr- NudConfig/TcpIpNdpArNudCon- fig/TcpIpNdpMaxRandomFactor`	`NDP-PROPS/TCP-IP-NDP-MAX- RANDOM-FACTOR`
	`TcpIpNdpConfig/TcpIpNdpAr- NudConfig/TcpIpNdpMinRandom- Factor`	`NDP-PROPS/TCP-IP-NDP-MIN- RANDOM-FACTOR`
	`TcpIpNdpConfig/TcpIpNdpAr- NudConfig/TcpIpNdpNeigh- borUnreachabilityDetectio- nEnabled`	`NDP-PROPS/TCP-IP-NDP-NEIGH- BOR-UNREACHABILITY-DETEC- TION-ENABLED`
	`TcpIpNdpConfig/TcpIpNdpAr- NudConfig/TcpIpNdpNumMulti- castSolicitations`	`NDP-PROPS/TCP-IP-NDP-NUM- MULTICAST-SOLICITATIONS`
	`TcpIpNdpConfig/TcpIpNdpAr- NudConfig/TcpIpNdpNumUnicas- tSolicitations`	`NDP-PROPS/TCP-IP-NDP-NUM- UNICAST-SOLICITATIONS`

Configuration parameters	Mapping description	
	TcpIpIpV6Config Parameter	**IP-V6-PROPS Parameter**
	`TcpIpNdpConfig/TcpIpNd-pArNudConfig/TcpIpNdpPack-etQueueEnabled`	`NDP-PROPS/TCP-IP-NDP-PACK-ET-QUEUE-ENABLED`
	`TcpIpNdpConfig/TcpIpNdpAr-NudConfig/TcpIpNdpRandom-ReachableTimeEnabled`	`NDP-PROPS/TCP-IP-NDP-RAN-DOM-REACHABLE-TIME-ENABLED`
	`TcpIpNdpConfig/TcpIpNdpAr-NudConfig/TcpIpNdpDefensive-Processing`	`NDP-PROPS/TCP-IP-NDP-DE-FENSIVE-PROCESSING`
	`TcpIpNdpConfig/TcpIp-NdpPrefixRouterDiscov-eryConfig/TcpIpNdpDefault-RouterListSize`	`NDP-PROPS/TCP-IP-NDP-DE-FAULT-ROUTER-LIST-SIZE`
	`TcpIpNdpConfig/TcpIpNdp-PrefixRouterDiscoveryCon-fig/TcpIpNdpDestination-CacheSize`	`NDP-PROPS/TCP-IP-NDP-DESTI-NATION-CACHE-SIZE`
	`TcpIpNdpConfig/TcpIpNdp-PrefixRouterDiscoveryCon-fig/TcpIpNdpDynamicHopLim-itEnabled`	`NDP-PROPS/TCP-IP-NDP-DY-NAMIC-HOP-LIMIT-ENABLED`
	`TcpIpNdpConfig/TcpIpNdp-PrefixRouterDiscoveryCon-fig/TcpIpNdpDynamicMtuEn-abled`	`NDP-PROPS/TCP-IP-NDP-DY-NAMIC-MTU-ENABLED`
	`TcpIpNdpConfig/TcpIpNdp-PrefixRouterDiscoveryCon-fig/TcpIpNdpDynamicReach-ableTimeEnabled`	`NDP-PROPS/TCP-IP-NDP-DY-NAMIC-REACHABLE-TIME-ENABLED`
	`TcpIpNdpConfig/TcpIpNdp-PrefixRouterDiscoveryCon-fig/TcpIpNdpDynamicRe-transTimeEnabled`	`NDP-PROPS/TCP-IP-NDP-DY-NAMIC-RETRANS-TIME-ENABLED`
	`TcpIpNdpConfig/TcpIpNdp-PrefixRouterDiscoveryCon-`	`NDP-PROPS/TCP-IP-NDP-MAX-RTR-SOLICITATION-DELAY`

Configuration parameters	Mapping description	
	TcpIpIpV6Config Parameter	**IP-V6-PROPS Parameter**
	`fig/TcpIpNdpMaxRtrSolicita-tionDelay`	
	`TcpIpNdpConfig/TcpIpNdp-PrefixRouterDiscoveryCon-fig/TcpIpNdpMaxRtrSolicita-tions`	`NDP-PROPS/TCP-IP-NDP-MAX-RTR-SOLICITATIONS`
	`TcpIpNdpConfig/TcpIpNdp-PrefixRouterDiscoveryCon-fig/TcpIpNdpPrefixListSize`	`NDP-PROPS/TCP-IP-NDP-PRE-FIX-LIST-SIZE`
	`TcpIpNdpConfig/TcpIpNdp-PrefixRouterDiscoveryCon-fig/TcpIpNdpRndRtrSolicita-tionDelayEnabled`	`NDP-PROPS/TCP-IP-NDP-RND-RTR-SOLICITATION-DELAY-EN-ABLED`
	`TcpIpNdpConfig/TcpIpNdp-PrefixRouterDiscoveryCon-fig/TcpIpNdpRtrSolicita-tionInterval`	`NDP-PROPS/TCP-IP-NDP-RTR-SOLICITATION-INTERVAL`
	`TcpIpNdpConfig/TcpIpNd-pSlaacConfig/TcpIpNdpSlaac-DadNumberOfTransmissions`	`NDP-PROPS/TCP-IP-NDP-SLAAC-DAD-NUMBER-OF-TRANSMISSIONS`
	`TcpIpNdpConfig/TcpIpNd-pSlaacConfig/TcpIpNdpSlaac-DadRetransmissionDelay`	`NDP-PROPS/TCP-IP-NDP-SLAAC-DAD-RETRANSMISSION-DELAY`
	`TcpIpNdpConfig/TcpIp-NdpSlaacConfig/TcpIpNd-pSlaacDelayEnabled`	`NDP-PROPS/TCP-IP-NDP-SLAAC-DELAY-ENABLED`
	`TcpIpNdpConfig/TcpIpNd-pSlaacConfig/TcpIpNdpSlaa-cOptimisticDadEnabled`	`NDP-PROPS/TCP-IP-NDP-SLAAC-OPTIMISTIC-DAD-ENABLED`
	If the imported `ECU-INSTANCE` references an `ETH-TCP-IP-ICMP-PROPS` element, one `TcpIpIcmpV6Config` sub container is created in `TcpIpIpV6Config` if the referenced `ETH-TCP-IP-ICMP-PROPS` element contains an `ICMP-V-6-PROPS` element, the following table displays how the parameters in the `TcpIpIpV6Config` container are configured using the parameter values of the `ICMP-V-6-PROPS` elements.	

Configuration parameters	Mapping description	
	TcpIpIpV6Config Parameter	**ICMP-V-6-PROPS Parameter**
	`TcpIpIcmpV6EchoReplyEnabled`	`CP-IP-ICMP-V-6-ECHO-RE-PLY-ENABLED`
	`TcpIpIcmpV6HopLimit`	`TCP-IP-ICMP-V-6-HOP-LIMIT`
	`TcpIpIcmpV6MsgDestinationUn-reachable-Enabled`	`TCP-IP-ICMP-V-6-MSG-DESTI-NATION-UNREACHABLE-ENABLED`
	`TcpIpIcmpV6MsgParameterProb-lemEnabled`	`TCP-IP-ICMP-V-6-MSG-PARAME-TER-PROBLEM-ENABLED`
	`TcpIpIcmpV6EchoReplyAvoid-Fragmentation`	`TCP-IP-ICMP-V-6-ECHO-RE-PLY-AVOID-FRAGMENTATION`

`TcpIpCon-fig/TcpIpTcpConfig`	One `TcpIpTcpConfig` container is created in one of the following cases:
	
	▶ If there is at least one `SOCKET-ADDRESS` element which represents a local IP port of the imported `ECU-INSTANCE` and which contains a `TP-CON-FIGURATION/TCP-TP` element in its `APPLICATION-ENDPOINT` which in turn contains at least one of the parameters
	
	    ▶ `KEEP-ALIVE-INTERVAL`
	
	    ▶ `KEEP-ALIVE-PROBES-MAX`
	
	    ▶ `KEEP-ALIVE-TIME`
	
	or has its `KEEP-ALIVE` parameter set to `true` then the parameter `TcpIpTcpKeepAliveEnabled` is set to `true`.
	
	▶ If the imported `ECU-INSTANCE` references an `ETH-TCP-IP-PROPS` element which in turn contains a `TCP-PROPS` element, the parameters listed below are configured in the `TcpIpTcpConfig` container using the parameter values of `TCP-PROPS`.	

TcpIpTcpConfig Parameter	TCP-PROPS Parameter
`TcpIpDelayedAckTimeout`	`TCP-DELAYED-ACK-TIMEOUT`
`TcpIpTcpCongestionAvoid-anceEnabled`	`TCP-CONGESTION-AVOI-DANCE-ENABLED`
`TcpIpTcpFastRecoveryEnabled`	`TCP-FAST-RECOVERY-ENABLED`
`TcpIpTcpFastRetransmitEn-abled`	`TCP-FAST-RETRANSMIT-ENABLED`
`TcpIpTcpFinWait2Timeout`	`TCP-FIN-WAIT-2-TIMEOUT`
`TcpIpTcpKeepAliveEnabled`	`TCP-KEEP-ALIVE-ENABLED`

Configuration parameters	Mapping description		
	TcpIpTcpConfig Parameter		**TCP-PROPS Parameter**
	`TcpIpTcpKeepAliveInterval`		`TCP-KEEP-ALIVE-INTERVAL`
	`TcpIpTcpKeepAliveProbesMax`		`TCP-KEEP-ALIVE-PROBES-MAX`
	`TcpIpTcpKeepAliveTime`		`TCP-KEEP-ALIVE-TIME`
	`TcpIpTcpMaxRtx`		`TCP-MAX-RTX`
	`TcpIpTcpMsl`		`TCP-MSL`
	`TcpIpTcpNagleEnabled`		`TCP-NAGLE-ENABLED`
	`TcpIpTcpReceiveWindowMax`		`TCP-RECEIVE-WINDOW-MAX`
	`TcpIpTcpRetransmissionTimeout`		`TCP-RETRANSMISSION-TIMEOUT`
	`TcpIpTcpSlowStartEnabled`		`TCP-SLOW-START-ENABLED`
	`TcpIpTcpSynMaxRtx`		`TCP-SYN-MAX-RTX`
	`TcpIpTcpSynReceivedTimeout`		`TCP-SYN-RECEIVED-TIMEOUT`
	`TcpIpTcpTtl`		`TCP-TTL`
`TcpIpConfig/TcpIpUdpConfig`	If the imported `ECU-INSTANCE` references an `ETH-TCP-IP-PROPS` element which in turn contains a `UDP-PROPS` element, one `TcpIpUdpConfig` container is created and `TcpIpUdpConfig/TcpIpUdpTtl` is configured using the value of `UDP-PROPS/UDP-TTL`.		
`TcpIpConfig/TcpIpLocalAddr`	A `NETWORK-ENDPOINT` is considered to be configured at runtime in one of the following situations: ▶ The `IPV-4-CONFIGURATION` of the `NETWORK-ENDPOINT` has its `IPV-4-ADDRESS-SOURCE` set to a value other than `FIXED`. ▶ The `IPV-6-CONFIGURATION` of the `NETWORK-ENDPOINT` has its `IPV-6-ADDRESS-SOURCE` set to a value other than `FIXED`. ▶ The `NETWORK-ENDPOINT` represents a multicast address that is configured at runtime. In all other cases, a `NETWORK-ENDPOINT` is considered to be configured at configuration time. A `NETWORK-ENDPOINT` represents a multicast address that is configured at runtime if either its `IPV-4-ADDRESS` or its `IPV-6-ADDRESS` is a multicast address and a *local* `SOCKET-ADDRESS` which meets the conditions described in Section 3.4.43.4, "Local SOCKET-ADDRESS elements configured by Sd" references it via `APPLICATION-ENDPOINT/NETWORK-ENDPOINT-REF`.		

Configuration parameters	Mapping description
	Every `NETWORK-ENDPOINT` that is associated with an `ETHERNET-COMMUNI-CATION-CONNECTOR` of a *virtual* `ETHERNET-COMMUNICATION-CONTROLLER` is taken as input for the creation of `TcpIpLocalAddr` containers. For further information about `ETHERNET-COMMUNICATION-CONTROLLER`, see [Section 3.4.17, "EthIf"](#). If a `NETWORK-ENDPOINT` is considered to be configured at configuration time, one `TcpIpLocalAddr` is created for it. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `NETWORK-END-POINT`. If a `NETWORK-ENDPOINT` is considered to be configured at runtime, one `TcpIpLocalAddr` container is created for each *local* `SOCKET-ADDRESS` that references it via its contained `APPLICATION-ENDPOINT`. The container name is `<PREFIX><name><suffix>`, where `<name>` is the `SHORT-NAME` of the `NET-WORK-ENDPOINT`. `TcpIpCtrlRef` references the `TcpIpCtrl` container created for the *virtual* `ETHERNET-COMMUNICATION-CONTROLLER`. The following parameters are configured using either the first `IPV-4-CONFIGU-RATION` which has `IPV-4-ADDRESS-SOURCE` set to `FIXED` or the first `IPV-6-CONFIGURATION` which has `IPV-6-ADDRESS-SOURCE` set to `FIXED`: `TcpIpDomainType` is set to `TCPIP_AF_INET` if `NETWORK-ENDPOINT/NET-WORK-ENDPOINT-ADDRESSES` contains at least one `IPV-4-CONFIGURATION` entry and no `IPV-6-CONFIGURATION` entries. It is set to `TCPIP_AF_INET6` if `NETWORK-ENDPOINT/NETWORK-ENDPOINT-ADDRESSES` contains at least one `IPV-6-CONFIGURATION` entry and no `IPV-4-CONFIGURATION` entries. `TcpIpAddressType` is set to `TCPIP_MULTICAST` if `IPV-4-ADDRESS` or `IPV-6-ADDRESS` represents a multicast address, otherwise `TcpIpAd-dressType` is set to `TCPIP_UNICAST`.
`TcpIpCon-fig/TcpIpLocalAd-dr/TcpIpStaticIpAd-dressConfig`	This container is only configured for `NETWORK-ENDPOINT` elements that do not represent multicast addresses that are configured at runtime. `TcpIpStaticIpAddress` is set to `ANY` if the `NETWORK-ENDPOINT` is a multicast address which is configured at runtime, or to `IPV-4-ADDRESS`, respectively `IPV-6-ADDRESS` in all other cases. If `TcpIpAddressType` is set to `TCPIP_MULTICAST`, `TcpIpNetmask` is not configured. Otherwise it is either set to the content of `IPV-4-CONFIGU-`

Configuration parameters	Mapping description
	RATION/NETWORK-MASK converted to the *Classless Inter-Domain Routing* (CIDR) notation, or to IPV-6-CONFIGURATION/IP-ADDRESS-PREFIX-LENGTH which is already provided in CIDR notation. If TcpIpAddressType is set to TCPIP_MULTICAST, TcpIpDefaultRouter is not configured. Otherwise the value of IPV-4-CONFIGURATION/DEFAULT-GATEWAY or IPV-6-CONFIGURATION/DEFAULT-ROUTER is retrieved. If that value does not equal ANY, it is used to configure TcpIpDefaultRouter.
TcpIpConfig/TcpIpLocalAddr/TcpIpAddrAssignment	For every element in the list of IPV-4-CONFIGURATION/IPV-4-ADDRESS-SOURCE entries respectively for every element in the list of IPV-6-CONFIGURATION/IPV-6-ADDRESS-SOURCE entries of the NETWORK-ENDPOINT a TcpIpAddrAssignment container is created. The container name is TcpIpAddrAssignment_<assignmentMethod>, where <assignmentMethod> depends on the value of IPV-4-ADDRESS-SOURCE or IPV-6-ADDRESS-SOURCE: ▶ TCPIP_LINKLOCAL for AUTO-IP or LINK-LOCAL ▶ TCPIP_LINKLOCAL_DOIP for AUTO-IP--DOIP or LINK-LOCAL--DOIP ▶ TCPIP_DHCP for DHCPV-4 or DHCPV-6 ▶ TCPIP_STATIC for FIXED ▶ TCPIP_IPV6_ROUTER for ROUTER-ADVERTISEMENT If IPV-4-ADDRESS-SOURCE is set to AUTO-IPDHCPV-4, two TcpIpAddrAssignment containers are created, one for AUTO-IP and one for DHCPV-4. TcpIpAssignmentMethod is set to <assignmentMethod>. If IPV-4-ADDRESS-SOURCE or IPV-6-ADDRESS-SOURCE contains a valid value other than AUTO-IPDHCPV-4, TcpIpAssignmentPriority is set to ASSIGNMENT-PRIORITY. TcpIpAssignmentTrigger is set to TCPIP_MANUAL if TcpIpStaticIpAddress is set to ANY. For NETWORK-ENDPOINT elements that do not contain any IPV-4-ADDRESS-SOURCE or any IPV-6-ADDRESS-SOURCE but represent a multicast address which is configured at runtime, one default TcpIpAddrAssignment container is created. Its name is TcpIpAddrAssignment_MANUAL. TcpIpAssignmentMethod is set to TCPIP_STATIC and TcpIpAssignmentTrigger is set to TCPIP_MANUAL.

Configuration parameters	Mapping description
`TcpIpConfig/TcpIpIpSecConfig`	If a `NETWORK-ENDPOINT` for which a `TcpIpLocalAddr` container was created contains one or more `IP-SEC-RULE` sub elements, one `TcpIpIpSecConfig` container is created. For each `IP-SEC-RULE` within a `NETWORK-ENDPOINT`, one `TcpIpIpSecSecurityAssociationCfg` container is created. Its name is `SASS_<name>`, where `<name>` is the short name of the `IP-SEC-RULE`. The following parameters are configured: `TcpIpIpSecSecurityAssociationKeyExchangeMethod` is set to `DYNAMIC` if `IP-SEC-RULE/IKE-AUTHENTICATION-METHOD` is available. `TcpIpIpSecSecurityAssociationProtocol` is set to `IP-SEC-RULE/HEADER-TYPE` if the value is either `AH` or `ESP`. For all other values, the parameter is not configured. For each `NETWORK-ENDPOINT` referenced by `IP-SEC-RULE/REMOTE-IP-ADDRESS-REFS/REMOTE-IP-ADDRESS-REF`, one `TcpIpIpSecConnections` container is created. Its name is `SCON_<name>_<remote_ip>`, where `<name>` is the short name of the `IP-SEC-RULE` and `<remote_ip>` is the short name of the remote `NETWORK-ENDPOINT`. If no `NETWORK-ENDPOINT` element is referenced via `IP-SEC-RULE/REMOTE-IP-ADDRESS-REFS/REMOTE-IP-ADDRESS-REF`, one `TcpIpIpSecConnections` container is created. Its name is `SCON_<name>`. In both cases, the following parameters are configured: `TcpIpIpSecDomainType` is set to the same value as the `TcpIpDomainType` parameter of the `TcpIpLocalAddr` container which `TcpIpIpSecLocalAddrRef` refers to. `TcpIpIpSecLocalAddrRef` is configured to refer to the `TcpIpLocalAddr` container that has been created for the `NETWORK-ENDPOINT` of the `IP-SEC-RULE`. If the `TcpIpIpSecConnections` container has been created for a valid remote `NETWORK-ENDPOINT`, `TcpIpIpSecRemoteAddrType` is configured depending on whether it is representing a statically configured multicast address or not. The parameter is set to `TCPIP_MULTICAST` for statically configured multicast addresses, and to `TCPIP_UNICAST` for all other types of addresses. If the `TcpIpIpSecConnections` has not been created for a valid remote `NETWORK-ENDPOINT`, `TcpIpIpSecRemoteAddrType` is not configured.

Configuration parameters	Mapping description
	If the `TcpIpIpSecConnections` has been created for a valid remote `NETWORK-ENDPOINT` which represents a statically configured IP address, `IpSecRemoteAddrConfig/SingleAddress/TcpIpIpSecRemoteAddr` is configured to contain that static IP address. If `IP-SEC-RULE/POLICY` is set to `IPSEC`, `TcpIpIpIpSecSecurityPolicy/TcpIpIpSecSecurityPolicyMechanism` is configured as `SECURED`, if `IP-SEC-RULE/POLICY` is set to `PASSTHROUGH`, the parameter is configured as `BYPASSED`. If `IP-SEC-RULE/DIRECTION` is set to `OUT`, `TcpIpIpIpSecSecurityPolicy/TcpIpIpSecSecurityPolicyDirection` is configured as `OUTBOUND`, if `IP-SEC-RULE/DIRECTION` is set to `IN`, the parameter is configured as `INBOUND`. `TcpIpIpIpSecSecurityPolicy/TcpIpIpSecSecurityPolicySecurityAssociationRef` is configured to refer to the `TcpIpIpSecSecurityAssociationCfg` container that has been created for the `IP-SEC-RULE`. In the `TcpIpIpIpSecSecurityPolicy` container, two `TcpIpIpIpSecSecurityRule` sub containers are created, named `TcpIpIpIpSecSecurityRule_LOCAL` and `TcpIpIpIpSecSecurityRule_REMOTE`. The following parameters are configured: In `TcpIpIpIpSecSecurityRule_LOCAL`, `TcpIpIpIpSecSecurityPolicyDirection` is set to `LOCAL`, in `TcpIpIpIpSecSecurityRule_REMOTE`, the parameter is set to `REMOTE`. `TcpIpIpSecSecurityPolicyUpperLayer` is set to `IP-SEC-RULE/IP-PROTOCOL`. In `TcpIpIpIpSecSecurityRule_LOCAL`, `TcpIpIpIpSecSecurityPolicyStartPort` is set to `IP-SEC-RULE/LOCAL-PORT-RANGE-START`, in `TcpIpIpIpSecSecurityRule_REMOTE`, the parameter is set to `IP-SEC-RULE/REMOTE-PORT-RANGE-START`. In `TcpIpIpIpSecSecurityRule_LOCAL`, `TcpIpIpIpSecSecurityPolicyEndPort` is set to `IP-SEC-RULE/LOCAL-PORT-RANGE-END`, in `TcpIpIpIpSecSecurityRule_REMOTE`, the parameter is set to `IP-SEC-RULE/REMOTE-PORT-RANGE-END`.

3.4.47. Tls

Configuration parameters	Mapping description
Tls/TlsConnection	For every `TLS-CRYPTO-SERVICE-MAPPING` that is referenced by the `APPLICATION-ENDPOINT` of a local or remote `SOCKET-ADDRESS` of the configured `ECU-INSTANCE`, one `TlsConnection` container is created. Its name is set to `TLSConn_<SocketAddressName>`, where `<SocketAddressName>` is the `SHORT-NAME` of the `SOCKET-ADDRESS`. `TlsCipherSuite` is configured depending on the first `TLS-CRYPTO-CIPHER-SUITE` element of the `TLS-CRYPTO-SERVICE-MAPPING`. Section 3.4.47.1, "TlsCipherSuite Mapping" describes the mapping in detail. `TlsEndpoint` is configured using `TLS-CRYPTO-SERVICE-MAPPING/CATEGORY` provided `TLS-CRYPTO-SERVICE-MAPPING/CATEGORY` either contains the value `TLS_CLIENT` or `TLS_SERVER`. `TlsDtls` is configured depending on the protocol of the `SOCKET-ADDRESS`, which can be either TCP or UDP. The parameter is set to `false` for TCP, and to `true` for UDP. The references ▶ `CsmPreMasterSecretKeyRef` ▶ `CsmMasterSecretKeyRef` ▶ `CsmMACGenerateKeyRef` ▶ `CsmMACVerifyKeyRef` ▶ `CsmEncryptKeyRef` ▶ `CsmDecryptKeyRef` ▶ `CsmPRFPreMasterSecretJobRef` ▶ `CsmPRFMasterSecretJobRef` ▶ `CsmMACGenerateJobRef` ▶ `CsmMACVerifyJobRef` ▶ `CsmHashJobRef` ▶ `CsmRandomJobRef` ▶ `CsmEncryptJobRef` ▶ `CsmDecryptJobRef` ▶ `TlsCsmLocalCertSigGenerateJobRef`

Configuration parameters	Mapping description
	are configured to refer to the corresponding containers in the `Csm` module configuration if a valid `TlsCipherSuite` was determined according to [Section 3.4.47.1, "TlsCipherSuite Mapping"](#). Please note that `CsmEncryptKeyRef`, `CsmDecryptKeyRef`, `CsmEncryptJobRef`, and `CsmDecryptJobRef` are not configured for cipher suites that do not provide encryption, e.g. `TLS_PSK_-WITH_NULL_SHA256`.

3.4.47.1. TlsCipherSuite Mapping

The `TlsCipherSuite` parameter is derived in one of two ways from the `TLS-CRYPTO-SERVICE-MAPPING` element. If the `TLS-CRYPTO-SERVICE-MAPPING` contains a `TLS/cipherSuiteId` SDG element which provides a valid TLS cipher suite ID, then the `TlsCipherSuite` is derived from that cipher suite ID. If no cipher suite id is available, `TlsCipherSuite` is derived from the properties of the `CRYPTO-SERVICE-PRIMITIVE` elements which the `TLS-CRYPTO-SERVICE-MAPPING` references via `TLS-CRYPTO-SERVICE-MAPPING/KEY-EXCHANGE-REF`, `TLS-CRYPTO-SERVICE-MAPPING/AUTHENTI-CATION-REF`, and `TLS-CRYPTO-SERVICE-MAPPING/ENCRYPTION-REF`.

The following table shows the configuration of `TlsCipherSuite` depending on the `TLS/cipherSuiteId` SDG element representing a cipher suite id.

Cipher suite Id	TlsCipherSuite
0x00B0	TLS_PSK_WITH_NULL_SHA256
0x00A8	TLS_PSK_WITH_AES_128_GCM_SHA256
0xC006	TLS_ECDHE_ECDSA_WITH_NULL_SHA
0xC02B	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256

The following tables show the configuration of `TlsCipherSuite` depending on the properties of `TLS-CRYPTO-SERVICE-MAPPING/AUTHENTICATION-REF` and `TLS-CRYPTO-SERVICE-MAPPING/ENCRYP-TION-REF`. Each of these `CRYPTO-SERVICE-PRIMITIVE` elements needs to contain a `TLS/keyOrHash-LengthBytes` SDG, and the attributes `ALGORITHM-FAMILY`, and optionally `ALGORITHM-MODE`. For all `CRYP-TO-SERVICE-PRIMITIVE` elements it is required that `ALGORITHM-SECONDARY-FAMILY` is not configured. Table entries that provide "n/a" in all of their fields indicate that the `TLS-CRYPTO-SERVICE-MAPPING` is expected not to refer to a `CRYPTO-SERVICE-PRIMITIVE` for the given reference. Moreover the `TlsCi-pherSuite` either needs to contain at least one element in `TLS-CRYPTO-SERVICE-MAPPING/KEY-EX-CHANGE-REFS` if the represented cipher suite uses key exchange (indicated in the table by `KeyExchange: present`), or must not contain any element if the represented cipher suite does not use key exchange (indicated in the table by `KeyExchange: none`). In `TLS-CRYPTO-SERVICE-MAPPING/KEY-EXCHANGE-REFS` only the presence or absence of elements is checked, not their actual content.

`TLS-CRYPTO-SERVICE-MAPPING` for `TLS_PSK_WITH_NULL_SHA256`

CRYPTO-SERVICE-PRIMITIVE	ALGORITHM-FAMILY	ALGORITHM-MODE	KeyOrHashLengthBytes
KeyExchange: none			
Encryption	n/a	n/a	n/a
Authentication	CRYPTO_ALGO-FAM_SHA2_256	n/a	32

`TlsCryptoServiceMapping` for `TLS_PSK_WITH_AES_128_GCM_SHA256`

CRYPTO-SERVICE-PRIMITIVE	ALGORITHM-FAMILY	ALGORITHM-MODE	KeyOrHashLengthBytes
KeyExchange: none			
Encryption	CRYPTO_ALGO-FAM_AES	CRYPTO_ALGO-MODE_GCM	16
Authentication	CRYPTO_ALGO-FAM_SHA2_256	n/a	32

`TlsCryptoServiceMapping` for `TLS_ECDHE_ECDSA_WITH_NULL_SHA`

CRYPTO-SERVICE-PRIMITIVE	ALGORITHM-FAMILY	ALGORITHM-MODE	KeyOrHashLengthBytes
KeyExchange: present			
Encryption	n/a	n/a	n/a
Authentication	CRYPTO_ALGO-FAM_SHA1	n/a	20

`TlsCryptoServiceMapping` for `TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256`

CRYPTO-SERVICE-PRIMITIVE	ALGORITHM-FAMILY	ALGORITHM-MODE	KeyOrHashLengthBytes
KeyExchange: present			
Encryption	CRYPTO_ALGO-FAM_AES	CRYPTO_ALGO-MODE_GCM	16
Authentication	CRYPTO_ALGO-FAM_SHA2_256	n/a	32

3.4.48. UdpNm

Configuration parameters	Mapping description
`UdpNmGlobalConfig`	`UdpNmComUserDataSupport` is set to `true` if `UdpNmRxUserDataPduRef` or `UdpNmTxUserDataPduRef` is set for any `UdpNmChannelConfig` or if any `UDP-NM-CLUSTER` linked to the imported `ECU-INSTANCE` has its `NM-PNC-PARTICIPATION` either not defined or set to `true`. Otherwise `UdpNmComUserDataSupport` is set to `false`. `UdpNmPnResetTime` is set to `PN-RESET-TIME` of the configured `ECU-INSTANCE`. The following parameters are set using the first `NM-ECU` of the imported `ECU-INSTANCE`: `UdpNmUserDataEnabled` is set to `NM-USER-DATA-ENABLED`. `UdpNmRemoteSleepIndEnabled` is set to `NM-REMOTE-SLEEP-IND-ENABLED`. `UdpNmBusSynchronizationEnabled` is set to `NM-BUS-SYNCHRONIZATION-ENABLED`. `UdpNmStateChangeIndEnabled` is set to `NM-STATE-CHANGE-IND-ENABLED`. `UdpNmPassiveModeEnabled`: see [Section 3.4.4, "CanNm"](#), `CanNmPassiveModeEnabled`. `UdpNmPduRxIndicationEnabled` is set to `NM-PDU-RX-INDICATION-ENABLED`. `UdpNmComControlEnabled` is set to `NM-COM-CONTROL-ENABLED`. `UdpNmMainFunctionPeriod` is set to `NM-CYCLETIME-MAIN-FUNCTION`. `UdpNmComMNetworkHandleRef` references the `ComMChannel` container that is created for the `COMMUNICATION-CLUSTER` referenced in `COMMUNICATION-CLUSTER-REF`. If inconsistencies are detected among parameters of multiple `NM-ECU` elements, a warning is reported. The following parameters are set using the first `UDP-NM-CLUSTER-COUPLING` of all `UDP-NM-CLUSTER` elements connected to the imported `ECU-INSTANCE`:

Configuration parameters	Mapping description
	If inconsistencies are detected among parameters of multiple `UDP-NM-CLUSTER-COUPLING` elements, a warning is reported. If a `UDP-NM-CLUSTER` configured as partial networking cluster (PNC) (see [Section 3.4.14, "EcuC"](#)) belongs to the imported `ECU-INSTANCE`, the following parameters are set: `UdpNmPnEiraRxNSduRef` references the corresponding container in the `EcuC` module configuration. `UdpNmPnEiraCalcEnabled` is set to `true`. `UdpNmPnInfo/UdpNmPnInfoOffset` is set to `PNC-VECTOR-OFFSET`. If `PNC-VECTOR-OFFSET` is not defined, a warning is issued and `UdpNmPnInfoOffset` is not set. `UdpNmPnInfo/UdpNmPnInfoLength` is set to `PNC-VECTOR-LENGTH`. If `PNC-VECTOR-LENGTH` is not defined, a warning is issued and `UdpNmPnInfoLength` is not set. For the configuration of `UdpNmPnFilterMaskByte` all `PNC-FILTER-DATA-MASK` values of the `ETHERNET-COMMUNICATION-CONNECTOR` elements of the configured `ECU-INSTANCE` are taken as input. The configuration algorithm is the same as described for `CanNmPnFilterMaskByte` in [Section 3.4.4, "CanNm"](#).
`UdpNmChannelConfig`	For every `ETHERNET-PHYSICAL-CHANNEL` for which the following conditions hold, one `UdpNmChannelConfig` container is created. ▶ The imported `ECU-INSTANCE` sends or receives at least one `NM-PDU` on the `ETHERNET-PHYSICAL-CHANNEL`. ▶ A `UDP-NM-CLUSTER` which belongs to the imported `ECU-INSTANCE` references the `ETHERNET-PHYSICAL-CHANNEL` either directly via `VLAN-REF` or it references the `ETHERNET-CLUSTER` that contains the `ETHERNET-PHYSICAL-CHANNEL`. The container name is `<PREFIX><name>`, where `<name>` is the `SHORT-NAME` of the `ETHERNET-PHYSICAL-CHANNEL`. A `UDP-NM-CLUSTER` belongs to the imported `ECU-INSTANCE` if at least one of its `UDP-NM-NODE` elements references an `ETHERNET-COMMUNICATION-CONTROLLER` of this `ECU-INSTANCE`.

Configuration parameters	Mapping description
	The `NM-ECU` used for configuring some of the `UdpNmChannelConfig` parameters is the `NM-ECU` which the first `UDP-NM-NODE` references via `NM-IF-ECU-REF`. `UdpNmNodeDetectionEnabled` is set to `NM-CLUSTER/NM-NODE-DETECTION-ENABLED`, or to `NM-ECU/NM-NODE-DETECTION-ENABLED` if `NM-CLUSTER/NM-NODE-DETECTION-ENABLED` is not available. `UdpNmRepeatMsgIndEnabled` is set to `NM-CLUSTER/NM-REPEAT-MSG-IND-ENABLED`, or to `NM-ECU/NM-REPEAT-MSG-IND-ENABLED` if `NM-CLUSTER/NM-REPEAT-MSG-IND-ENABLED` is not available. `UdpNmNodeIdEnabled` is set to `NM-CLUSTER/NM-NODE-ID-ENABLED`, or to `NM-ECU/NM-NODE-ID-ENABLED` if `NM-CLUSTER/NM-NODE-ID-ENABLED` is not available. `UdpNmTimeoutTime` is set to `NM-NETWORK-TIMEOUT`. `UdpNmWaitBusSleepTime` is set to `NM-WAIT-BUS-SLEEP-TIME`. `UdpNmRepeatMessageTime` is set to `NM-REPEAT-MESSAGE-TIME`. `UdpNmRemoteSleepIndTime` is set to `NM-REMOTE-SLEEP-INDICATION-TIME`. `UdpNmMsgCycleTime` is set to `NM-MSG-CYCLE-TIME`. `UdpNmMsgTimeoutTime` is set to `NM-MESSAGE-TIMEOUT-TIME`. `UdpNmImmediateNmCycleTime` is set to `NM-IMMEDIATE-NM-CYCLE-TIME`. `UdpNmImmediateNmTransmissions` is set to `NM-IMMEDIATE-NM-TRANSMISSIONS`. `UdpNmPduNidPosition` is set depending on the value of `NM-NID-POSITION`: ▶ `UDPNM_PDU_BYTE_0` for 0. ▶ `UDPNM_PDU_BYTE_1` for 1. ▶ `UDPNM_PDU_OFF` for any other value. `UdpNmPduCbvPosition` is set depending on the value of `NM-CBV-POSITION`: ▶ `UDPNM_PDU_BYTE_0` for 0. ▶ `UDPNM_PDU_BYTE_1` for 1.

Configuration parameters	Mapping description
	► UDPNM_PDU_OFF for any other value. UdpNmRxPdu/<EcuC container name>/UdpNmRxPduRef references the container in the EcuC module which has been created for the Rx NM-PDU. The Rx NM-PDU is the first NM-PDU that is transmitted on the ETHERNET-PHYSICAL-CHANNEL and that is referenced via RX-NM-PDU-REFS/RX-NM-PDU-REF of the NM-NODE associated with the ETHERNET-PHYSICAL-CHANNEL. The NM-NODE associated with an ETHERNET-PHYSICAL-CHANNEL is the first NM-NODE that receives an NM-PDU on the ETHERNET-PHYSICAL-CHANNEL. <EcuC container name> is the name of the referenced EcuC container. If no Rx NM-PDU is available, a warning is reported. If an NmUserDataPdu container has been created for the NM-PDU in the EcuC module configuration, UdpNmUserDataRxPdu/UdpNmRxUserDataPduRef references this container. The name of UdpNmUserDataRxPdu is set to the name of the referenced EcuC container. UdpNmTxPdu/UdpNmTxPduRef references the container in the EcuC module which has been created for the Tx NM-PDU. The Tx NM-PDU is the first NM-PDU that is transmitted on the ETHERNET-PHYSICAL-CHANNEL and that is referenced via TX-NM-PDU-REFS/TX-NM-PDU-REF of the NM-NODE associated with the ETHERNET-PHYSICAL-CHANNEL. The name of UdpNmTxPdu is set to the name of the referenced EcuC container. If an NmUserDataPdu container has been created for the NM-PDU in the EcuC module configuration, UdpNmUserDataTxPdu/UdpNmTxUserDataPduRef references this container. The name of UdpNmUserDataTxPdu is set to the name of the referenced EcuC container. The following parameters are set using the first NM-NODE of the UDP-NM-CLUSTER connected to the imported ECU-INSTANCE. If inconsistencies are detected among parameters of multiple NM-NODE elements, a warning is reported. UdpNmMsgCycleOffset is set to NM-MSG-CYCLE-OFFSET. UdpNmNodeId is set to NM-NODE-ID. The configuration of PNC-related parameters is done in analogy to the configuration of the PNC-related CanNm parameters, see Section 3.4.4, "CanNm". The parameters and configuration containers obtain the prefix UdpNmPn. UdpNmPnFilterMaskByte is not configured.

3.4.49. Xcp

Configuration parameters	Mapping description
`Xcp/XcpConfig/XcpPdu/XcpTxPdu`	For every `GENERAL-PURPOSE-I-PDU` that the imported `ECU-INSTANCE` sends without routing it (see [Section 3.3.6, "PDU routing"](#)) and which has its `CATEGORY` field set to `XCP`, a `XcpTxPdu` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `GENERAL-PURPOSE-I-PDU`. `XcpTxPdu/XcpTxPduRef` is set to reference the configuration container of the `GENERAL-PURPOSE-I-PDU` in the `EcuC` module configuration.
`Xcp/XcpConfig/XcpPdu/XcpRxPdu`	For every `GENERAL-PURPOSE-I-PDU` that the imported `ECU-INSTANCE` receives without routing it (see [Section 3.3.6, "PDU routing"](#)) and which has its `CATEGORY` field set to `XCP`, a `XcpRxPdu` container is created. The container name is `<PREFIX><name><INSTSUFFIX>`, where `<name>` is the `SHORT-NAME` of the `GENERAL-PURPOSE-I-PDU`. `XcpRxPdu/XcpRxPduRef` is set to reference the configuration container of the `GENERAL-PURPOSE-I-PDU` in the `EcuC` module configuration.

4. Bibliography

Bibliography

[1] *AUTOSAR System Template*, Issue Version 4.2.0, Release 4.0, Revision 3, Publisher: AUTOSAR

[2] *LIN Specification Package Revision 2.0*, Publish date: September 23, 2003, Publisher: LIN Consortium

[3] *AUTOSAR System Template*, Issue Version 4.6.0, Release 4.2, Revision 2, Publisher: AUTOSAR

[4] *AUTOSAR System Template*, Issue AUTOSAR CP Release 19-11, Publisher: AUTOSAR

[5] *AUTOSAR System Template*, Issue AUTOSAR CP Release 4.4.0, Publisher: AUTOSAR

[6] *AUTOSAR System Template*, Issue AUTOSAR CP Release 19-11, Publisher: AUTOSAR