

Document Title	Specification of Diagnostic Communication Manager
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	18

Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R25-11

Document Change History			
Date	Release	Changed by	Description
2025-11-27	R25-11	AUTOSAR Release Management	• DcmDspRoutineInfoByte support for UDS • OBD RID ranges • DTCSettingControlOptionRecord optional in ControlDTCSetting service • Authentication distance supervision • Multiple security events for single diagnostic service • Dcm_StatusType removed • DcmDspMaxDidToRead for UDS • Support of 64-bit DiagnosticDataElements • Minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2024-11-27	R24-11	AUTOSAR Release Management	• DCM Context Data for IdsM event • Improvement of various Requirements related to Authentication Service 0x29 • Remove protocol type and related APIs • Minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation





2023-11-23	R23-11	AUTOSAR Release Management	• Fixed StartOfReception • Added Authentication TransmitCertificate • Introduced Secure Variant Coding • Reworked the structure of elements in SecurityEvents to enable usage in multiple platforms • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2022-11-24	R22-11	AUTOSAR Release Management	• Diagnostic Extract shall be completed to describe multi-dimensional arrays and arrays with complex data types • Update RoE to match latest ISO 14229-1 • Handling of Dem_GetNumberOfFreezeFrameRecords • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2021-11-25	R21-11	AUTOSAR Release Management	• Incorporation of Concept 670 Classic Platform Flexibility • Separated SAE J2012_4 DTCs and UDS DTCs • OBD on UDS shall be supported • Removed the Mirror Memory following ISO 14229-1:2020 • Support subfunctions 1A and 56 for UDS services 0x19
2020-11-30	R20-11	AUTOSAR Release Management	• Incorporation of Concept 671 Intrusion Detection System Manager • Added DcmDspExternalSRDataElementClass • Updated the Error Classification chapter • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation





2019-11-28	R19-11	AUTOSAR Release Management	• Incorporation of Concept NVData Handling Enhancements • Incorporation of PeriodicDID Scheduler Type2 • Renaming of the SRS_Diagnostics to RS_Diagnostics • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• Incorporation of Concept 636 Security Extensions • Rework of SenderReceiver interface support for DIDs: Atomic SenderReceiver interfaces added. • Rework of SenderReceiver interface support for controlling DIDs via service InputOutputControlByIdentifier (0x2F) • Support added for input signals for the RequestRoutineResults (0x03) subfunction of the RoutineControl (0x31) service • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2017-12-08	4.3.1	AUTOSAR Release Management	• Cleanup SRS_Diagnostic requirement traceability • Fix Dcm/Dem interactions inconsistencies • Add constraints requirements for parameter configuration • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation





2016-11-30	4.3.0	AUTOSAR Release Management	• Redesign interfaces between Dem and Dcm • Rework Security Access management • Add management for parallel support for OBD and UDS protocols • Clarify usage of Diagnosis scaling • minor corrections / clarifications / editorial changes; For details please refer to the BWCStatement
2015-07-31	4.2.2	AUTOSAR Release Management	• Specify the NRCS to be sent by the Dcm in case of Dem interfaces return negative values. • Clarify Routine operation prototypes • Debugging support marked as obsolete • Minor corrections / clarifications / editorial changes; For details please refer to the Change Documentation
2014-10-31	4.2.1	AUTOSAR Release Management	• Update to ISO 14229-1:2013 (Order of NRCS, SID 0x19 and 0x28 extended subfunctions, SID 0x38) • Specify security mechanisms (security Lock time, static seed). • Refine service ReadDataByPeriodicIdentifier (0x2A) and provide UUDT transfer. • Reorganize the configuration parameters for the routines.
2014-03-31	4.1.3	AUTOSAR Release Management	• Added functional description for DIDRange usage • Added support for bootloader interaction • Revised the header file structure • Editorial changes





2013-10-31	4.1.2	AUTOSAR Release Management	• Created API tables for service interfaces • Provided synchronous and asynchronous APIs for DataServices callouts • Harmonization for the length parameter interpretation all over RDBI, WDBI and RC services to be in bytes • Editorial changes • Removed chapter(s) on change documentation
2013-03-15	4.1.1	AUTOSAR Administration	• Added Response on Event support • Rework configuration for S/R communication • Rework OBD Service \$06 management
2011-12-22	4.0.3	AUTOSAR Administration	• Change interaction with BswM module for mode management • Change of callout configuration management for services and sub-services processing • Synchronous and asynchronous clarification
2009-12-18	4.0.1	AUTOSAR Administration	• ComM_DCM_InactiveDiagnostic and ComM_DCM_ActiveDiagnostic has been defined as mandatory interfaces. • DcmDslPeriodicTxConfirmationPduld multiplicity changed and creation of DcmDslPeriodicConnection parameter in order to link the confirmation Id with TxPdu Id for PeriodicTransmission. • Dem_GetDTCOfOBDFreezeFrame, Dlt_ConditionCheckRead added as optional interfaces • DsplInternal_<DiagnosticService> Api moved to mandatory internal interface to support the ECU Supplier diagnosis. • Rework of ReadData operation





2009-12-04	3.1.4	AUTOSAR Administration	• Add support of following UDS services : ReadMemoryByAddress, WriteMemoryByAddress, RequestDownload, RequestUpload, TransferData, RequestTransferExit, CommunicationControl, ResponseOnEvent. • Add of bootloader interaction • Add of BswM interaction • Add of IoHwAb interaction • Add of DLT interaction • Add of Signal based approach on RTE interfaces • Legal nvocation revised
2008-08-13	3.1.1	AUTOSAR Administration	• Introduction of OBD support • generation of artefacts from the models according to the AUTOSAR process • Identification of requirements and correct formulation of specification items as requirements • General cleanup • Legal nvocation revised
2007-12-21	3.0.1	AUTOSAR Administration	• Rework of the interfaces with RTE (remove of Central Diagnostic SWC concept) • Correction of issues identified on R2.1 • Document meta information extended • Small layout adaptations made
2007-01-24	2.1.15	AUTOSAR Administration	• "Advice for users" revised • "Revision Information" added



△

	2.1.14	AUTOSAR Administration	• Corrections in configuration chapter • Rework on interface between DCM and DEM according to changes in DEM SWS • Corrections in Sequence diagram • Addition of header files inclusions • Legal disclaimer revised
2006-11-28	2.1	AUTOSAR Administration	• Layout Adaptations
2006-05-16	2.0	AUTOSAR Administration	• Document structure adapted to common Release 2.0 SWS Template • Major changes in chapter 10 • Structure of document changed partly • Other changes see chapter 11
2005-05-31	1.0	AUTOSAR Administration	• Initial release

Disclaimer

This work (specification and/or software implementation) and the material contained in it, as released by AUTOSAR, is for the purpose of information only. AUTOSAR and the companies that have contributed to it shall not be liable for any use of the work.

The material contained in this work is protected by copyright and other types of intellectual property rights. The commercial exploitation of the material contained in this work requires a license to such intellectual property rights.

This work may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only. For any other purpose, no part of the work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The work has been developed for automotive applications only. It has neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Table of Contents

1	Introduction and functional overview	24
2	Acronyms and Abbreviations	27
2.1	Typographical Conventions	29
3	Related documentation	30
3.1	Input documents & related standards and norms	30
3.2	Related specification	31
4	Constraints and assumptions	32
4.1	Limitations	32
4.2	Applicability to car domains	33
4.3	Applicability to emission-related environments (OBD)	34
5	Dependencies to other modules	35
6	Requirements Tracing	37
7	Functional specification	43
7.1	General design elements	43
7.1.1	Submodules within the <code>Dcm</code> module	43
7.1.2	Negative Response Code (NRC)	44
7.1.3	Non-volatile information	45
7.1.4	Types	45
7.1.4.1	Atomic types overview	45
7.1.4.2	Data array types overview	46
7.1.4.3	Nested Data types overview	46
7.1.4.4	Data types constraints	51
7.1.4.5	<code>Dcm_OpStatusType</code>	52
7.1.4.6	<code>Dcm_Cemr_{DID}Type</code>	52
7.2	Diagnostic Session Layer (DSL)	53
7.2.1	Introduction	53
7.2.2	Use cases	53
7.2.3	Interaction with other modules	53
7.2.4	Functional description	54
7.2.4.1	Overview	54
7.2.4.2	Forward requests from the <code>PduR</code> module to the <code>DSD</code> submodule	55
7.2.4.2.1	<code>Dcm_StartOfReception</code>	56
7.2.4.2.2	<code>Dcm_CopyRxData</code>	58
7.2.4.2.3	<code>Dcm_TpRxIndication</code>	58
7.2.4.3	Concurrent <code>TesterPresent</code> ("keep alive logic")	58
7.2.4.3.1	<code>Dcm_CopyTxData</code>	59
7.2.4.3.2	<code>Dcm_TpTxConfirmation</code>	59
7.2.4.4	Forward responses from the <code>DSD</code> submodule to the <code>PduR</code> module	60

7.2.4.5	Generic Connection Handling	61
7.2.4.6	Guarantee timing to tester by sending busy responses	62
7.2.4.7	Support of periodic transmission	63
7.2.4.8	Support of segmented response (paged-buffer)	63
7.2.4.9	Support of ResponsePending response triggered by the Application	64
7.2.4.10	Manage security level	64
7.2.4.10.1	Initialization sequence	65
7.2.4.10.2	AttemptCounter update	66
7.2.4.11	Manage session state	67
7.2.4.12	Manage authentication state	67
7.2.4.13	Non-default sessions	70
7.2.4.14	Allow to modify timings	71
7.2.4.14.1	Different service tables	72
7.2.4.14.2	Prioritization of protocol	72
7.2.4.14.3	Preemption of protocol	72
7.2.4.14.4	Parallel diagnostic protocol processing	74
7.2.4.15	Manage resources	76
7.2.4.16	Communication Mode Handling	77
7.2.4.16.1	No Communication	77
7.2.4.16.2	Silent Communication	78
7.2.4.16.3	Full Communication	79
7.2.4.16.4	Diagnostic Activation State	79
7.3	Diagnostic Service Dispatcher (DSD)	81
7.3.1	Introduction	81
7.3.2	Use cases	81
7.3.2.1	Receive a request message and transmit a positive response message	81
7.3.2.2	Receive a request message and suppress a positive response	82
7.3.2.3	Receive a request message and suppress a negative response	82
7.3.2.4	Receive a request message and transmit a negative response message	82
7.3.2.5	Send a positive response message without corresponding request	82
7.3.2.6	Segmented Responses (paged-buffer)	83
7.3.3	Interaction of the DSD with other modules	83
7.3.3.1	Interaction of the DSD with the DSL main functionality	84
7.3.3.2	Interaction of the DSD with the DSP	84
7.3.4	Functional Description of the DSD	84
7.3.4.1	Support checking the diagnostic service identifier and adapting the diagnostic message	84
7.3.4.2	Handling of "suppressPosRspMsgIndicationBit"	86
7.3.4.3	Verification functionality	87
7.3.4.3.1	Verification of the diagnostic service access rights	87

7.3.4.3.2	Verification of the Diagnostic Session	90
7.3.4.3.3	Verification of the Service Security Access levels	90
7.3.4.3.4	Verification of the Service mode dependencies	91
7.3.4.4	Check format and subfunction support	91
7.3.4.4.1	Verification of the Manufacturer Application environmen- t/permission	92
7.3.4.4.2	Verification of the Supplier Application environment/per- mission	92
7.3.4.5	Distribution of diagnostic message to <code>DSP</code> submodule	93
7.3.4.6	Assemble positive or negative response	93
7.3.4.6.1	Positive Response	94
7.3.4.6.2	Negative Response	94
7.3.4.6.3	Suppression of response	94
7.3.4.7	Initiate transmission	95
7.4	Diagnostic Service Processing (DSP)	96
7.4.1	General	96
7.4.1.1	Check format and subfunction support	96
7.4.1.2	Assemble response	97
7.4.1.3	Negative Response Codes handling	97
7.4.1.4	Diagnostic mode declaration groups	97
7.4.1.5	Environmental condition dependent execution	99
7.4.1.6	Sender/Receiver Communication	102
7.4.1.7	Passing <code>SwDataDefProps</code> properties from DEXT file to the <code>Dcm</code> Service <code>SW-C</code>	103
7.4.1.7.1	<code>DcmDspDiagnosticDataElementRef</code> workflow	103
7.4.1.7.2	<code>DcmDspAlternativeDataType.DcmApplication-</code> <code>DataType</code> workflow	104
7.4.1.8	Asynchronous call behavior	104
7.4.2	UDS Services	105
7.4.2.1	General behavior using DEM interfaces	106
7.4.2.2	Service 0x10 - Diagnostic Session Control	107
7.4.2.3	Service 0x11 - ECUReset	108
7.4.2.4	Service 0x14 - Clear Diagnostic Information	109
7.4.2.5	Service 0x19 - Read <code>DTC</code> Information	110
7.4.2.5.1	Subfunctions 0x01, 0x07 and 0x12	111
7.4.2.5.2	Subfunctions 0x02, 0x0A, 0x13, 0x15 and 0x17	113
7.4.2.5.3	Subfunction 0x08	115
7.4.2.5.4	Subfunction 0x09	116
7.4.2.5.5	Subfunctions 0x06/0x19	118
7.4.2.5.6	Subfunction 0x03	119
7.4.2.5.7	Subfunctions 0x04 and 0x18	120
7.4.2.5.8	Subfunction 0x05	122
7.4.2.5.9	Subfunctions 0x0B, 0x0C, 0x0D and 0x0E	123
7.4.2.5.10	Subfunction 0x14	124

7.4.2.5.11	Subfunction 0x1A	126
7.4.2.5.12	Subfunction 0x42	126
7.4.2.5.13	Subfunction 0x55	128
7.4.2.5.14	Subfunction 0x56	129
7.4.2.6	Service 0x22 - ReadDataByIdentifier	129
7.4.2.6.1	UDS DID	133
7.4.2.6.2	OBD DID	135
7.4.2.7	Service 0x24 - ReadScalingDataByIdentifier	137
7.4.2.8	Service 0x27 - SecurityAccess	137
7.4.2.9	Service 0x28 - CommunicationControl	140
7.4.2.10	Service 0x29 - Authentication	143
7.4.2.10.1	De-authentication	145
7.4.2.10.2	Verify Certificates	146
7.4.2.10.3	Proof of ownership client	152
7.4.2.10.4	Definition and verification of roles	154
7.4.2.10.5	Definition and verification of white lists	155
7.4.2.10.6	AuthenticationConfiguration	159
7.4.2.10.7	TransmitCertificate	159
7.4.2.11	Service 0x2A - ReadDataByPeriodicIdentifier	160
7.4.2.11.1	Scheduler PeriodicTransmission	162
7.4.2.12	Service 0x2C - DynamicallyDefineDataIdentifier	172
7.4.2.13	Service 0x2E - WriteDataByIdentifier	173
7.4.2.14	Service 0x2F - InputOutputControlByIdentifier	177
7.4.2.15	Service 0x31 - RoutineControl	189
7.4.2.15.1	Handling of OBD RID	197
7.4.2.16	Service 0x3E - Tester Present	198
7.4.2.17	Service 0x3D - WriteMemoryByAddress	198
7.4.2.18	Service 0x23 - ReadMemoryByAddress	200
7.4.2.19	Service 0x34 - RequestDownload	201
7.4.2.20	Service 0x35 - RequestUpload	203
7.4.2.21	Service 0x36 - TransferData	204
7.4.2.22	Service 0x37 - RequestTransferExit	206
7.4.2.23	Service 0x38 - RequestFileTransfer	208
7.4.2.24	Service 0x85 - ControlDTCSetting	210
7.4.2.25	Service 0x86 - ResponseOnEvent (RoE)	212
7.4.2.26	Service 0x87 - LinkControl	214
7.4.3	OBD Services	215
7.4.3.1	Overview	215
7.4.3.2	General behavior	215
7.4.3.3	Service \$01 - Request Current Powertrain Diagnostic Data	216
7.4.3.4	Service \$02 - Request Power Train FreezeFrame Data	219
7.4.3.4.1	Service \$02 - PID\$02	219
7.4.3.4.2	Service \$02 - availability PID	220

7.4.3.4.3	Service \$02 - other PIDs	220
7.4.3.5	Service \$03 \$07 \$0A - Obtaining DTCs	221
7.4.3.6	Service \$04 - Clear/reset emission-related diagnostic information	222
7.4.3.7	Service \$06 - Request On-Board Monitoring Test-results for Specific Monitored Systems	223
7.4.3.7.1	General requirements	223
7.4.3.7.2	Test results obtained via <i>Dem</i> interaction	224
7.4.3.8	Service \$08 - Request Control of On-Board System, Test or Component	224
7.4.3.9	Service \$09 - Request Vehicle Information	225
7.4.4	Interaction usecases	227
7.4.4.1	Jump to Bootloader	227
7.4.4.2	Jump due to ECUReset	230
7.4.4.3	Jump from Bootloader / ECUReset	231
7.4.4.4	Flags management	231
7.4.4.4.1	Jump to Bootloader	231
7.4.4.4.2	Jump from Bootloader	232
7.5	Error notification	232
7.6	Synchronous and Asynchronous implementation	232
7.7	DID configuration	233
7.7.1	Individual DID	234
7.7.2	DID ranges	235
7.8	Secure Coding	236
7.9	Startup behavior	242
7.10	Error Classification	242
7.10.1	Development Errors	242
7.10.2	Runtime Errors	243
7.10.3	Production Errors	243
7.10.4	Extended Production Errors	243
7.11	Security Events	243
8	API specification	259
8.1	Imported types	259
8.2	Type definitions	260
8.2.1	Dcm_CommunicationModeType	260
8.2.2	Dcm_ConfigType	261
8.2.3	Dcm_ReturnReadMemoryType	261
8.2.4	Dcm_ReturnWriteMemoryType	262
8.2.5	Dcm_EcuStartModeType	262
8.2.6	Dcm_ProgConditionsType	262
8.2.7	Dcm_MsgItemType	263
8.2.8	Dcm_MsgType	263
8.2.9	Dcm_MsgLenType	264
8.2.10	Dcm_MsgAddInfoType	264

8.2.11 Dcm_IdContextType	265
8.2.12 Dcm_MsgContextType	265
8.2.13 Dcm_ExtendedOpStatusType	266
8.3 Function definitions	267
8.3.1 Functions provided for other BSW components	267
8.3.1.1 Dcm_Init	267
8.3.1.2 Dcm_GetVersionInfo	268
8.3.1.3 Dcm_DemTriggerOnDTCStatus	268
8.3.1.4 Dcm_GetVin	269
8.3.1.5 Dcm_BndMWriteBlockFinish	269
8.3.2 Functions provided to BSW modules and to SW-Cs	270
8.3.2.1 Dcm_SetDeauthenticatedRole	270
8.3.2.2 Dcm_GetSecurityLevel	270
8.3.2.3 Dcm_GetSesCtrlType	271
8.3.2.4 Dcm_ResetToDefaultSession	272
8.3.2.5 Dcm_SetActiveDiagnostic	272
8.4 Callback notifications	272
8.4.1 Dcm_StartOfReception	273
8.4.2 Dcm_CopyRxData	274
8.4.3 Dcm_TpRxIndication	275
8.4.4 Dcm_CopyTxData	275
8.4.5 Dcm_TpTxConfirmation	276
8.4.6 Dcm_TxConfirmation	277
8.4.7 Dcm_ComM_NoComModeEntered	277
8.4.8 Dcm_ComM_SilentComModeEntered	278
8.4.9 Dcm_ComM_FullComModeEntered	278
8.4.10 Dcm_CsmAsyncJobFinished	279
8.4.11 Dcm_KeyMAsyncCertificateVerifyFinished	279
8.5 Callout Definitions	280
8.5.1 Dcm_ReadMemory	281
8.5.2 Dcm_WriteMemory	282
8.5.3 Dcm_SetProgConditions	283
8.5.4 Dcm_GetProgConditions	283
8.5.5 Dcm_ProcessRequestAddFile	284
8.5.6 Dcm_ProcessRequestDeleteFile	285
8.5.7 Dcm_ProcessRequestReplaceFile	286
8.5.8 Dcm_ProcessRequestReadFile	287
8.5.9 Dcm_ProcessRequestReadDir	288
8.5.10 Dcm_WriteFile	289
8.5.11 Dcm_ReadFileOrDir	290
8.5.12 UploadDownloadServices	291
8.5.12.1 ProcessRequestDownload	291
8.5.12.2 ProcessRequestTransferExit	292

8.5.12.3 ProcessRequestUpload	292
8.5.12.4 ProcessTransferDataRead	293
8.5.12.5 ProcessTransferDataWrite	294
8.6 Scheduled functions	295
8.6.1 Dcm_MainFunction	295
8.7 Expected interfaces	295
8.7.1 Mandatory interfaces	296
8.7.2 Optional interfaces	296
8.7.3 Configurable interfaces	300
8.7.3.1 SecurityAccess	300
8.7.3.1.1 GetSeed	300
8.7.3.1.2 CompareKey	302
8.7.3.1.3 GetSecurityAttemptCounter	302
8.7.3.1.4 SetSecurityAttemptCounter	303
8.7.3.2 DataServices	303
8.7.3.2.1 ReadData	303
8.7.3.2.2 WriteData	306
8.7.3.2.3 ReadDataLength	310
8.7.3.2.4 ConditionCheckRead	312
8.7.3.2.5 GetScalingInformation	313
8.7.3.2.6 ReturnControlToECU	316
8.7.3.2.7 ResetToDefault	318
8.7.3.2.7.1 Synchronous interface	318
8.7.3.2.7.2 Asynchronous interface	320
8.7.3.2.8 FreezeCurrentState	321
8.7.3.2.8.1 Synchronous interface	321
8.7.3.2.8.2 Asynchronous interface	323
8.7.3.2.9 ShortTermAdjustment	325
8.7.3.2.9.1 Synchronous interface	325
8.7.3.2.9.2 Asynchronous interface	327
8.7.3.3 DataServices_DIDRange	329
8.7.3.3.1 IsDidAvailable	329
8.7.3.3.2 ReadDidData	330
8.7.3.3.3 WriteDidData	331
8.7.3.3.4 ReadDidRangeDataLength	331
8.7.3.4 InfoTypesServices	332
8.7.3.4.1 GetInfotypeValueData	332
8.7.3.5 RoutineServices	333
8.7.3.5.1 Xxx_Start Operation	333
8.7.3.5.2 Xxx_StartConfirmation Operation	335
8.7.3.5.3 Xxx_Stop Operation	335
8.7.3.5.4 Xxx_StopConfirmation Operation	337
8.7.3.5.5 Xxx_RequestResults Operation	338

8.7.3.5.6	Xxx_RequestResultsConfirmation Operation	340
8.7.3.6	RequestControlServices	340
8.7.3.6.1	RequestControl callout	340
8.7.3.7	ServiceRequestNotification	341
8.7.3.7.1	Indication	341
8.7.3.7.2	Confirmation	342
8.7.3.8	ClearDTCCheckFnc	342
8.8	Dcm as Service-Component	343
8.8.1	Implementation Data Types	343
8.8.1.1	Dcm_OpStatusType	343
8.8.1.2	Dcm_ConfirmationStatusType	343
8.8.1.3	Dcm_SecLevelType	344
8.8.1.4	Dcm_SesCtrlType	344
8.8.1.5	Dcm_ProtocolType	345
8.8.1.6	Dcm_NegativeResponseCodeType	346
8.8.1.7	Dcm_DataElementType_{Data}Type	350
8.8.1.8	Dcm_DataElement_{Data}StructuredType	351
8.8.1.9	Dcm_DataElementType_{Data}ArrayType	351
8.8.1.10	Dcm_DataElementType_{Data}PrimitiveType	352
8.8.1.11	Dcm_DataArrayTypeUint8_{Data}Type	354
8.8.1.12	{DID}_Struct_DataType	354
8.8.1.13	Dcm_RangeArray_{Range}Type	355
8.8.1.14	Dcm_InfoTypeServicesArray_{VehInfoData}Type	355
8.8.1.15	Dcm_RequestControlServicesInArray_{Tid}Type	355
8.8.1.16	Dcm_RequestControlServicesOutArray_{Tid}Type	356
8.8.1.17	Dcm_ScalingInfoArray_{Data}Type	356
8.8.1.18	Dcm_RequestDataOut_{Routine}_{Signal}PrimitivType	356
8.8.1.19	Dcm_RequestDataIn_{Routine}_{Signal}PrimitiveType	358
8.8.1.20	Dcm_RequestDataOut_{Routine}_{Signal}Type	359
8.8.1.21	Dcm_RequestDataOut_{Routine}_{Signal}StructuredType	359
8.8.1.22	Dcm_RequestDataIn_{Routine}_{Signal}Type	360
8.8.1.23	Dcm_RequestDataIn_{Routine}_{Signal}StructuredType	360
8.8.1.24	Dcm_RequestDataOut_{Routine}_{Signal}ArrayType	361
8.8.1.25	Dcm_RequestDataIn_{Routine}_{Signal}ArrayType	362
8.8.1.26	Dcm_RequestFlexibleOutArrayData_{Routine}_{Signal}Type	363
8.8.1.27	Dcm_RequestFlexibleInArrayData_{Routine}_{Signal}Type	363
8.8.1.28	Dcm_StartDataIn_{Routine}_{Signal}PrimitivType	364
8.8.1.29	Dcm_StartDataIn_{Routine}_{Signal}Type	365
8.8.1.30	Dcm_StartDataIn_{Routine}_{Signal}StructuredType	365
8.8.1.31	Dcm_StartDataIn_{Routine}_{Signal}ArrayType	366
8.8.1.32	Dcm_StartDataOut_{Routine}_{Signal}PrimitivType	367
8.8.1.33	Dcm_StartDataOut_{Routine}_{Signal}Type	368
8.8.1.34	Dcm_StartDataOut_{Routine}_{Signal}StructuredType	368

8.8.1.35 Dcm_StartDataOut_{Routine}_{Signal}ArrayType	369
8.8.1.36 Dcm_StartFlexibleInArrayData_{Routine}_{Signal}Type	370
8.8.1.37 Dcm_StartFlexibleOutArrayData_{Routine}_{Signal}Type	370
8.8.1.38 Dcm_StopDataIn_{Routine}_{Signal}PrimitivType	370
8.8.1.39 Dcm_StopDataIn_{Routine}_{Signal}Type	371
8.8.1.40 Dcm_StopDataIn_{Routine}_{Signal}StructuredType	372
8.8.1.41 Dcm_StopDataIn_{Routine}_{Signal}ArrayType	372
8.8.1.42 Dcm_StopDataOut_{Routine}_{Signal}PrimitivType	373
8.8.1.43 Dcm_StopDataOut_{Routine}_{Signal}Type	374
8.8.1.44 Dcm_StopDataOut_{Routine}_{Signal}StructuredType	375
8.8.1.45 Dcm_StopDataOut_{Routine}_{Signal}ArrayType	375
8.8.1.46 Dcm_StopFlexibleInArrayData_{Routine}_{Signal}Type	377
8.8.1.47 Dcm_StopFlexibleOutArrayData_{Routine}_{Signal}Type	377
8.8.1.48 Dcm_KeyArray_{SecurityLevel}Type	378
8.8.1.49 Dcm_SeedArray_{SecurityLevel}Type	378
8.8.1.50 Dcm_SecurityAccessDataRecordArray_{SecurityLevel}Type	378
8.8.1.51 Dcm_RequestDataArrayType	379
8.8.1.52 Dcm_ControlMask_{DID}Type	379
8.8.1.53 Dcm_inputOutputControlParameterType	380
8.8.1.54 Dcm_IOOperationRequest_{DID}Type	380
8.8.1.55 Dcm_IOOperationResponseType	381
8.8.1.56 Dcm_DidSupportedType	381
8.8.1.57 Dcm_FileAndDirNameType	382
8.8.1.58 Dcm_ResponseDataArrayType	382
8.8.1.59 Dcm_AuthenticationRoleType	383
8.8.1.60 Dcm_ControlMask_{Data}ArrayType	383
8.8.1.61 Dcm_ControlMask_{Data}PrimitiveType	383
8.8.1.62 Dcm_Cemr_{DID}Type	384
8.8.2 Sender-Receiver-Interfaces	385
8.8.2.1 DataServices_{DID}	385
8.8.2.2 DataServices_{Data}	385
8.8.2.3 IOControlRequest_{DID}	386
8.8.2.4 IOControlResponse_{DID}	387
8.8.3 Client-Server-Interfaces	387
8.8.3.1 SecurityAccess_{SecurityLevel}	387
8.8.3.2 DataServices_{Data}	390
8.8.3.2.1 ReadData	411
8.8.3.2.2 WriteData	411
8.8.3.2.3 ReadDataLength	411
8.8.3.2.4 ConditionCheckRead	412
8.8.3.2.5 GetScalingInformation	412
8.8.3.2.6 ReturnControlToEcu	412
8.8.3.2.7 ResetToDefault	412

8.8.3.2.8	FreezeCurrentState	412
8.8.3.2.9	ShortTermAdjustment	412
8.8.3.3	DataService_DIDRange_{Range}	412
8.8.3.4	InfotypeServices_{VehInfoData}	415
8.8.3.5	RoutineServices_{RoutineName}	417
8.8.3.6	RequestControlServices_{Tid}	431
8.8.3.7	ServiceRequestNotification	432
8.8.3.8	UploadDownloadServices	435
8.8.3.9	DCMServices	448
8.8.3.10	Authentication	449
8.8.4	NvDataInterface	450
8.8.4.1	DataService_{DID}	450
8.8.5	Ports	450
8.8.5.1	DataService_DIDRange_{Range}	452
8.8.5.2	DataService_{DID}	452
8.8.5.3	DataService_{Data}	453
8.8.5.4	IOControlRequest_{DID}	455
8.8.5.5	IOControlResponse_{DID}	455
8.8.5.6	DCMServices	455
8.8.5.7	InfotypeServices_{VehInfoData}	456
8.8.5.8	RequestControlServices_{Tid}	456
8.8.5.9	ServiceRequestManufacturerNotification_{Name}	456
8.8.5.10	ServiceRequestSupplierNotification_{Name}	457
8.8.5.11	RoutineServices_{RoutineName}	457
8.8.5.12	SecurityAccess_{SecurityLevel}	457
8.8.5.13	Dcm_DiagnosticSessionControlModeSwitchInterface	458
8.8.5.14	Dcm_EcuResetModeSwitchInterface	458
8.8.5.15	Dcm_ModeRapidPowerShutDownModeSwitchInterface	458
8.8.5.16	Dcm_CommunicationControlModeSwitchInterface_{ComM-ChannelName}	459
8.8.5.17	Dcm_ControlDTCSettingModeSwitchInterface	459
8.8.5.18	Dcm_SecurityAccessModeSwitchInterface	459
8.8.5.19	Dcm_UploadDownloadServices	460
8.8.5.20	Dcm_Authentication_{Connection}	460
8.8.5.21	Dcm_AuthenticationStateMSI	460
8.8.6	ModeDeclarationGroups	461
8.8.6.1	DcmDiagnosticSessionControl	461
8.8.6.2	DcmEcuReset	461
8.8.6.3	DcmModeRapidPowerShutDown	462
8.8.6.4	DcmCommunicationControl	462
8.8.6.5	DcmControlDTCSetting	463
8.8.6.6	DcmSecurityAccess	463
8.8.6.7	DcmAuthenticationState	463

8.8.7	Mode-Switch-Interfaces	464
8.8.7.1	Dcm_DiagnosticSessionControlModeSwitchInterface	464
8.8.7.2	Dcm_EcuResetModeSwitchInterface	464
8.8.7.3	Dcm_ModeRapidPowerShutDownModeSwitchInterface	464
8.8.7.4	Dcm_CommunicationControlModeSwitchInterface	465
8.8.7.5	Dcm_ControlDTCSettingModeSwitchInterface	465
8.8.7.6	Dcm_SecurityAccessModeSwitchInterface	465
8.8.7.7	Dcm_AuthenticationStateModeSwitchInterface	466
8.9	External diagnostic service processing	466
8.9.1	<Module>_<DiagnosticService>	466
8.9.2	<Module>_<DiagnosticService>_<SubService>	467
8.10	Internal interfaces (not normative)	467
8.10.1	DslInternal_SetSecurityLevel	468
8.10.2	DslInternal_SetSesCtrlType	468
8.10.3	DsplInternal_DcmConfirmation	468
8.10.4	DsdInternal_StartPagedProcessing	468
8.10.5	DsplInternal_CancelPagedBufferProcessing	468
8.10.6	DsdInternal_ProcessPage	468
9	Sequence diagrams	469
9.1	Overview	469
9.2	DSL (Diagnostic Session Layer)	470
9.2.1	Process Busy behavior	470
9.2.2	Update Diagnostic Session Control when timeout occurs	471
9.2.3	Process single response of ReadDataByPeriodicIdentifier	472
9.2.4	Process single event-triggered response of ResponseOnEvent	473
9.2.5	Process concurrent requests	474
9.2.6	Interface to ComManager	476
9.2.6.1	Handling in Default Session	476
9.2.6.2	Handling in Non-Default Session	477
9.2.6.3	Session transitions	477
9.2.6.4	Communication States	478
9.2.7	Receive request message and transmit negative response message	480
9.2.8	Process Service Request with paged-buffer	481
9.2.9	Process copy data in reception	483
9.2.10	Process copy data in transmission	484
9.3	DSP (Diagnostic Service Processing)	484
9.3.1	Interface DSP - DEM (service 0x19, 0x14, 0x85)	484
9.3.2	Interface special services	484
9.3.2.1	Process Diagnostic Session Control	484
9.3.2.2	Process Tester Present	485
9.3.2.3	Process Security Access	486
9.3.2.4	Process ResponseOnEvent OnDtcChange	487
9.3.2.5	Process ResponseOnEvent OnChangeOfDataIdentifier	488

9.3.2.6	Process Jump to Bootloader	489
9.3.2.7	Process IOControlRequest handling	490
10	Configuration specification	493
10.1	How to read this chapter	493
10.2	Containers and configuration parameters	493
10.2.1	Dcm	494
10.2.2	DcmConfigSet	495
10.2.2.1	DcmPageBufferCfg	495
10.2.2.2	DcmProcessingConditions	496
10.2.3	DcmDsd	499
10.2.3.1	DcmDsd	499
10.2.3.2	DcmDsdService	501
10.2.3.3	DcmDsdServiceRequestManufacturerNotification	505
10.2.3.4	DcmDsdServiceRequestSupplierNotification	506
10.2.3.5	DcmDsdServiceTable	506
10.2.3.6	DcmDsdSubService	507
10.2.4	DcmDsl	512
10.2.4.1	DcmDsl	512
10.2.4.2	DcmDslBuffer	512
10.2.4.3	DcmDslDiagResp	514
10.2.4.4	DcmDslProtocol	515
10.2.4.5	DcmDslProtocolRow	517
10.2.4.6	DcmDslConnection	524
10.2.4.7	DcmDslMainConnection	525
10.2.4.8	DcmDslProtocolRx	528
10.2.4.9	DcmDslProtocolTx	530
10.2.4.10	DcmDslPeriodicTransmission	531
10.2.4.11	DcmDslPeriodicConnection	532
10.2.5	DcmDsp	533
10.2.5.1	DcmDspReadDTCInformation	540
10.2.5.2	DcmDspReadDTCInformationUserDefinedFaultMemory	541
10.2.5.3	DcmDspAuthentication	542
10.2.5.3.1	DcmDspAuthenticationTransmitCertificate	548
10.2.5.4	DcmDspAuthenticationRow	550
10.2.5.5	DcmDspAuthenticationConnection	551
10.2.5.6	Communication Control	557
10.2.5.6.1	DcmDspComControl	557
10.2.5.6.2	DcmDspComControlAllChannel	558
10.2.5.6.3	DcmDspComControlSetting	560
10.2.5.6.4	DcmDspComControlSpecificChannel	560
10.2.5.6.5	DcmDspComControlSubNode	562
10.2.5.7	DcmDspCommonAuthorization	564
10.2.5.8	DIDs	566

10.2.5.8.1 DcmDspDid	566
10.2.5.8.2 DcmDspDidInfo	572
10.2.5.8.3 DcmDspDidRead	574
10.2.5.8.4 DcmDspDidSignal	576
10.2.5.8.5 DcmDspDidSignalCompositePool	578
10.2.5.8.6 DcmDspDidSupportInfo	579
10.2.5.8.7 DcmDspDidRange	580
10.2.5.8.8 DcmDspDidWrite	586
10.2.5.9 DcmDspControlDTCSetting	588
10.2.5.10 Data elements	590
10.2.5.10.1 DcmDspData	590
10.2.5.10.2 DcmDspDiagnosisScaling	604
10.2.5.10.3 DcmDspArgumentScaling	605
10.2.5.10.4 DcmDspAlternativeArgumentData	606
10.2.5.10.5 DcmDspTextTableMapping	607
10.2.5.10.6 DcmDspAlternativeDataInterface	609
10.2.5.10.7 DcmDspAlternativeDataType	611
10.2.5.10.8 DcmDspAlternativeDiagnosticDataElement	612
10.2.5.10.9 DcmDataElementInstance	612
10.2.5.10.10 DcmDspExternalSRDataElementClass	613
10.2.5.10.11 DcmDspPidService01ExternalSRDataElementClass	614
10.2.5.10.12 DcmSubElementInDataElementInstance	614
10.2.5.10.13 DcmSubElementInImplDataElementInstance	615
10.2.5.10.14 DcmDspDidDataSupportInfo	616
10.2.5.10.15 DcmDspDataInfo	618
10.2.5.11 DcmDspDidControl	619
10.2.5.12 DcmDspDidControlEnableMask	624
10.2.5.13 Ecu Reset	625
10.2.5.13.1 DcmDspEcuReset	625
10.2.5.13.2 DcmDspEcuResetRow	626
10.2.5.14 Memory	628
10.2.5.14.1 DcmDspMemory	628
10.2.5.14.2 DcmDspMemoryTransfer	628
10.2.5.14.3 DcmDspAddressAndLengthFormatIdentifier	629
10.2.5.14.4 DcmDspMemoryIdInfo	630
10.2.5.14.5 DcmDspMemoryTransferIdInfo	632
10.2.5.14.6 DcmDspReadMemoryRangeByLabelInfo	633
10.2.5.14.7 DcmDspReadMemoryRangeInfo	636
10.2.5.14.8 DcmDspWriteMemoryRangeByLabelInfo	639
10.2.5.14.9 DcmDspWriteMemoryRangeInfo	643
10.2.5.15 PIDs	647
10.2.5.15.1 DcmDspPid	647
10.2.5.15.2 DcmDspPidSupportInfo	651

10.2.5.15.3 DcmDspPidData	653
10.2.5.15.4 DcmDspPidService01	654
10.2.5.15.5 DcmDspPidService02	657
10.2.5.15.6 DcmDspPidDataSupportInfo	658
10.2.5.16 DcmDspRequestControl	659
10.2.5.17 DcmDspRequestFileTransfer	662
10.2.5.18 Response on Event	665
10.2.5.18.1 DcmDspRoe	665
10.2.5.19 Routines	669
10.2.5.19.1 DcmDspRoutine	669
10.2.5.19.2 DcmDspRequestRoutineResults	672
10.2.5.19.3 DcmDspRequestRoutineResultsIn	677
10.2.5.19.4 DcmDspRequestRoutineResultsInSignal	678
10.2.5.19.5 DcmDspRequestRoutineResultsInSignalCompositePool	681
10.2.5.19.6 DcmDspRequestRoutineResultsOut	682
10.2.5.19.7 DcmDspRequestRoutineResultsOutSignal	682
10.2.5.19.8 DcmDspRequestRoutineResultsOutSignalCompositePool	686
10.2.5.19.9 DcmDspStartRoutine	687
10.2.5.19.10 DcmDspStartRoutineIn	690
10.2.5.19.11 DcmDspStartRoutineInSignal	693
10.2.5.19.12 DcmDspStartRoutineInSignalCompositePool	696
10.2.5.19.13 DcmDspStartRoutineOut	697
10.2.5.19.14 DcmDspStartRoutineOutSignal	699
10.2.5.19.15 DcmDspStopRoutine	702
10.2.5.19.16 DcmDspStopRoutineIn	706
10.2.5.19.17 DcmDspStartRoutineOutSignalCompositePool	709
10.2.5.19.18 DcmDspStopRoutineInSignal	709
10.2.5.19.19 DcmDspStopRoutineInSignalCompositePool	712
10.2.5.19.20 DcmDspStopRoutineOut	713
10.2.5.19.21 DcmDspStopRoutineOutSignal	715
10.2.5.19.22 DcmDspStopRoutineOutSignalCompositePool	718
10.2.5.20 Session Security and Modes	719
10.2.5.20.1 DcmDspSecurity	719
10.2.5.20.2 DcmDspSecurityRow	722
10.2.5.20.3 DcmDspSession	728
10.2.5.20.4 DcmDspSessionRow	729
10.2.5.20.5 DcmModeCondition	732
10.2.5.20.6 DcmSwcDataElementValue	735
10.2.5.20.7 DcmSwcDataElementPrimitive	735
10.2.5.20.8 DcmSwcDataElementArray	736
10.2.5.20.9 DcmSwcDataElementArrayElement	737
10.2.5.20.10 DcmModeRule	738
10.2.5.21 DcmDspVehInfo	740

10.2.5.22 DcmDspVehInfoData	742
10.2.5.23 DcmDspPeriodicTransmission	745
10.2.5.24 DcmDspClearDTC	750
10.2.6 DcmGeneral	752
10.2.6.1 DcmSecurityEventRefs	759
10.2.6.2 DcmSecureCoding	774
10.3 Protocol Configuration Example	776
10.4 Published Information	777
A Change history of AUTOSAR traceable items	778
A.1 Traceable item history of this document according to AUTOSAR Release R25-11	778
A.1.1 Added Specification Items in R25-11	778
A.1.2 Changed Specification Items in R25-11	778
A.1.3 Deleted Specification Items in R25-11	781
A.1.4 Added Constraints in R25-11	781
A.1.5 Changed Constraints in R25-11	782
A.1.6 Deleted Constraints in R25-11	782
A.2 Traceable item history of this document according to AUTOSAR Release R24-11	783
A.2.1 Added Specification Items in R24-11	783
A.2.2 Changed Specification Items in R24-11	786
A.2.3 Deleted Specification Items in R24-11	788
A.2.4 Added Constraints in R24-11	789
A.2.5 Changed Constraints in R24-11	789
A.2.6 Deleted Constraints in R24-11	790
A.3 Traceable item history of this document according to AUTOSAR Release R23-11	790
A.3.1 Added Specification Items in R23-11	790
A.3.2 Changed Specification Items in R23-11	792
A.3.3 Deleted Specification Items in R23-11	800
A.3.4 Added Constraints in R23-11	800
A.3.5 Changed Constraints in R23-11	801
A.3.6 Deleted Constraints in R23-11	801
A.4 Not applicable requirements	802

1 Introduction and functional overview

The **Dcm** SWS describes the functionality, the API, and the configuration of the AUTOSAR Basic Software module **Dcm** (Diagnostic Communication Manager). The **Dcm** module provides a common **API** for diagnostic services. The functionality of the **Dcm** module is used by external diagnostic tools during the development, manufacturing or service.

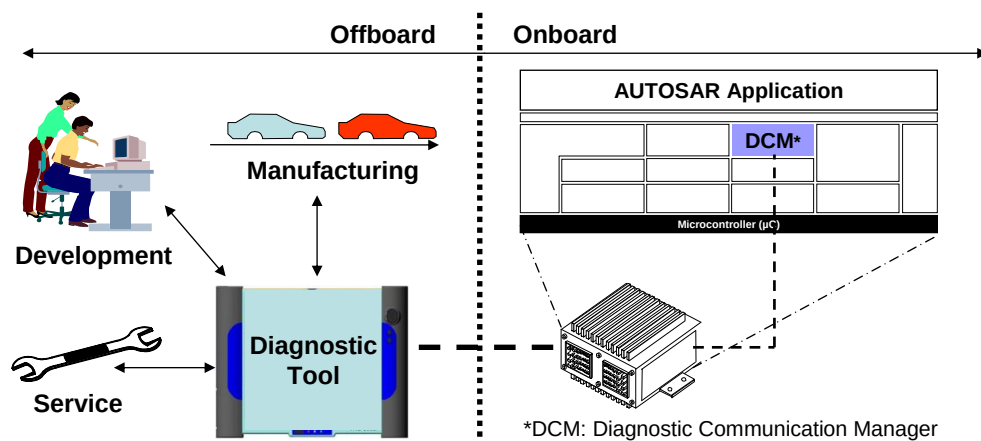


Figure 1.1: Overview of the communication between the external diagnostic tools and the onboard AUTOSAR Application

The **Dcm** module ensures diagnostic data flow and manages the diagnostic states, especially diagnostic sessions and security states. Furthermore, the **Dcm** module checks if the diagnostic service request is supported and if the service may be executed in the current session according to the diagnostic states. The **Dcm** module provides the OSI-Layers 5 to 7 of Table 1: Diagnostic protocols and OSI-Layer.

OSI-Layer	Protocols					
7	UDS-Protocol - ISO14229-1 [1] - SAE J1979-2	-	-	-	-	Legislated OBD - ISO15031-5 [2] - SAE J1979
6	ISO 14229-1:2013 [1]	SAE J1979DA, SAE J2012DA	-	-	-	SAE J1979DA, SAE J2012DA
5	ISO15765-3	-	-	-	-	ISO14229-2 [3]
4	ISO15765-2	ISO 15765-4	-	-	ISO 13400-2	ISO 15765-4
3	ISO15765-2	ISO 15765-4	-	-	ISO 13400-2	ISO14229-2 [3]
2	CAN-Protocol	ISO 15765-4	LIN-Protocol	FlexRay	ISO 13400-3	ISO14229-2 [3]

OSI-Layer	Protocols					
1	CAN-Protocol	ISO 15765-4	LIN-Protocol	FlexRay	ISO 13400-3	ISO14229-2 [3]

Table 1.1: Diagnostic protocols and OSI-Layers

At OSI-level 7, the **Dcm** module provides an extensive set of ISO14229-1 [1] services. In addition, the **Dcm** module provides mechanisms to support the **OBD** services \$01 - \$0A defined in documents [4, SAE J1979] and [2, ISO 15031-5]. With these services, Autosar **OBD** functionality is capable of meeting all light duty **OBD** regulations worldwide (California OBDII, EOBD, Japan OBD, and all others). At OSI-level 5, the **Dcm** module handles the network-independent sections of the following specifications:

- ISO15765-3 [5]: Implementation of unified diagnostic services (UDS on CAN)
- ISO15765-4 [6]: Requirements for emission-related systems, Chapter 5 "Session Layer"

In the AUTOSAR Architecture the Diagnostic Communication Manager is located in the Communication Services (Service Layer).

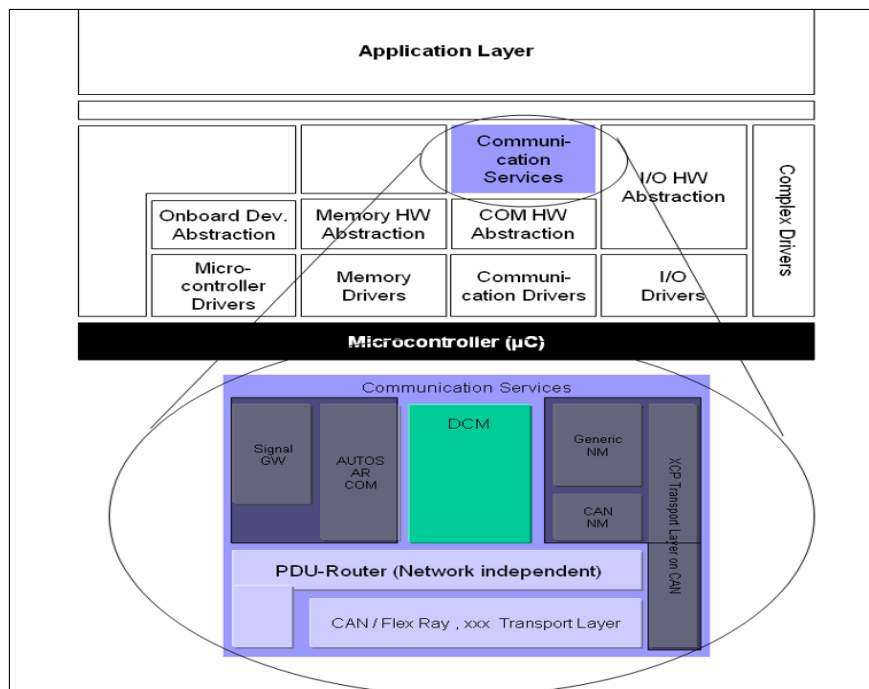


Figure 1.2: Position of the **Dcm module in AUTOSAR Architecture**

The **Dcm** module is network-independent. All network-specific functionality (the specifics of networks like CAN, LIN, FlexRay or MOST) is handled outside of the **Dcm** module. The PDU Router (PduR) module provides a network-independent interface to the **Dcm** module. The **Dcm** module receives a diagnostic message from the PduR module. The **Dcm** module processes and checks internally the diagnostic message. As part of processing the requested diagnostic service, the **Dcm** will interact with other

BSW modules or with SW-Components (through the RTE) to obtain requested data or to execute requested commands. This processing is very service-specific. Typically, the [Dcm](#) will assemble the gathered information and send a message back through the PduR module.

2 Acronyms and Abbreviations

The glossary below includes acronyms and abbreviations relevant to the Dcm module that are not included in the [7, AUTOSAR Glossary].

Abbreviation / Acronym:	Description:
AddressAndLengthFormat Identifier	Defines the number of bytes used for the memoryAddress and memorySize parameter in the request messages.
Application Layer	The Application Layer is placed above the RTE. Within the Application Layer the AUTOSAR Software-Components are placed.
Atomic Sender/Receiver interface	An atomic sender-receiver interface can be used to group DID data elements into one record data element prototype. All data elements can be read or write having a single read or write operation.
Channel	A link at which a data transfer can take place. If there is more than one Channel, there is normally some kind of ID assigned to the Channel.
Concurrent TesterPresent	A UDS service TesterPresent with set suppressPosRspMsgIndicationBit (3E 80) which is received with functional addressing. It is also known as keep-alive logic..
Diagnostic Channel	A link at which a data transfer between a diagnostic tool and an ECU can take place. Example: An ECU is connected via CAN and the diagnostic channel has an assigned CAN-ID. Diagnostic channels connected to other bus-systems such as MOST, FlexRay, LIN, etc. are also possible.
External Diagnostic Tool	A device which is NOT permanently connected to the vehicle communication network. This External Diagnostic Tool can be connected to the vehicle for various purposes, as e.g. for: • development • manufacturing • service (in a garage) Example External Diagnostic Tools are: • a diagnostic tester • an OBD scan tool The External Diagnostic Tool is to be connected by a mechanic to gather information from "inside" the car.
Freeze Frame	A set of the vehicle/system operation conditions at a specific time.
Functional Addressing	The diagnostic communication model where a group or all nodes of a specific communication network receive a message from one sending node (1-n communication). This model is also referred to as 'broadcast' or 'multicast'. OBD communication will always be done in the Functional Addressing mode.

Abbreviation / Acronym:	Description:
Internal Diagnostic Tool	A device/ECU which is connected to the vehicle communication network. The Internal Diagnostic Tool can be used for: • advanced event tracking • advanced analysis • for service The behavior of the Internal Diagnostic Tool can be the same as of an External Diagnostic Tool. The notion of "Internal Diagnostic Tool" does not imply that it is included in each ECU as an AUTOSAR Software-Component.
Physical Addressing	The diagnostic communication model where a node of a specific communication network receives a message from one sending node (1-1 communication). This model is also referred to as 'unicast'.
UDS Service	this refers to a UDS Service as defined in ISO14229-1 [1].
OBD Scan tool	See definition External Diagnostic Tool.
OBD Service	This refers to an OBD Service as defined in ISO15031-5 [2].
OBD UDS DTC separation	The feature will report a different 3 byte DTC number for SAE J1979-2 based UDS communication than for none J1979-2 UDS communication.
periodic transmission rate	Time interval value that defines the time between two calls of a periodic data identifier transmission. The value is configuration specific and there are separate values for fast, medium and slow periodic data transmission. The configured value is always an integer multiple of the Dcm main task time.

Terms:	Description:
API	Application Programming Interface
CAN	Controller Area Network
CEMR	ControlEnableMaskRecord
Dcm	Diagnostic Communication Manager
Dem	Diagnostic Event Manager
Det	Default Error Tracer
DID	Data Identifier
DSD	Diagnostic Service Dispatcher (submodule of the Dcm module)
DSL	Diagnostic Session Layer (submodule of the Dcm module)
DSP	Diagnostic Service Processing (submodule of the Dcm module)
DTC	Diagnostic Trouble Codes
ID	Identifier
LIN	Local Interconnect Network
MCU	Micro-Controller Unit
MOST	Media Orientated System Transport
NRC	Negative Response Code
OBD	On-Board Diagnosis
OSI	Open Systems Interconnection
PDID	Periodic Data Identifier, periodically send by the Dcm after a request of ReadDataByPeriodicIdentifier
PDU	Protocol Data Unit
PID	Parameter Identifier
RCRRP	Response correctly received - response pending
RID	Routine Identifier
ROE	ResponseOnEvent

Terms:	Description:
RTE	Runtime Environment
SAP	Service Access Point
SDU	Service Data Unit
SID	Service Identifier
SW-C	Software-Component
TP	Transport Protocol
UDS	Unified Diagnostic Services
Xxx_	Placeholder for an API provider
SPRMIB	suppressPosRspMsgIndicationBit

2.1 Typographical Conventions

This document uses the following typographical conventions:

- see configuration parameter `myConfigurationParameter`: this is a reference to a configuration parameter which can be found in [Chapter 10](#).
- `myFunction()`: this is a function provided or required by the module as defined in [Chapter 8](#)

3 Related documentation

3.1 Input documents & related standards and norms

- [1] ISO 14229-1(2013) – Unified diagnostic services (UDS) – Part 1: Application layer (Release 2013-03)
<https://www.iso.org>
- [2] ISO 15031-5 – Road vehicles – Communication between vehicle and external equipment for emission-related diagnostic – Part 5: Emission-related diagnostic services.
<https://www.iso.org/standard/66368.html>
- [3] ISO 14229-2(2021) – Unified diagnostic services (UDS) - Part 2: Session layer services (Release 2021-10)
<https://www.iso.org>
- [4] SAE J1979 – E/E Diagnostic Test Modes
- [5] ISO 15765-3 – Diagnostics on controller area network (CAN) – Part 3: Implementation of unified diagnostic services (UDS on CAN) (Release 2004 10-06)
- [6] ISO 15765-4 – Diagnostics on controller area network (CAN) – Part 4: Requirements for emission-related systems (Release 2005 01-04)
- [7] Glossary
AUTOSAR_FO_TR_Glossary
- [8] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral
- [9] ISO 14229-1(2020) – Unified diagnostic services (UDS) – Part 1: Application layer (Release 2020-02)
<https://www.iso.org>
- [10] Requirements on Diagnostics
AUTOSAR_FO_RS_Diagnostics
- [11] Requirements on Intrusion Detection System
AUTOSAR_FO_RS_IntrusionDetectionSystem
- [12] General Requirements on Basic Software Modules
AUTOSAR_CP_RS_BSWGeneral
- [13] ISO 17356-3: Road vehicles – Open interface for embedded automotive applications – Part 3: OSEK/VDX Operating System (OS)
- [14] Specification of PDU Router
AUTOSAR_CP_SWS_PDURouter
- [15] ISO 15765-2 – Road vehicles – Diagnostics on Controller Area Networks (CAN) – Part2: Network layer services

- [16] ISO 15031-6 – Road vehicles – Communication between vehicle and external equipment for emission-related diagnostic – Part 6: Diagnostic trouble code definitions
<https://www.iso.org>
- [17] Specification of Crypto Service Manager
AUTOSAR_CP_SWS_CryptoServiceManager
- [18] Specification of Key Manager
AUTOSAR_CP_SWS_KeyManager
- [19] Specification of I/O Hardware Abstraction
AUTOSAR_CP_SWS_IOHardwareAbstraction
- [20] Specification of Diagnostic Event Manager
AUTOSAR_CP_SWS_DiagnosticEventManager

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [8, SWS BSW General] , which is also valid for Diagnostic Communication Manager.

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

4 Constraints and assumptions

4.1 Limitations

The following limitations apply when using the `Dcm` module:

- `UDS` Service AccessTimingParameter (0x83) is not supported by the ISO standards in `CAN` and `LIN`. Also it is not planned to support this service with `FlexRay`. Therefore no support for this service is planned.
- Subfunction `onTimerInterrupt` of Service `ResponseOnEvent` is not supported in the current release.
- `UDS` Service SecuredDataTransmission (0x84) is not supported in the current release.
- The `Dcm` SWS does not cover any SAE J1939 related diagnostic requirements.
- Due to DEM limitation, the diagnostic service \$19 05 is limited to the OBD legislative freeze frame.
- Management of `IOControl` service without `InputOutputControlParameter` in request and response is not supported
- The length of `controlState` parameter in `IOControl` request and response has to be of same size (due to the one configuration parameter `DcmDspDataByteSize`)
- Same layout of a `DID` which is used in `RDBI`, `WDBI` or `IOCBI` services
- The user optional parameter `DTCSettingControlOptionRecord` in the `ControlDTCSetting` request is only supported if it corresponds to a `groupOfAllDTCs` (0xFFFF) value. In other cases it has to be managed in a vendor specific implementation.
- Only the `ControlDTCSetting` sub-functions 0x01 and 0x02 are supported.
- The handling of infrastructure errors reported by the `RTE` during `DCM/DEM` <-> `SW-C` interactions is missing from the SWS and might have to be taken into account by implementers if they need it.
- The `Dcm` does not support `DLT` for `ROE`
- The `ROE` `ServiceToRespondTo` does not support `PageBuffering`
- `DID` range feature cannot be applied for services `DynamicallyDefineDataIdentifier`, `ReadDataByPeriodicIdentifier` and `InputOutputControlById`
- AUTOSAR `Dcm` is not intended to be used in the bootloader
- `PeriodicTransmission` is not possible on `FlexRay`, as ISO 14229-4 demands header information (address information (source and target address) and `FPL` (Frame Payload length)). This information can't be filled with the specified concept of `IF` interface.

- The specification of the transformer for intra ecu communication between the [Dcm](#) module and the `NvBlockSoftwareComponentType` is not standardized in the current AUTOSAR release. For this scenario custom transformers implemented by a complex driver can be used. To elaborate on this the responsible stakeholder (usually the OEM) needs to specify the custom transformer from a behavioral point of view in a separate document (this might include definition of byte-ordering or alignment). If there is the necessity to define transformer specific attributes in the model this can be done using special data groups in `UserDefinedTransformationDescription` and `UserDefinedTransformationSignalProps`. For the configuration of this scenario, a `DataPrototypeMapping` shall exist for the affected `SenderReceiverInterfaces` of the [Dcm](#) module and the `NvBlockSoftwareComponentType` which refers to a `DataTransformation` in the role `firstToSecondDataTransformation`. This `DataTransformation` shall reference exactly one `TransformationTechnology` in the role `transformerChain` with the `transformerClass` attribute set to "serializer" and may compose a `UserDefinedTransformationDescription` in the role `transformationDescription`.
- For UDS service 0x29, the [Dcm](#) supports only the sub-functions for PKI. Authentication via challenge response is not supported.
- For UDS service 0x29, secure diagnostic communication with Diffie-Hellmann key exchange is not supported.
- For UDS service 0x29 the [Dcm](#) does not support [NRC](#) 'Certificate verification failed - Invalid Content'.
- The [Dcm](#) supports only selected subset of functionality according to ISO 14229-1:2020 [9]. Unless explicitly stated the [Dcm](#) follows the ISO 14229-1:2013 [1].
- Autosar PduR architecture allows the use of streaming protocols. As diagnostic protocols do not use streaming protocols, the [Dcm](#) does not support it either and therefore the [Dcm_StartOfReception](#) with `TpSduLength` equal to zero is not supported.
- The [Dcm](#) has a different order of [NRCs](#) for UDS service 0x36. The ISO14229-1 [1] defined [NRC](#) order for that service would require further APIs that would make the overall service processing more complicated. Details can be found in chapter 7.4.2.21.
- The [Dcm](#) does not support the behavior described in the ISO 14229-1:2020 [9] chapter 10.4. that after a powerup a single false access attempt in the previous power cycle already starts the security delay time. ([[SWS_Dcm_01355](#)] is followed instead.) .

4.2 Applicability to car domains

The [Dcm](#) module can be used for all car domains.

4.3 Applicability to emission-related environments (OBD)

This [Dcm](#) SWS is intended to fulfill the emission related requirements given by legislator. However, the supplier of the emission related system is responsible to fulfill the [OBD](#) requirements. Certain requirements cannot be fulfilled by the [Dcm](#) module by itself, but need to be considered at the level of the entire ECU or system. Example: During the integration of the [Dcm](#) module within the system, the timing requirements (50ms response time) must be fulfilled.

For WWH-OBD only the FunctionalGroupIdentifier 0x33 is currently supported.

5 Dependencies to other modules

The AUTOSAR Diagnostic Communication Manager (DCM) has interfaces and dependencies to the following Basic Software modules and SW-Cs:

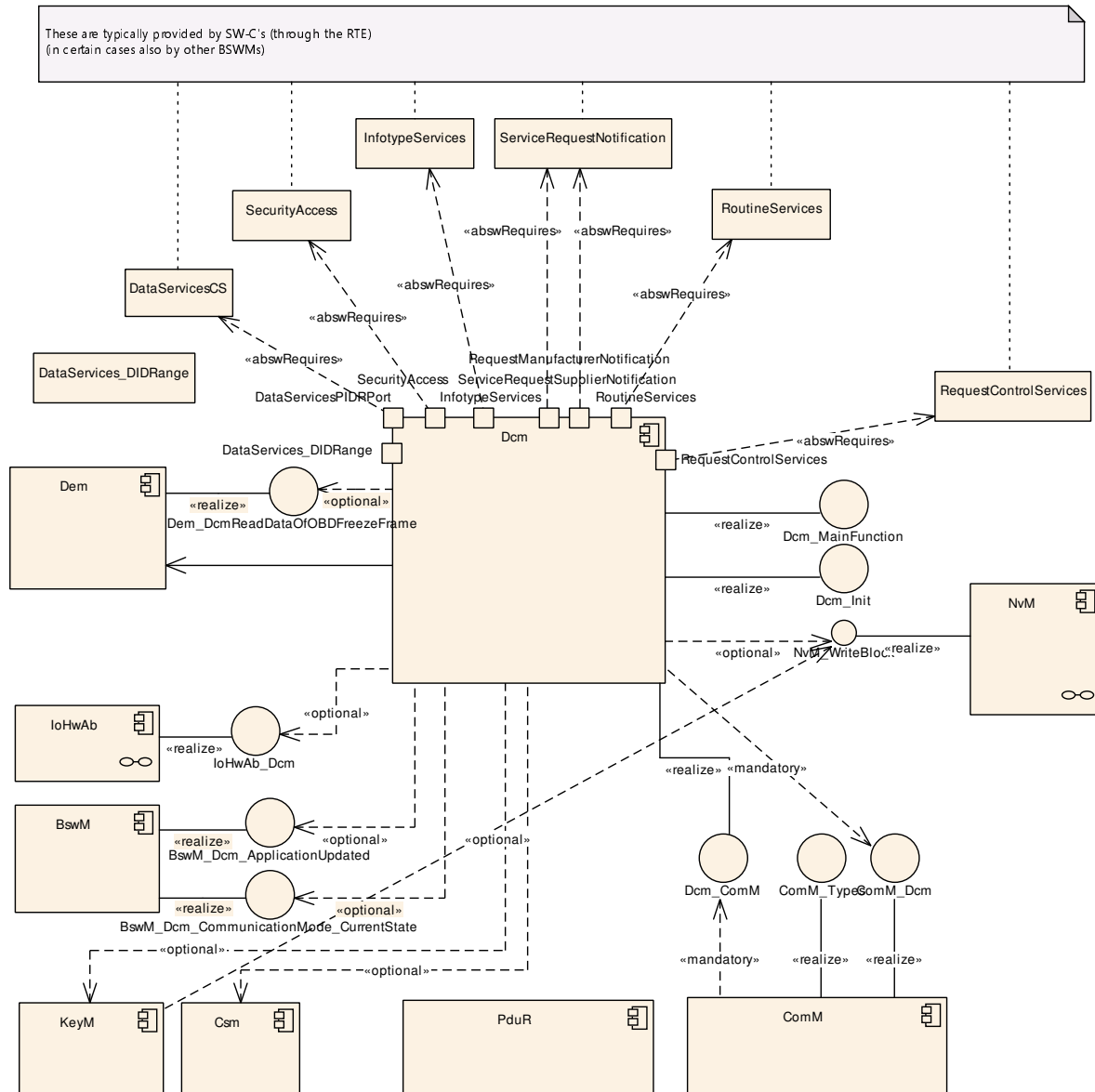


Figure 5.1: Interaction of the *Dcm* with other modules

- **Diagnostic Event Manager (DEM):** The DEM module provides function to retrieve all information related to fault memory such that the *Dcm* module is able to respond to tester requests by reading data from the fault memory.
- **Protocol Data Unit Router (PduR module):** The PduR module provides functions to transmit and receive diagnostic data. Proper operation of the *Dcm* module presumes that the PduR interface supports all service primitives defined for the Service Access Point (SAP) between diagnostic application layer and underlying transport layer (see ISO14229-1 [1], chapter 5 Application layer services).

- **Communication Manager (ComM):** The ComM module provides functions such that the [Dcm](#) module can indicate the states "active" and "inactive" for diagnostic communication. The [Dcm](#) module provides functionality to handle the communication requirements "Full-/ Silent-/ No-Communication". Additionally, the [Dcm](#) module provides the functionality to enable and disable Diagnostic Communication if requested by the ComM module.
- **SW-C and RTE:** The [Dcm](#) module has the capability to analyze the received diagnostic request data stream and handles all functionalities related to diagnostic communication such as protocol handling and timing. Based on the analysis of the request data stream the [Dcm](#) module assembles the response data stream and delegates routines or IO-Control executions to SW-Cs .If any of the data elements or functional states cannot be provided by the [Dcm](#) module itself the [Dcm](#) requests data or functional states from SW-Cs via port-interfaces or from other BSW modules through direct function-calls.
- **BswM:** The [Dcm](#) notifies the BswM that the application was updated if the initialization of the [Dcm](#) is the consequence of a jump from the bootloader . The [Dcm](#) also indicates to the BswM a communication mode change.
- **Crypto Service Manager (Csm):** The crypto service module provides a wide range of cryptographic algorithms. The Csm is used for authentication calculation.
- **Key Manager (KeyM):** The key manager module provides support for certificate handling and APIs to realize authenticated diagnostics via certificates.

6 Requirements Tracing

The following tables reference the requirements specified in [10], [11] as well as [12] and links to the fulfillment of these. Please note that if column "Satisfied by" is empty for a specific requirement this means that this requirement is not fulfilled by this document.

Requirement	Description	Satisfied by
[RS_Diag_04003]	Network independent design	[SWS_Dcm_00030]
[RS_Diag_04005]	Manage Security Access level handling	[SWS_Dcm_00020] [SWS_Dcm_00033] [SWS_Dcm_00252] [SWS_Dcm_00338] [SWS_Dcm_01397] [SWS_Dcm_01535] [SWS_Dcm_CONSTR_06083]
[RS_Diag_04006]	Manage session handling	[SWS_Dcm_00022] [SWS_Dcm_00250] [SWS_Dcm_00339] [SWS_Dcm_01373] [SWS_Dcm_01374] [SWS_Dcm_01375] [SWS_Dcm_01376] [SWS_Dcm_01377] [SWS_Dcm_01378] [SWS_Dcm_01535]
[RS_Diag_04011]	Provide diagnostic state information to applications	[SWS_Dcm_00338] [SWS_Dcm_00339] [SWS_Dcm_01321] [SWS_Dcm_01322]
[RS_Diag_04015]	Timing handling according to ISO14229-2	[SWS_Dcm_00027] [SWS_Dcm_00030] [SWS_Dcm_00143] [SWS_Dcm_01679]
[RS_Diag_04016]	Support "Busy handling" by sending a negative response 0x78	[SWS_Dcm_00024]
[RS_Diag_04020]	Suppress responses to diagnostic tool requests	[SWS_Dcm_00001] [SWS_Dcm_00200]
[RS_Diag_04021]	Handling of different diagnostic sessions in parallel	[SWS_Dcm_00015] [SWS_Dcm_00079] [SWS_Dcm_00460] [SWS_Dcm_00575] [SWS_Dcm_00576] [SWS_Dcm_00727] [SWS_Dcm_01046] [SWS_Dcm_01602] [SWS_Dcm_01603] [SWS_Dcm_01604] [SWS_Dcm_01605]
[RS_Diag_04033]	Support the upload/download services for reading/writing data in an ECU in an extended and manufacturer specific diagnostic session	[SWS_Dcm_00496] [SWS_Dcm_00499] [SWS_Dcm_00502] [SWS_Dcm_00503] [SWS_Dcm_00504] [SWS_Dcm_00505] [SWS_Dcm_01417] [SWS_Dcm_01418] [SWS_Dcm_01419] [SWS_Dcm_01420] [SWS_Dcm_01421] [SWS_Dcm_01422] [SWS_Dcm_01591] [SWS_Dcm_01592] [SWS_Dcm_01593] [SWS_Dcm_01594] [SWS_Dcm_01595] [SWS_Dcm_01596] [SWS_Dcm_01598]
[RS_Diag_04058]	Ability to access different event memories	[SWS_Dcm_00004] [SWS_Dcm_00005] [SWS_Dcm_00077] [SWS_Dcm_00279] [SWS_Dcm_00293] [SWS_Dcm_00295] [SWS_Dcm_00378] [SWS_Dcm_00388] [SWS_Dcm_00389] [SWS_Dcm_00393] [SWS_Dcm_00465] [SWS_Dcm_01263]
[RS_Diag_04067]	Provide the diagnostic status information according to ISO 14229-1	[SWS_Dcm_00293] [SWS_Dcm_00378]
[RS_Diag_04098]	Interact with standard bootloader	[SWS_Dcm_00532] [SWS_Dcm_00535] [SWS_Dcm_00536] [SWS_Dcm_00592] [SWS_Dcm_00654] [SWS_Dcm_00767] [SWS_Dcm_01163] [SWS_Dcm_01177] [SWS_Dcm_01423] [SWS_Dcm_01424] [SWS_Dcm_01425] [SWS_Dcm_CONSTR_06080]





Requirement	Description	Satisfied by
[RS_Diag_04115]	The optional parameter DTCSettingControlOptionRecord as part of UDS service ControlDTCSetting shall be limited to GroupOfDTC	[SWS_Dcm_00406] [SWS_Dcm_01063]
[RS_Diag_04119]	Handle the execution of diagnostic services according to the assigned diagnostic session	[SWS_Dcm_00628] [SWS_Dcm_00858] [SWS_Dcm_00859] [SWS_Dcm_01435]
[RS_Diag_04135]	Support UDS service \$38 (Request FileTransfer)	[SWS_Dcm_01083] [SWS_Dcm_01088] [SWS_Dcm_01090] [SWS_Dcm_01091] [SWS_Dcm_01446] [SWS_Dcm_01447] [SWS_Dcm_01448] [SWS_Dcm_01449] [SWS_Dcm_01450] [SWS_Dcm_01451] [SWS_Dcm_01452] [SWS_Dcm_01453] [SWS_Dcm_01454] [SWS_Dcm_01455] [SWS_Dcm_01456] [SWS_Dcm_01457] [SWS_Dcm_01458] [SWS_Dcm_91064] [SWS_Dcm_91065] [SWS_Dcm_91066] [SWS_Dcm_91078] [SWS_Dcm_91079] [SWS_Dcm_91080] [SWS_Dcm_91081] [SWS_Dcm_91082] [SWS_Dcm_91083] [SWS_Dcm_91084] [SWS_Dcm_91085]
[RS_Diag_04147]	Communication with the transport layers to receive and send diagnostic data	[SWS_Dcm_00642] [SWS_Dcm_01186]
[RS_Diag_04153]	Support generic connections	[SWS_Dcm_00849] [SWS_Dcm_01347] [SWS_Dcm_01348]
[RS_Diag_04159]	Control of DTC storage	[SWS_Dcm_00249] [SWS_Dcm_01399]
[RS_Diag_04160]	ResponseOnEvent according to ISO 14229-1	[SWS_Dcm_00135] [SWS_Dcm_00894] [SWS_Dcm_01410] [SWS_Dcm_01625] [SWS_Dcm_01626] [SWS_Dcm_01627] [SWS_Dcm_01628] [SWS_Dcm_01629] [SWS_Dcm_01630] [SWS_Dcm_01631] [SWS_Dcm_01632]
[RS_Diag_04162]	Parallel fault memory access	[SWS_Dcm_01369] [SWS_Dcm_01370] [SWS_Dcm_01371] [SWS_Dcm_01372]
[RS_Diag_04163]	Parallel OBD and UDS processing	[SWS_Dcm_01367]
[RS_Diag_04180]	Process all UDS Services related to diagnostic fault memory of ISO 14229-1 internally	[SWS_Dcm_01623]
[RS_Diag_04196]	UDS Service handling for all diagnostic services defined in ISO 14229-2	[SWS_Dcm_01622] [SWS_Dcm_01671] [SWS_Dcm_01672] [SWS_Dcm_01673]
[RS_Diag_04203]	Common checks on all supported UDS Services Requests	[SWS_Dcm_01622]
[RS_Diag_04205]	Support of SnapshotRecords	[SWS_Dcm_01623] [SWS_Dcm_01624]
[RS_Diag_04209]	Pseudo parallel client interaction according to ISO	[SWS_Dcm_01674] [SWS_Dcm_01675] [SWS_Dcm_01676] [SWS_Dcm_01677] [SWS_Dcm_01678]





Requirement	Description	Satisfied by
[RS_Diag_04215]	Support of UDS service ReadDataByPeriodicIdentifier (0x2A)	[SWS_Dcm_00721] [SWS_Dcm_00722] [SWS_Dcm_00820] [SWS_Dcm_00843] [SWS_Dcm_01096] [SWS_Dcm_01097] [SWS_Dcm_01098] [SWS_Dcm_01099] [SWS_Dcm_01101] [SWS_Dcm_01103] [SWS_Dcm_01104] [SWS_Dcm_01105] [SWS_Dcm_01106] [SWS_Dcm_01111] [SWS_Dcm_01112] [SWS_Dcm_01115] [SWS_Dcm_01117] [SWS_Dcm_01118] [SWS_Dcm_01568] [SWS_Dcm_01569] [SWS_Dcm_01570] [SWS_Dcm_01571] [SWS_Dcm_01572] [SWS_Dcm_01573] [SWS_Dcm_01574] [SWS_Dcm_01575] [SWS_Dcm_01576] [SWS_Dcm_01577] [SWS_Dcm_01578] [SWS_Dcm_01613]
[RS_Diag_04218]	Support of UDS service 0x2F Input OutputControlByIdentifier	[SWS_Dcm_00256] [SWS_Dcm_00396] [SWS_Dcm_00397] [SWS_Dcm_00398] [SWS_Dcm_00399] [SWS_Dcm_00563] [SWS_Dcm_00564] [SWS_Dcm_00565] [SWS_Dcm_00579] [SWS_Dcm_00580] [SWS_Dcm_00581] [SWS_Dcm_00640] [SWS_Dcm_00680] [SWS_Dcm_00682] [SWS_Dcm_01273] [SWS_Dcm_01274] [SWS_Dcm_01275] [SWS_Dcm_01277] [SWS_Dcm_01313] [SWS_Dcm_01434] [SWS_Dcm_01436] [SWS_Dcm_01437] [SWS_Dcm_01438] [SWS_Dcm_01439] [SWS_Dcm_01440] [SWS_Dcm_01441] [SWS_Dcm_01554] [SWS_Dcm_01600] [SWS_Dcm_91057] [SWS_Dcm_91058] [SWS_Dcm_91059] [SWS_Dcm_91060] [SWS_Dcm_91061] [SWS_Dcm_CONSTR_06084] [SWS_Dcm_CONSTR_06085] [SWS_Dcm_CONSTR_06086]
[RS_Diag_04224]	Support the UDS service 0x31 (RoutineControl) according to ISO 14229-1	[SWS_Dcm_01135] [SWS_Dcm_01442] [SWS_Dcm_01443] [SWS_Dcm_01580] [SWS_Dcm_01581]
[RS_Diag_04230]	Support of UDS service 0x29 (Authentication)	[SWS_Dcm_01459] [SWS_Dcm_01460] [SWS_Dcm_01461] [SWS_Dcm_01463] [SWS_Dcm_01464] [SWS_Dcm_01465] [SWS_Dcm_01466] [SWS_Dcm_01467] [SWS_Dcm_01468] [SWS_Dcm_01469] [SWS_Dcm_01477] [SWS_Dcm_01478] [SWS_Dcm_01479] [SWS_Dcm_01480] [SWS_Dcm_01481] [SWS_Dcm_01482] [SWS_Dcm_01483] [SWS_Dcm_01484] [SWS_Dcm_01485] [SWS_Dcm_01486] [SWS_Dcm_01487] [SWS_Dcm_01488] [SWS_Dcm_01489] [SWS_Dcm_01493] [SWS_Dcm_01503] [SWS_Dcm_01504] [SWS_Dcm_01506] [SWS_Dcm_01507] [SWS_Dcm_01508] [SWS_Dcm_01509] [SWS_Dcm_01510] [SWS_Dcm_01511] [SWS_Dcm_01512] [SWS_Dcm_01513] [SWS_Dcm_01515] [SWS_Dcm_01518] [SWS_Dcm_01528] [SWS_Dcm_01529] [SWS_Dcm_01530] [SWS_Dcm_01532] [SWS_Dcm_01533] [SWS_Dcm_01534] [SWS_Dcm_01535] [SWS_Dcm_01536] [SWS_Dcm_01537] [SWS_Dcm_01538] [SWS_Dcm_01542] [SWS_Dcm_01544]





Requirement	Description	Satisfied by
		△ [SWS_Dcm_01551] [SWS_Dcm_01558] [SWS_Dcm_01559] [SWS_Dcm_01560] [SWS_Dcm_01561] [SWS_Dcm_01565] [SWS_Dcm_01579] [SWS_Dcm_01588] [SWS_Dcm_CONSTR_06094] [SWS_Dcm_CONSTR_06096]
[RS_Diag_04232]	Access rights in client certificates	[SWS_Dcm_01516] [SWS_Dcm_CONSTR_06087] [SWS_Dcm_CONSTR_06089]
[RS_Diag_04233]	Access granularity of diagnostic services	[SWS_Dcm_01496] [SWS_Dcm_01514] [SWS_Dcm_01522] [SWS_Dcm_01523] [SWS_Dcm_01524] [SWS_Dcm_01525] [SWS_Dcm_01526] [SWS_Dcm_01527] [SWS_Dcm_01539] [SWS_Dcm_01540] [SWS_Dcm_01541] [SWS_Dcm_01545] [SWS_Dcm_01546] [SWS_Dcm_01547] [SWS_Dcm_01548] [SWS_Dcm_01549] [SWS_Dcm_01552] [SWS_Dcm_01553] [SWS_Dcm_01555] [SWS_Dcm_01556] [SWS_Dcm_01557] [SWS_Dcm_01562] [SWS_Dcm_CONSTR_06088]
[RS_Diag_04234]	Binary compatibility of white list for individual access	[SWS_Dcm_01521] [SWS_Dcm_01522] [SWS_Dcm_01523] [SWS_Dcm_01524] [SWS_Dcm_01525] [SWS_Dcm_01526] [SWS_Dcm_01527]
[RS_Diag_04235]	Client certificate validity	[SWS_Dcm_01470] [SWS_Dcm_01471] [SWS_Dcm_01472] [SWS_Dcm_01473] [SWS_Dcm_01474] [SWS_Dcm_01475] [SWS_Dcm_01476]
[RS_Diag_04243]	Update of constant parameters through diagnostics	[SWS_Dcm_01582] [SWS_Dcm_01583] [SWS_Dcm_01584] [SWS_Dcm_01585] [SWS_Dcm_01586] [SWS_Dcm_01587] [SWS_Dcm_91088]
[RS_Diag_04248]	Support of session control service	[SWS_Dcm_00311]
[RS_Diag_04249]	Support of session layer service	[SWS_Dcm_00024] [SWS_Dcm_00027] [SWS_Dcm_00111] [SWS_Dcm_00114] [SWS_Dcm_00115] [SWS_Dcm_00117] [SWS_Dcm_00143] [SWS_Dcm_00654] [SWS_Dcm_00703] [SWS_Dcm_00837] [SWS_Dcm_00839] [SWS_Dcm_01073] [SWS_Dcm_01425] [SWS_Dcm_01666] [SWS_Dcm_01667] [SWS_Dcm_01668] [SWS_Dcm_01669] [SWS_Dcm_01670] [SWS_Dcm_01679]
[RS_Diag_04250]	Support subfunction 0x1A and 0x56 of UDS service 0x19	[SWS_Dcm_01607] [SWS_Dcm_01608] [SWS_Dcm_01609] [SWS_Dcm_01610] [SWS_Dcm_01611] [SWS_Dcm_01612] [SWS_Dcm_01640]
[RS_Diag_04252]	Support of SAE J1979-2	[SWS_Dcm_00484]
[RS_Diag_04253]	Support separated DTCs for UDS and OBD based on J1979-2	[SWS_Dcm_01618] [SWS_Dcm_01619]
[RS_Diag_04254]	Independent CP Software Cluster development	[SWS_Dcm_91090] [SWS_Dcm_91091] [SWS_Dcm_91092] [SWS_Dcm_91093] [SWS_Dcm_91094] [SWS_Dcm_91095] [SWS_Dcm_91098] [SWS_Dcm_91099] [SWS_Dcm_91100] [SWS_Dcm_91101] [SWS_Dcm_91102] [SWS_Dcm_91103] [SWS_Dcm_91104] [SWS_Dcm_91105] [SWS_Dcm_91106] [SWS_Dcm_91107]





Requirement	Description	Satisfied by
[RS_Diag_04271]	Support of Transmit Certificate	[SWS_Dcm_01682] [SWS_Dcm_01683]
[RS_Diag_04272]	Support of secure data coding	[SWS_Dcm_01684] [SWS_Dcm_01685] [SWS_Dcm_01686] [SWS_Dcm_01687] [SWS_Dcm_01688] [SWS_Dcm_01689] [SWS_Dcm_01690] [SWS_Dcm_01691] [SWS_Dcm_01692] [SWS_Dcm_01693] [SWS_Dcm_01694] [SWS_Dcm_01695]
[RS_Diag_04275]	Support of UDS service 0x22 Read DataByIdentifier	[SWS_Dcm_00484] [SWS_Dcm_01335]
[RS_Ids_00810]	Basic SW security events	[SWS_Dcm_01589] [SWS_Dcm_01590] [SWS_Dcm_01702] [SWS_Dcm_01703] [SWS_Dcm_01704] [SWS_Dcm_01705] [SWS_Dcm_01706] [SWS_Dcm_01707] [SWS_Dcm_01708] [SWS_Dcm_01709] [SWS_Dcm_01710] [SWS_Dcm_01711] [SWS_Dcm_01712] [SWS_Dcm_01713] [SWS_Dcm_01714] [SWS_Dcm_01715] [SWS_Dcm_01716] [SWS_Dcm_01717] [SWS_Dcm_01718] [SWS_Dcm_01719] [SWS_Dcm_01720] [SWS_Dcm_01721] [SWS_Dcm_01722] [SWS_Dcm_01723] [SWS_Dcm_01724] [SWS_Dcm_01725] [SWS_Dcm_01726] [SWS_Dcm_01727] [SWS_Dcm_01728] [SWS_Dcm_01729] [SWS_Dcm_01730] [SWS_Dcm_01731] [SWS_Dcm_01732] [SWS_Dcm_01733] [SWS_Dcm_01734] [SWS_Dcm_01735] [SWS_Dcm_01736] [SWS_Dcm_01737] [SWS_Dcm_01738] [SWS_Dcm_01739] [SWS_Dcm_01740] [SWS_Dcm_01741] [SWS_Dcm_01742] [SWS_Dcm_01743] [SWS_Dcm_01744] [SWS_Dcm_01745] [SWS_Dcm_01746] [SWS_Dcm_01747] [SWS_Dcm_01748] [SWS_Dcm_01749] [SWS_Dcm_01750] [SWS_Dcm_01751] [SWS_Dcm_01752] [SWS_Dcm_01753]
[SRS_BSW_00003]	All software modules shall provide version and identification information	[SWS_Dcm_00065]
[SRS_BSW_00101]	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	[SWS_Dcm_00033] [SWS_Dcm_00034] [SWS_Dcm_00035] [SWS_Dcm_00037]
[SRS_BSW_00337]	Classification of development errors	[SWS_Dcm_00040]
[SRS_BSW_00369]	All AUTOSAR Basic Software Modules shall not return specific development error codes via the API	[SWS_Dcm_00044]
[SRS_BSW_00373]	The main processing function of each AUTOSAR Basic Software Module shall be named according the defined convention	[SWS_Dcm_00053]
[SRS_BSW_00407]	Each BSW module shall provide a function to read out the version information of a dedicated module implementation	[SWS_Dcm_00065]
[SRS_BSW_00424]	BSW module main processing functions shall not be allowed to enter a wait state	[SWS_Dcm_00053]





Requirement	Description	Satisfied by
[SRS_BSW_00438]	Configuration data shall be defined in a structure	[SWS_Dcm_00037]
[SRS_BSW_00452]	Classification of runtime errors	[SWS_Dcm_01416]
[SRS_BSW_00482]	Get version information function shall follow a naming rule	[SWS_Dcm_00065]

Table 6.1: Requirements Tracing

7 Functional specification

7.1 General design elements

7.1.1 Submodules within the Dcm module

To define the functionality of the Dcm module, The Dcm SWS models the Dcm module as consisting of the following submodules:

- Diagnostic Session Layer (DSL) submodule: The DSL submodule ensures data flow concerning diagnostic requests and responses, supervises and guarantees diagnostic protocol timing and manages diagnostic states (especially diagnostic session and security).
- Diagnostic Service Dispatcher (DSD) submodule: The DSD submodule processes a stream of diagnostic data. The submodule:
 - Receives a new diagnostic request over a network and forwards it to a data processor.
 - Transmits a diagnostic response over a network when triggered by the data processor (e.g. by the DSP submodule).
- Diagnostic Service Processing (DSP) submodule: The DSP submodule handles the actual diagnostic service (respectively subservice) requests.

The next graphic gives an overview of the interfaces between the submodules DSP, DSD, and DSL within the Dcm module.

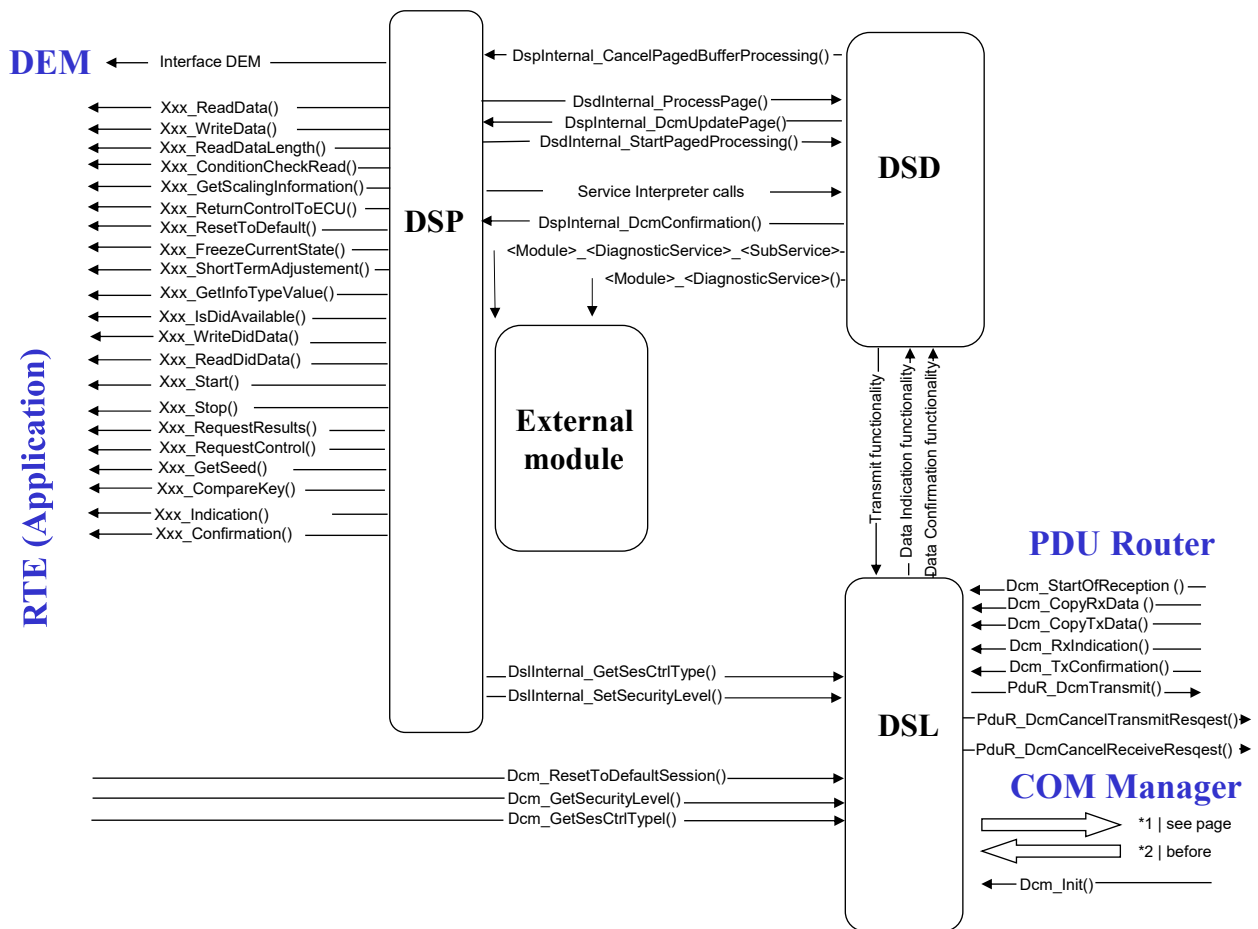


Figure 7.1: Possible interaction between the submodules in the DCM

Note: The implementation of these submodules and the interfaces between them is not mandatory. They are introduced only to improve the readability of the specification.

7.1.2 Negative Response Code (NRC)

The standards defining the [UDS Services](#) and [OBD Services](#) define the negative response codes (NRCs). The [Dcm SWS](#) uses these NRCs in the interfaces between the [Dcm](#) and other BSW modules and the SW-Cs. These NRCs are defined in the data type [Dcm_NegativeResponseCodeType](#).

[SWS_Dcm_01075] [The order of the transmitted [NRC](#) shall be compliant with the one described in ISO14229-1 [\[1\]](#).]

7.1.3 Non-volatile information

Several features of the [Dcm](#) require non-volatile information to be initialized. AUTOSAR does not describe how this information is accessed or if the information is already available when the [Dcm](#) is initialized. Therefore the access for the non-volatile information is implementation specific and has to be ensured during integration.

[SWS_Dcm_00870] [The [Dcm](#) shall check if the NvM is read out correctly. If the non-volatile information could not read out correct the [Dcm](#) shall start a default reaction. The default reaction is described in the chapter were the usage of the non-volatile data is described.]

[SWS_Dcm_01048] [If the [Dcm](#) cancels a service with NvM access, it shall call `NvM_CancelJobs()`.]

The service is cancelled either by reaching the maximum number of RCRRP NRCs or by protocol preemption.

7.1.4 Types

[SWS_Dcm_00969] [The [Dcm](#) shall treat non-integer data types (e.g. `uint8[n]`) either like integer data types of the matching size or leave their contents uninterpreted in case [DcmDspDataEndianness](#) is configured to OPAQUE.]

[SWS_Dcm_00970] [The [Dcm](#) module shall interpret opaque data as `uint8[n]` and shall always map it to an n-bytes sized signal. For opaque data endianness, [DcmDspDataEndianness](#) has to be configured to OPAQUE.]

[SWS_Dcm_00971] [The [Dcm](#) shall extend the endianness conversion defined in [13], to signed data types.]

In [13] (Chapter 2.4) the endianness conversion is defined for unsigned data types. The associated configurations can be found in the configuration [DcmDspData](#).

7.1.4.1 Atomic types overview

Data bit size	ATOMIC						
	1 (Byte aligned)	8	16	32	8	16	32
DcmDspDidDataType	BOOLEAN	UINT8	UINT16	UINT32	SINT8	SINT16	SINT32
DcmDspDataEndianness	N/A	N/A	LE,BE	LE,BE	N/A	LE,BE	LE,BE
DcmDspDataUse Port	S/R, C/S and I/O						
resulting ImplType	boolean	UINT8	UINT16	UINT32	SINT8	SINT16	SINT32

Figure 7.2: Atomic types overview

7.1.4.2 Data array types overview

Data bit size	Field (Static)						Field (Dynamic)
	8-8*N		16-16*N		32-32*N		8-8*N
DcmDspDataByteSize	Any		(size MOD 2)=0		(size MOD 4)=0		Any
DcmDspDidDataType	UINT8_N	SINT8_N	UINT16_N	SINT16_N	UINT32_N	SINT32_N	UINT8_DYN
DcmDspDataEndianness	N/A		LE, BE				N/A
DcmDspDataUse Port	S/R, C/S, FNC, NVM		S/R, C/S				C/S, FNC
resulting ImplType	DataArrayTypeUint8_(Data)	DataArrayTypeSint8_(Data)	DataArrayTypeUint16_(Data)	DataArrayTypeSint16_(Data)	DataArrayTypeUint32_(Data)	DataArrayTypeSint32_(Data)	DataArrayTypeUint8_(Data)

Figure 7.3: Data array types overview

7.1.4.3 Nested Data types overview

The data types used for DIDs and Diagnostic Routines may be defined in a nested way. A DID or Routine Argument may consist of several elements, e.g. an array where each array element is a structure, and that structure consists of several data elements typed by primitives, arrays, or further structures.

In the configuration this is represented by a dedicated pattern:

- the root data element defines a pool of nested elements (e.g. [DcmDspDidSignalCompositePool](#) in [DcmDspDid](#))
- the root data element has an anchor object (e.g. [DcmDspDidSignal](#)), this anchor object may represent primitive or array data, in this case there is no reference to a pool element defined (e.g. [DcmDspDidSignalCompositeRef](#)).
- if the anchor object represents a structure, then it defines one or more references to pool elements (e.g. [DcmDspDidSignalCompositeRef](#)). This defines that the referenced pool elements build the content of that structure.

A pool element may itself be a nested definition. This allows for an arbitrary nesting of structured elements.

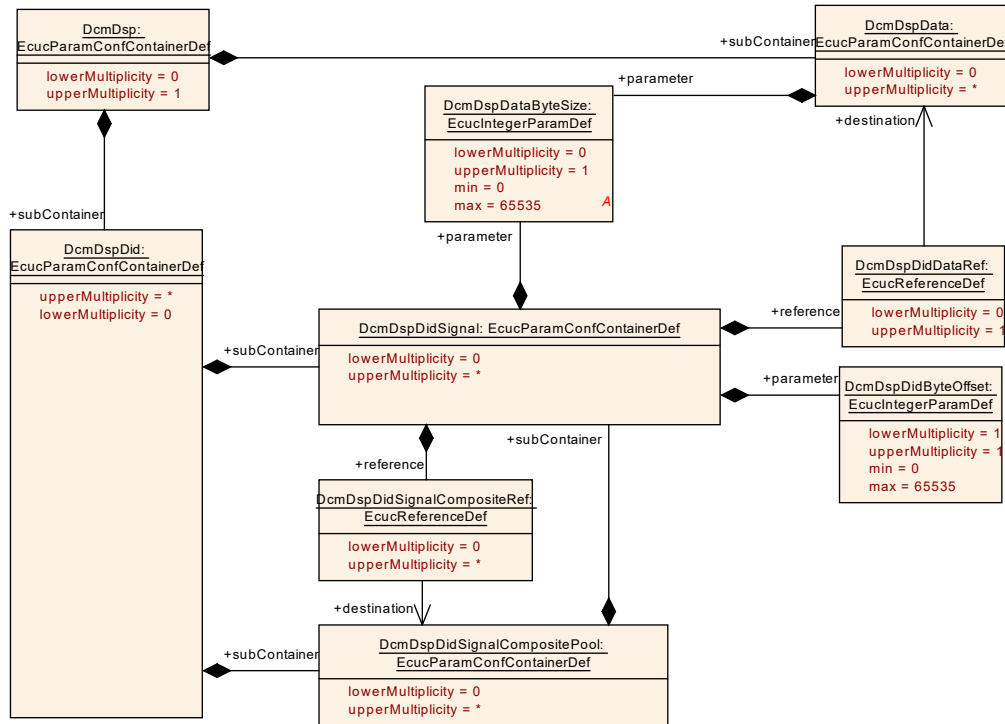


Figure 7.4: Nested data for **DcmDspDid**

[SWS_Dcm_01633] [If a **DcmDspDidSignal** contains at least one **DcmDspDidSignalCompositeRef**, then that **DcmDspDidSignal** shall only define the following parameters:

- **DcmDspDidByteOffset**
- **DcmDspDataByteSize**.

]

Specifically this means that **DcmDspDidSignal** shall not define any **DcmDspDidDataRef**, as this would give details about specific data type information of primitive or array data types. If the **DcmDspDidSignal** contains a **DcmDspDidSignalCompositeRef** then this **DcmDspDidSignal** is a nested data type.

[SWS_Dcm_01634] [The nested data type occupies a consecutive number of bytes (**DcmDspDataByteSize**) starting at the position **DcmDspDidByteOffset**.]

[SWS_Dcm_01635] [All of the data types that are defined in the scope of the nested data type shall be placed inside the area defined by **DcmDspDataByteSize** and **DcmDspDidByteOffset**.]

[SWS_Dcm_01636] [The **DcmDspDidByteOffset** values of nested data elements shall be given to the absolute starting position of the DID (NOT relative to the respective composite structure).]

[SWS_Dcm_01637] [Each `DcmDspDidSignalCompositePool` shall only be referenced up to once by a `DcmDspDidSignalCompositeRef`. Each `DcmDspDidSignalCompositePool` element is specifically defined to be at a dedicated position in the DID and can only occur once in the definition of the `DcmDspDid`.]

[SWS_Dcm_01638] [A `DcmDspDidSignalCompositeRef` shall only refer to `DcmDspDidSignalCompositePool` elements which are defined in the scope of the same `DcmDspDid` as the owner of the `DcmDspDidSignalCompositeRef`.]

The specification above has been defined on the basis of the `DcmDspDid`. The same pattern for the definition of nested data types has been applied to further parts of the configuration. The following list gives the respective correspondence between the configuration items used above and the further configuration items used for the definition of nested data types.

[SWS_Dcm_01639] [The nesting of data types does not only apply to `DcmDspDid`, but is also supported for the following definitions:

- `DcmDspDid` corresponds to
 - `DcmDspStartRoutineIn`
 - `DcmDspStartRoutineOut`
 - `DcmDspStopRoutineIn`
 - `DcmDspStopRoutineOut`
 - `DcmDspRequestRoutineResultsIn`
 - `DcmDspRequestRoutineResultsOut`
- `DcmDspDidSignal` corresponds to
 - `DcmDspStartRoutineInSignal`
 - `DcmDspStartRoutineOutSignal`
 - `DcmDspStopRoutineInSignal`
 - `DcmDspStopRoutineOutSignal`
 - `DcmDspRequestRoutineResultsInSignal`
 - `DcmDspRequestRoutineResultsOutSignal`
- `DcmDspDidSignalCompositePool` corresponds to
 - `DcmDspStartRoutineInSignalCompositePool`
 - `DcmDspStartRoutineOutSignalCompositePool`
 - `DcmDspStopRoutineInSignalCompositePool`
 - `DcmDspStopRoutineOutSignalCompositePool`

- DcmDspRequestRoutineResultsInSignalCompositePool
- DcmDspRequestRoutineResultsOutSignalCompositePool
- DcmDspDidSignalCompositeRef corresponds to
 - DcmDspStartRoutineInSignalCompositeSignalRef
 - DcmDspStartRoutineOutSignalCompositeSignalRef
 - DcmDspStopRoutineInSignalCompositeSignalRef
 - DcmDspStopRoutineOutSignalCompositeSignalRef
 - DcmDspRequestRoutineResultsInSignalCompositeSignalRef
 - DcmDspRequestRoutineResultsOutSignalCompositeSignalRef

]

The example nested data type in figure 7.5 is defined as *A* which consists of *X* and *Y* defined as primitive types.

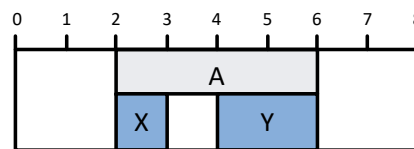


Figure 7.5: Nested data type for DcmDspStartRoutineIn

A starts at position 2 of the DcmDspStartRoutineIn and has a size of 4 byte. The nested leaf elements *X* and *Y* are defined within the range of the structure *A* and do not overlap. In this example there is even a gap left between *X* and *Y*.

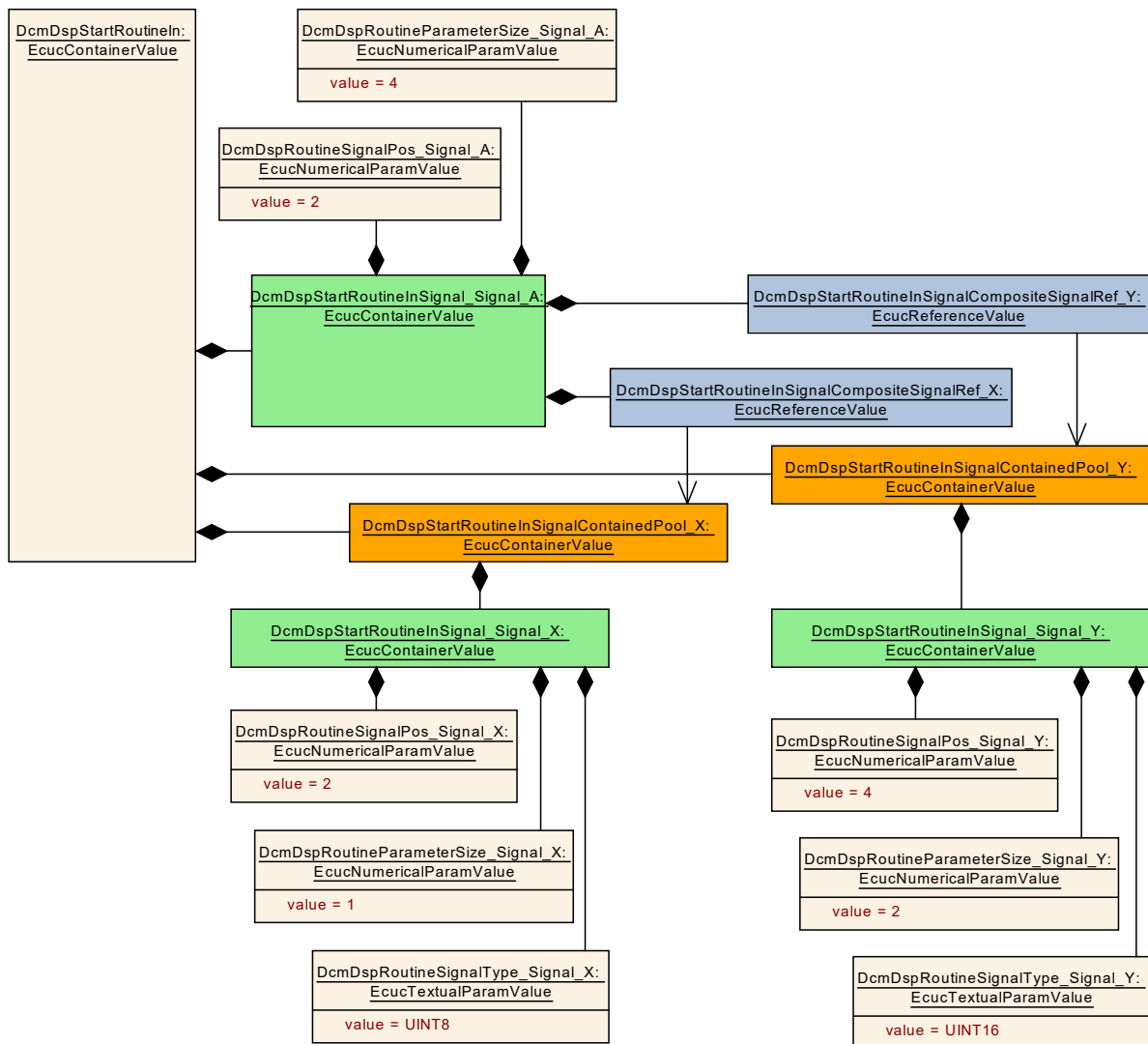


Figure 7.6: Example configuration of nested data for `DcmDspStartRoutineIn`

In figure 7.6 an example configuration is shown for a nested data type of figure 7.5.

The container `DcmDspStartRoutineIn` holds the definition of the root element `DcmDspStartRoutineInSignal_Signal_A` as well as the two pool elements `DcmDspStartRoutineInSignalContainedPool_X` and `DcmDspStartRoutineInSignalContainedPool_Y`.

The root element `DcmDspStartRoutineInSignal_Signal_A` defines its absolute start position as 2 (`DcmDspRoutineSignalPos_Signal_A`) and its size in byte as 4 (`DcmDspRoutineParameterSize_Signal_A`). The root element `DcmDspStartRoutineInSignal_Signal_A` also defines that there are two nested elements part of that root element using the references `DcmDspStartRoutineInSignalContainedPool_X` and `DcmDspStartRoutineInSignalContainedPool_Y`. These references point to pool elements defined in the scope of the same `DcmDspStartRoutineIn`.

The referenced pool elements define each their absolute position (*DcmDspRoutineSignalPos_Signal_X*) as well as their size (*DcmDspRoutineParameterSize_Signal_X*) and primitive type (*DcmDspRoutineSignalType_Signal_X*).

7.1.4.4 Data types constraints

[SWS_Dcm_CONSTR_06002] Existence of size parameter [*DcmDspDataByteSize* shall be present if *DcmDspDataType* is set to: UINT8_N, SINT8_N, UINT16_N, SINT16_N, UINT32_N, SINT32_N, DOUBLE_N, UINT64_N, SINT64_N, FLOAT_N or UINT8_DYN.]

Note: *DcmDspDataByteSize* is not required for primitive datatypes

[SWS_Dcm_CONSTR_06035] Restrictions on size parameter for 16 Bit arrays [*DcmDspDataByteSize* shall be a multiple of 2 if the value is greater than 2 and *DcmDspDataType* is UINT16_N or SINT16_N.]

[SWS_Dcm_CONSTR_06036] Restrictions on size parameter for 32 Bit arrays [*DcmDspDataByteSize* shall be a multiple of 4 if the value is greater than 4 and *DcmDspDataType* is UINT32_N or SINT32_N.]

[SWS_Dcm_CONSTR_06037] Restrictions on size parameter for 64 Bit arrays [*DcmDspDataByteSize* shall be a multiple of 8 if the value is greater than 8 and *DcmDspDataType* is DOUBLE_N, UINT64_N or or SINT64_N.]

[SWS_Dcm_CONSTR_06008] Define the usage of *DcmDspRoutineParameterSize* parameter [*DcmDspRoutineParameterSize* is only required if *DcmDspRoutineSignalType* is set to SINT8_N, SINT16_N, SINT32_N,, SINT64_N, UINT8_N, UINT16_N, UINT32_N, UINT64_N, DOUBLE_N, FLOAT_N or VARIABLE_LENGTH.]

[SWS_Dcm_CONSTR_06011] Only last parameters in *RID* may have a variable length [*DcmDspRoutineSignalType* with VARIABLE_LENGTH is only valid for the last signal.]

[SWS_Dcm_CONSTR_06012] Existence of size parameter [*DcmDspPidDataByteSize* shall be present if *DcmDspPidDataType* is set to: UINT8_N, SINT8_N, UINT16_N, SINT16_N, UINT32_N, SINT32_N, UINT64_N, SINT64_N, FLOAT_N, DOUBLE_N.]

Note: *DcmDspPidDataByteSize* is not required for primitive datatypes

[SWS_Dcm_CONSTR_06040] Restrictions on size parameter for 16 Bit arrays [*DcmDspPidDataByteSize* shall be a multiple of 2 if the value is greater than 2 and *DcmDspPIDDataType* is UINT16_N or SINT16_N.]

[SWS_Dcm_CONSTR_06041] Restrictions on size parameter for 32 Bit arrays [[DcmDspPidDataByteSize](#) shall be a multiple of 4 if the value is greater than 4 and [DcmDspPIDDataType](#) is [UINT32_N](#) or [SINT32_N](#).]

[SWS_Dcm_CONSTR_06042] Restrictions on size parameter for 64 Bit arrays [[DcmDspPidDataByteSize](#) shall be a multiple of 8 if the value is greater than 8 and [DcmDspPIDDataType](#) is [DOUBLE_N](#), [UINT64_N](#) or [SINT64_N](#).]

[UINT8](#) shall be used as (implementation) data type for bit lengths between 1 and 8

[SWS_Dcm_CONSTR_06026] Usage of variable data length in case of S/R communication, NvRam access or ECU signal access [In case [DcmDspDataUsePort](#) is set to {[USE_DATA_SENDER_RECEIVER](#), [USE_DATA_SENDER_RECEIVER_AS_SERVICE](#), [USE_ECU_SIGNAL](#)}, the usage of variable data length shall be not allowed.]

[SWS_Dcm_CONSTR_06031] [The [DcmDspData.SHORT-NAME](#) and [DcmDspPidData.SHORT-NAME](#) shall be distinct.]

Note: Variable data length is only possible with [UINT8](#) arrays with [DcmDspDataType](#) set to [UINT8_DYN](#).

7.1.4.5 Dcm_OpStatusType

For the operation using the [Dcm_OpStatusType](#), the [Dcm](#) shall work as follow :

[SWS_Dcm_00527] [At first call of an operation using the [Dcm_OpStatusType](#), the [Dcm](#) call the operation with [OpStatus](#) = [DCM_INITIAL](#)]

[SWS_Dcm_00528] [If the value [DCM_E_FORCE_RCRRP](#) is returned from an operation using [Dcm_OpStatusType](#), the [Dcm](#) shall invoke the transmit request for RCR-RP (NRC 0x78 transmission) and the [Dcm](#) shall not realize further invocation of the operation till RCR-RP is transmitted.]

[SWS_Dcm_00529] [After transmit confirmation of a RCR-RP transmitted on the context of [\[SWS_Dcm_00528\]](#), the [Dcm](#) calls, from [Dcm_MainFunction](#) (due to call context), the operation again with [OpStatus](#) = [DCM_FORCE_RCRRP_OK](#).]

[SWS_Dcm_00530] [If a [DCM_E_PENDING](#) value is returned from an operation using the [Dcm_OpStatusType](#), the [Dcm](#) call the operation on each [Dcm_MainFunction](#) call with [OpStatus](#) = [DCM_PENDING](#) as long as [DCM_E_PENDING](#) is returned.]

7.1.4.6 Dcm_Cemr_{DID}Type

For ease of use in SWC, the [Dcm](#) generates a symbolic name to access the [CEMR](#) bit according to [Dcm_Cemr_{DID}Type](#) (see [\[SWS_Dcm_91087\]](#)) for each data element

used in a [DID](#) with IO control. The SWC can work only with the generated symbolic value of the bitfield text table to mask out a certain bit. This helps to avoid confusion, while the first [RID](#) bit on the network controls the first parameter of the [DID](#) but that Bit is the most significant Bit in the MSB. The symbolic name also helps if the size of the [RID](#) is changing. In that case the bitmask changes as well, but the symbolic value is always updated.

7.2 Diagnostic Session Layer (DSL)

7.2.1 Introduction

[SWS_Dcm_00030]

Upstream requirements: [RS_Diag_04003](#), [RS_Diag_04015](#)

[All functional areas of the [DSL](#) submodule shall be in conformance with the specifications ISO14229-1 [1], ISO14229-2 [3] and the network-independent part of ISO15765-3 [5].]

There is no network-dependent functional area in the [DSL](#) submodule. Within the configuration, some parameters can be set dependent on the network.

7.2.2 Use cases

The [DSL](#) submodule provides the following functionalities:

- Session handling (as required by ISO14229-1 [1] and ISO 15765-3 [5])
- Application layer timing handling (as required by ISO14229-1 [1] and ISO 15765-3 [5])
- Specific response behavior (as required by ISO14229-1 [1] and ISO 15765-3 [5])
- Authentication state handling per diagnostic connection (as required by ISO 14229-1:2018)
 - Provide authentication state per connection
 - Manage authentication state transitions

7.2.3 Interaction with other modules

The [DSL](#) has the following interaction with other modules:

- PduR module
 - PduR module provides data of incoming diagnostic requests.
 - The [DSL](#) submodule triggers output of diagnostic responses.

- DSD submodule
 - The DSL submodule informs the DSD submodule about incoming requests and provides the data.
 - The DSD submodule triggers output of diagnostic responses.
- SW-Cs / DSP submodule. The DSL submodule provides access to security and session state.
- ComM module
 - The DSL submodule guarantees the communication behavior required by the ComM module

7.2.4 Functional description

7.2.4.1 Overview

The DSL submodule provides the following functionality:

Request Handling

- Forward requests from the PduR module to the DSD submodule.
- Concurrent TesterPresent ("keep alive logic").

Response Handling

- Forward responses from the DSD submodule to the PduR module.
- Guarantee response timing to tester.
- Support of periodic transmission.
- Support of ResponseOnEvent (ROE) transmission.
- Support of segmented response.
- Support of ResponsePending response triggered by the application.

Security Level Handling

- Manage security level.

Authentication

- Manage authentication state per diagnostic connection

Session State Handling

- Manage session state.
- Keep track of active non-default sessions.
- Allows modifying timings.

Diagnostic Protocol Handling

- Handling of different diagnostic protocols.
- Manage resources.

Communication Mode Handling

- Handling of communication requirements (Full- / Silent- / No Communication).
- Indicating of active / inactive diagnostic.
- Enabling / disabling all kinds of diagnostic transmissions.

7.2.4.2 Forward requests from the PduR module to the DSD submodule

The PduR module indicates the Dcm module whenever a reception of new diagnostic request content is started on a `DcmDslProtocolRxPduId`, which is assigned to the Dcm module. This is done by calling `Dcm_StartOfReception`, which inform the Dcm module of the data size to be received and provides the data of the first frame or single frame, and allows the Dcm to reject the reception if the data size overflows its buffer size, or if the requested service is not available. The further call to `Dcm_CopyRxData` request the Dcm module to copy the data from the provided buffer to the Dcm buffer. If the reception of a diagnostic request is finished (when `Dcm_StartOfReception` succeeded) the PduR module will call `Dcm_TpRxIndication` to give a receive indication to the Dcm module. The Dcm shall be able to use generic connections, where the addressing information is provided to Dcm by `Dcm_StartOfReception` via the MetaData of the `DcmRxPdu`. This addressing information must be stored and used for the response and for detection of requests from the same tester. see section 7.2.4.5 Generic Connection Handling for further details.

[SWS_Dcm_00111]

Upstream requirements: [RS_Diag_04249](#)

[The DSL submodule shall forward received data to the DSD submodule only after a call of `Dcm_TpRxIndication` with parameter Result = E_OK (see [SWS_Dcm_00093]).]

[SWS_Dcm_00241] [As soon as a request message is received (after a call of `Dcm_TpRxIndication` with parameter Result = E_OK (see [SWS_Dcm_00093]) and until a call to `Dcm_TpTxConfirmation` (see [SWS_Dcm_00351]) for the associated Tx-DcmPduId), the DSL submodule shall block the corresponding DcmPduId. During the processing of this request, no other request of the same `DcmDslConnection` (e.g. an enhanced session can be ended by a OBD session) can be received, until the corresponding response message is sent and the DcmPduId is released again (except for `Concurrent TesterPresent` requests).]

More descriptions of the APIs (prototype, input/output parameter) can be found in the interface description of PduR module [14].

It is allowed to have different DcmPduls for different diagnostic communication applications. For example:

- [OBD DcmDslProtocolRxPduId](#): for reception of [OBD](#) requests,
- [OBD DcmTxPdul](#): for transmission of [OBD](#) responses,
- [UDS phys DcmDslProtocolRxPduId](#): for reception of [UDS](#) physically addressed requests,
- [UDS func DcmDslProtocolRxPduId](#): for reception of [UDS](#) functionally addressed requests,
- [UDS DcmTxPdul](#): for transmission of [UDS](#) responses.

Address type (physical/functional addressing) is configured per [DcmDslProtocolRxPduId](#). A configuration per [DcmDslProtocolRxPduId](#) is possible because there will always be different [DcmDslProtocolRxPduId](#) values for functional and physical receptions, independent of the addressing format of the Transport Layer (extended addressing, normal addressing).

7.2.4.2.1 Dcm_StartOfReception

[SWS_Dcm_00444] [If the requested size is large than the buffer available in the DCM, the function [Dcm_StartOfReception](#) shall return BUFREQ_E_OVFL (see [\[SWS_Dcm_00094\]](#)).]

[SWS_Dcm_00642]

Upstream requirements: [RS_Diag_04147](#)

[When the [API Dcm_StartOfReception](#) is invoked with TpSduLength equal to 0, the value BUFREQ_E_NOT_OK shall be returned.]

[SWS_Dcm_01671] Message reception according to ISO 14229-2

Upstream requirements: [RS_Diag_04196](#)

[The [Dcm](#) shall receive and handle diagnostic messages according to the session layer services of ISO14229-2 [\[3\]](#).]

[SWS_Dcm_01672] Mapping of T_DataSOM.ind

Upstream requirements: [RS_Diag_04196](#)

[The [Dcm](#) shall map a call of [Dcm_StartOfReception](#) to the T_DataSOM.ind from ISO14229-2 [\[3\]](#).]

[SWS_Dcm_01673] Mapping of T_Data.ind

Upstream requirements: [RS_Diag_04196](#)

[The [Dcm](#) shall map a call of [Dcm_TpRxIndication](#) to T_Data.ind from ISO14229-2 [\[3\]](#).]

[SWS_Dcm_01674] Multi client handling

Upstream requirements: [RS_Diag_04209](#)

[The [Dcm](#) shall handle requests from multiple clients according to ISO14229-1 [1] Annex J.]

"Figure J.2 - Multiple client handling flow" of ISO14229-1 [1] gives a comprehensive overview of parallel client request handling. A major aspect is the support of additional protocol resources. The [Dcm](#) follows this flow chart and provides protocol resources per configured [DcmDslProtocolRow](#). In default session this allows the [Dcm](#) to process requests in parallel.

[SWS_Dcm_01675] Protocol resource availability

Upstream requirements: [RS_Diag_04209](#)

[The [Dcm](#) shall assign additional protocol resource to each configured [DcmDslProtocolRow](#) with a own [DcmDslBuffer](#).]

The configuration parameter [DcmDslDiagRespOnSecondDeclinedRequest](#) defines the behavior of the decision box "Is NRC 0x21 handling supported?" of "Figure J.2 - Multiple client handling flow" of ISO14229-1 [1].

[SWS_Dcm_01676] Decline a second request with NRC 0x21

Upstream requirements: [RS_Diag_04209](#)

[If the [Dcm](#) is rejecting a second received request according to ISO14229-1 [1] Annex J and [DcmDslDiagRespOnSecondDeclinedRequest](#) is set to TRUE, the [Dcm](#) shall return a NRC 0x21 as response for the rejected second received request.]

[SWS_Dcm_01677] Ignoring a second request

Upstream requirements: [RS_Diag_04209](#)

[If the [Dcm](#) is rejecting a second received request according to ISO14229-1 [1] Annex J and [DcmDslDiagRespOnSecondDeclinedRequest](#) is set to FALSE, the [Dcm](#) shall ignore the new request. .]

ISO14229-1 [1] Annex J misses a definition of the server in case a second diagnostic request is received from the same connection, while the server is still processing a request on that connection. The [Dcm](#) fills this gap by not accepting further diagnostic requests for processing. In that case no response is send, even no NRC 0x21.

[SWS_Dcm_01678] Multiple requests on the same connection

Upstream requirements: [RS_Diag_04209](#)

[If the [Dcm](#) receives a further diagnostic request for processing and the [Dcm](#) is still processing an ongoing request on the same connection, the [Dcm](#) shall decline the new received request.]

7.2.4.2.2 Dcm_CopyRxData

[SWS_Dcm_00443] [If [Dcm_StartOfReception](#) returns BUFREQ_OK, the further call to [Dcm_CopyRxData](#) shall copy the data from the buffer provided in info parameter) to the [Dcm](#) buffer and update the bufferSizePtr parameter with remaining free place in [Dcm](#) receive buffer after completion of this call.]

[SWS_Dcm_00996] [When the [API Dcm_CopyRxData](#) is invoked with SduLength from info equal to 0, the value BUFREQ_OK shall be returned and bufferSizePtr shall be filled with the remaining size of the Rx buffer.]

Note: The size of the Rx buffer is based on the buffer length, which is returned in the parameter RxBufferSizePtr of [API Dcm_StartOfReception](#).

[SWS_Dcm_00342] [After starting to copy the received data (see [\[SWS_Dcm_00443\]](#)), the [Dcm](#) module shall not access the receive buffer until it is notified by the service [Dcm_TpRxIndication](#) about the successful completion or unsuccessful termination of the reception.]

Note: [Dcm_TpRxIndication](#) is only expected when [Dcm_StartOfReception](#) succeeded

7.2.4.2.3 Dcm_TpRxIndication

[SWS_Dcm_00344] [If [Dcm_TpRxIndication](#) is called with parameter Result different from E_OK, then the [Dcm](#) module shall not evaluate the buffer assigned to the I-PDU, which is referenced in parameter DcmRxDulId.]

Rationale for [\[SWS_Dcm_00344\]](#): It is undefined which part of the buffer contains valid data in this case

7.2.4.3 Concurrent TesterPresent ("keep alive logic")

The [Concurrent TesterPresent](#) is defined by ISO14229-1 [\[1\]](#) and also called "bypass logic" or "keep alive logic". The purpose is to keep the non-default session active and reset the S3 timer defined by ISO14229-2 [\[3\]](#). [Concurrent TesterPresent](#) are only received by functional addressing and are processed independent from any activity on a physical request and service processing.

[SWS_Dcm_01666] Concurrent tester present support

Upstream requirements: [RS_Diag_04249](#)

[The [Dcm](#) shall handle and process the [Concurrent TesterPresent](#) according to ISO14229-1 [\[1\]](#).]

In addition to ISO14229-1 [1] the *Dcm* limits the processing for received *Concurrent TesterPresent* messages to the connection from where the request to enter a non-default session was received. This avoids that a tester that did not request the non-default session can influence the protocol and timing behavior of another tester.

[SWS_Dcm_01667] Concurrent tester present from different connections

Upstream requirements: [RS_Diag_04249](#)

[The *Dcm* shall only process and handle *Concurrent TesterPresent* messages received on the same *DcmDslConnection* that requested the *Dcm* to enter the non-default session.]

7.2.4.3.1 Dcm_CopyTxData

If the copied data is smaller than the length requested to transmit within the service *PduR_DcmTransmit()* the *Dcm* module will be requested by the service *Dcm_CopyTxData* to provide another data when the current copied data have been transmitted.

[SWS_Dcm_00346] [If the function *Dcm_CopyTxData* is called and the *Dcm* module successfully copied the data in the buffer provided in *info* parameter, then the function shall return *BUFREQ_OK*.]

[SWS_Dcm_00350] [Caveats of *Dcm_CopyTxData*:

- The value of parameter *availableDataPtr* of function *Dcm_CopyTxData* shall not exceed the number of Bytes still to be sent.
- If this service returns *BUFREQ_E_NOT_OK* the transmit requests issued by calling the service *PduR_DcmTransmit()* is still not finished. A final confirmation (indicating an error with call of service *Dcm_TpTxConfirmation*) is required to finish this service and to be able to start another transmission (call to *PduR_DcmTransmit()*). So it is up to the transport protocol to confirm the abort of transmission.

]

7.2.4.3.2 Dcm_TpTxConfirmation

[SWS_Dcm_00352] [If the function *Dcm_TpTxConfirmation* is called, then the *Dcm* module shall unlock the transmit buffer.]

[SWS_Dcm_00353] [If the function *Dcm_TpTxConfirmation* is called, then the *Dcm* module shall stop error handling (Page buffer timeout, $P2ServerMax/P2*ServerMax$ timeout).]

For transmission via FlexRay the following restriction has to be considered: Since the FlexRay Specification does not mandate the existence of a transmit interrupt, the exact meaning of this confirmation (i.e. "transfer into the FlexRay controller's send buffer" OR "transmission onto the FlexRay network") depends on the capabilities of the FlexRay communication controller and the configuration of the FlexRay Interface.

7.2.4.4 Forward responses from the DSD submodule to the PduR module

[SWS_Dcm_00114]

Upstream requirements: [RS_Diag_04249](#)

[The [DSD](#) submodule shall request the [DSL](#) submodule for transmission of responses.]

[SWS_Dcm_00115]

Upstream requirements: [RS_Diag_04249](#)

[When the diagnostic response of a [DcmDslMainConnection](#) is ready, the [DSL](#) submodule shall trigger the transmission of the diagnostic response to the PduR module by calling `PduR_DcmTransmit()` using the corresponding [DcmDslProtocolTxPduRef](#) parameter as `PduId`.]

[SWS_Dcm_01072] [In case of PeriodicTransmission, the [Dcm](#) shall provide in the call to `PduR_DcmTransmit()` the full payload data and expect no call to [Dcm_CopyTxData](#).]

[SWS_Dcm_01073]

Upstream requirements: [RS_Diag_04249](#)

[In case of PeriodicTransmission, the [Dcm](#) will be called for periodic transmission with [Dcm_TxConfirmation](#) to indicate the transmission result.]

Responses are sent with the `DcmTxPduId`, which is linked in the [Dcm](#) module configuration to the [DcmDslProtocolRxPduId](#), i.e. the [ID](#) the request was received with (see configuration parameter [DcmDslProtocolTx](#)). Within `PduR_DcmTransmit()` only the length information and, for generic connections, the addressing information, is given to the PduR module. After the [Dcm](#) module has called successfully `PduR_DcmTransmit()`, the PduR module will call [Dcm_CopyTxData](#) to request the [Dcm](#) module to provide the data to be transmitted and will call [Dcm_TpTxConfirmation](#) after the complete [PDU](#) has successfully been transmitted or an error occurred. see section 7.2.4.5 "Generic Connection Handling for further details on address information handling within generic connections".

[SWS_Dcm_00117]

Upstream requirements: [RS_Diag_04249](#)

[If the [DSL](#) submodule receives a confirmation after the complete [Dcm PDU](#) has successfully been transmitted or an error occurred by a call of [Dcm_TpTxConfirmation](#), then the [DSL](#) submodule shall forward this confirmation to the [DSD](#) submodule.]

[SWS_Dcm_00118] [In case of a failed transmission (failed PduR_DcmTransmit() request) or error confirmation (Dcm_TpTxConfirmation with error), the DSD submodule shall not repeat the diagnostic response transmission.]

Note: Dcm_TpTxConfirmation is only expected when PduR_DcmTransmit succeeded.

[SWS_Dcm_01166] [If the Multiplicity of DcmDslProtocolTx is set to "0" the Dcm shall process the received diagnostic request without sending a response.]

More descriptions of the APIs (prototype, input/output parameter) can be found in the interface description of the PduR module [14].

7.2.4.5 Generic Connection Handling

The Dcm shall be able to handle generic connections, identified by DcmPdus with MetaDataItems of type SOURCE_ADDRESS_16 and TARGET_ADDRESS_16. These connections carry the actual tester address at run time. Generic connections are supported for diagnostics over IP and FlexRay diagnostics, and CAN diagnostics using normal fixed or mixed 29 bit addressing formats according to ISO15765-2 [15]. Depending on the actual layout of the CAN IDs, generic connections could also be used for extended or normal and mixed 11 bit addressing formats. The Dcm is not aware of the actual addressing format used by CanTp. Several connections may reference the same DcmPdus.

[SWS_Dcm_CONSTR_06044] [Generic connections shall be consistent. This means that the MetaDataItems and the PduLength of all referenced PDUs of a DcmDslConnection (DcmDslProtocolRxPduRef, DcmDslProtocolTxPduRef, DcmDslPeriodicTxPduRef) are identical.]

[SWS_Dcm_00848] [The source address of diagnostic requests received via a generic connection must be stored. It is provided in the MetaDataItem SOURCE_ADDRESS_16 provided via Dcm_StartOfReception.]

[SWS_Dcm_00849] Target address for generic connection transmission

Upstream requirements: RS_Diag_04153

[If the Dcm is about to send a response, response on event, or periodic message for a generic connection request, the Dcm shall set TARGET_ADDRESS_16 to the value of the stored source address in the MetaDataPtr in the PduR_DcmTransmit().]

[SWS_Dcm_01429] [The source address of diagnostic requests received via a generic connection shall be provided in the parameter TesterSourceAddress to the application [SWS_Dcm_01341], [SWS_Dcm_01342], [SWS_Dcm_00694], [SWS_Dcm_00698]].]

[SWS_Dcm_01347]

Upstream requirements: [RS_Diag_04153](#)

[The target address of diagnostic requests received via a generic connection can be provided in the MetaDataItem TARGET_ADDRESS_16 received via Dcm_StartOfReception(). In this case, the Dcm shall ignore physical requests where the target address is not equal to the configured ECU address [DcmDspProtocolEcuAddr](#).]

[SWS_Dcm_01348]

Upstream requirements: [RS_Diag_04153](#)

[The source address of the response transmitted via generic connections can be read from the configuration parameter [DcmDspProtocolEcuAddr](#). It shall be provided to PduR_DcmTransmit() in the MetaDataItem SOURCE_ADDRESS_16, if that is configured for the transmit PDU.]

Note: If different source addresses are required for certain transmitted diagnostic messages of the same [DcmDslProtocolRow](#), the MetaDataItem SOURCE_ADDRESS_16 can be omitted from the PDUs, and the address can then be configured in the lower layers. The same is possible for physical requests, where the TARGET_ADDRESS_16 can be omitted from the PDUs.

7.2.4.6 Guarantee timing to tester by sending busy responses**[SWS_Dcm_00024]**

Upstream requirements: [RS_Diag_04016](#), [RS_Diag_04249](#)

[If the Application (or the DSP submodule) is able to perform a requested diagnostic task, but needs additional time to finish the task and prepare the response, then the DSL submodule shall send a negative response with NRC 0x78 (Response pending) when reaching the response time ([DcmDspSessionP2ServerMax](#) - [DcmTimStrP2ServerAdjust](#) respectively [DcmDspSessionP2StarServerMax](#) - [DcmTimStrP2StarServerAdjust](#)).]

Rationale for [\[SWS_Dcm_00024\]](#): The DSL submodule guarantees the response timing to tester.

[SWS_Dcm_00119] [The DSL submodule shall send negative responses as required in [\[SWS_Dcm_00024\]](#) from a separate buffer.]

Rationale for [\[SWS_Dcm_00119\]](#): This is needed in order to avoid overwriting the ongoing processing of requests, e.g. the application already prepared response contents in the diagnostic buffer. The number of negative responses with NRC 0x78 (response pending) for one diagnostic request can be limited by the configuration parameter [DcmDslDiagRespMaxNumRespPend](#) to avoid endless NRC 0x78 transmission in case of an application deadlock.

[SWS_Dcm_01567] [The maximum number of negative responses with [NRC 0x78](#) can be configured using the optional configuration parameter [DcmDslDiagRespMaxNumRespPend](#) (see ECUC_Dcm_00693). If this parameter is not configured, the default amount of negative responses with [NRC 0x78](#) is infinite.]

7.2.4.7 Support of periodic transmission

The [UDS](#) service [ReadDataByPeriodicIdentifier](#) (0x2A) allows the tester to request the periodic transmission of data record values from the ECU identified by one or more [periodicDataIdentifiers](#).

[SWS_Dcm_00122] [The [Dcm](#) module shall send responses for periodic transmissions using a separate protocol and a separate buffer of configurable size.]

The [DcmDslPeriodicTransmissionConRef](#) configuration parameter allows linking the protocol used to receive the periodic transmission request / transmit the periodic transmission response to the protocol used for the transmission of the periodic transmission messages. Note that multiple [DcmTxPduls](#) can be assigned to the periodic transmission protocol. The [Dcm](#) module respects several restrictions according to the communication mode:

[SWS_Dcm_00123] [Periodic transmission communication shall only take place in Full Communication Mode.]

Periodic transmission events can occur when not in Full Communication Mode. So the following requirement exists:

[SWS_Dcm_00125] [The [Dcm](#) module shall discard periodic transmission events beside Full Communication Mode and shall not queue it for transmission.]

[SWS_Dcm_00126] [Periodic transmission events shall not activate the Full Communication Mode.]

7.2.4.8 Support of segmented response (paged-buffer)

[SWS_Dcm_00028] [If enabled ([DcmPagedBufferEnabled](#)=TRUE), the [Dcm](#) module shall provide a mechanism to send responses larger than the configured and allocated diagnostic buffer.]

[SWS_Dcm_CONSTR_06055] **Dependency for [DcmDslProtocolMaximumResponseSize](#)** [[DcmDslProtocolMaximumResponseSize](#) shall be only present if [DcmPagedBufferEnabled](#) is set to TRUE.]

[SWS_Dcm_01058] [If `DcmPagedBufferEnabled` == TRUE and the generated Response for a Request is longer than `DcmDslProtocolMaximumResponseSize`, the `Dcm` shall respond with `NRC 0x14` (DCM_E_RESPONSETOOLONG).]

[SWS_Dcm_01059] [If `DcmPagedBufferEnabled` == FALSE and the generated Response for a Request is longer than `Dcm_MsgContextType` structure element `resMaxDataLen`, the `Dcm` shall respond with `NRC 0x14` (DCM_E_RESPONSETOOLONG) .]

With paged-buffer handling the ECU is not forced to provide a buffer, which is as large as the maximum length of response. Please note:

- paged-buffer handling is for transmit only - no support for reception.
- paged-buffer handling is not available for the Application (DCM-internal use only).

[SWS_Dcm_01186]

Upstream requirements: [RS_Diag_04147](#)

[The `Dcm` shall provide the correct amount of Data requested by the `TP` or return `BUFREQ_E_BUSY` in case the requested amount of data is not available.]

Note: In case the requested amount of data is not available, the `Dcm` should fill up the paged buffer immediately.

7.2.4.9 Support of ResponsePending response triggered by the Application

In some cases, e.g. in case of routine execution, the Application needs to request an immediate `NRC 0x78` (Response pending), which shall be sent immediately and not just before reaching the response time (`P2ServerMax` respectively `P2*ServerMax`).

When the `Dcm` module calls an operation and gets an error status `DCM_E_FORCE_RCRRP`, the `DSL` submodule will trigger the transmission of a negative response with `NRC 0x78` (Response pending). This response needs to be sent from a separate buffer, in order to avoid overwriting the ongoing processing of the request.

7.2.4.10 Manage security level

[SWS_Dcm_00020]

Upstream requirements: [RS_Diag_04005](#)

[The `DSL` submodule shall save the level of the current active security level.]

For accessing this level, the `DSL` submodule provides interfaces to:

- get the current active security level: `Dcm_GetSecurityLevel`
- set a new security level: `DslInternal_SetSecurityLevel()`

[SWS_Dcm_00033]

Upstream requirements: [SRS_BSW_00101](#), [RS_Diag_04005](#)

[During [Dcm](#) initialization the security level is set to the value 0x00 (DCM_SEC_LEV_LOCKED).]

[SWS_Dcm_00139] [The [DSL](#) shall reset the security level to the value 0x00 (i.e. the security is enabled) under one of the following conditions: - if a transition from any diagnostic session other than the defaultSession to another session other than the defaultSession (including the currently active diagnostic session) is performed or - if a transition from any diagnostic session other than the defaultSession to the defaultSession ([DslInternal_SetSecurityLevel\(\)](#)) (initiated by [UDS](#) Service DiagnosticSessionControl (0x10) or S3Server timeout) is performed.]

Only one security level can be active at a time.

[SWS_Dcm_01329] [On every security level change the [Dcm](#) shall update the ModeDeclarationGroup [DcmSecurityAccess](#) with the new security level.]

[SWS_Dcm_CONSTR_06083] Dependency on [DcmDspSecurityAttemptCounterEnabled](#)

Upstream requirements: [RS_Diag_04005](#)

[If [DcmDspSecurityNumAttDelay](#) is not configured, the [DcmDspSecurityAttemptCounterEnabled](#) on the same [DcmDspSecurityRow](#) shall be set to FALSE.]

[SWS_Dcm_CONSTR_06101] [[DcmDspSecurityResetAttemptCounterOnTimeout](#) shall be present only if the [DcmDspSecurityAttemptCounterEnabled](#) for [DcmDspSecurityRow](#) is set to TRUE.]

7.2.4.10.1 Initialization sequence

[SWS_Dcm_01154] [At initialization, for each [DcmDspSecurityRow](#) entry for which the [DcmDspSecurityAttemptCounterEnabled](#) configuration parameter is set to TRUE, the corresponding [Xxx_GetSecurityAttemptCounter](#) shall be called in order to get the value of the AttemptCounter for each of these [DcmDspSecurityRow](#) entries.]

[SWS_Dcm_01156] [If [Xxx_GetSecurityAttemptCounter](#) has returned E_NOT_OK the attempt counter shall be set to the value configured in [DcmDspSecurityNumAttDelay](#) of the according SecurityLevel.]

[SWS_Dcm_01351] [If any [Xxx_GetSecurityAttemptCounter](#) operation returns a DCM_E_PENDING value, the [Dcm](#) shall interrupt calling the [Xxx_GetSecurityAttemptCounter\(\)](#) in order to resume this chain of calls within the next [Dcm_MainFunction\(\)](#) cycle.]

Note: this may be the case when these values are stored within some specific non-volatile memory.

[SWS_Dcm_CONSTR_06076] Dependency for [DcmDspSecurityGetAttemptCounterFnc](#) [[DcmDspSecurityGetAttemptCounterFnc](#) shall be present only if [DcmDspSecurityUsePort](#) is set to [USE_ASYNC_FNC](#) and [DcmDspSecurityAttemptCounterEnabled](#) is set to TRUE.]

[SWS_Dcm_01352] [If the delay after the first call of the [Dcm_MainFunction\(\)](#) which is configured in [DcmDspSecurityMaxAttemptCounterReadoutTime](#) has been reached and all the [Xxx_GetSecurityAttemptCounter](#) have not been called yet (i.e. one operation has returned a [DCM_E_PENDING](#) status in the previous [Dcm_MainFunction\(\)](#) cycle), the pending operation shall be cancelled by a call with the [OpStatus](#) set to [DCM_CANCEL](#).]

[SWS_Dcm_01353] [In the conditions of [\[SWS_Dcm_01352\]](#), the [AttemptCounters](#) of remaining security levels (which have not been obtained via the calls to their [Xxx_GetSecurityAttemptCounter](#)) shall be initialized with the value configured in [DcmDspSecurityNumAttDelay](#) of the according [SecurityLevel](#).]

[SWS_Dcm_01354] [While not all [Xxx_GetSecurityAttemptCounter](#) operations have returned a final status and the operation chain has not been cancelled, the conditionsNotCorrect (0x22) NRC shall be returned to any [SecurityAccess](#) (0x27) request-Seed subfunction request.]

[SWS_Dcm_01355] [Once all the [AttemptCounter](#) values have been successfully or unsuccessfully retrieved (all the [Xxx_GetSecurityAttemptCounter\(\)](#) operations have been executed and have returned a final, non-PENDING error value or the operation chain has been cancelled), if at least one of the restored [AttemptCounter](#) values is greater than or equal to the [DcmDspSecurityNumAttDelay](#) configured for its corresponding [DcmDspSecurityRow](#), the [Dcm](#) shall start the [SecurityDelayTimer](#) with the higher value of [DcmDspSecurityDelayTimeOnBoot](#) / [DcmDspSecurityDelayTime](#) of the according [DcmDspSecurityRow](#).]

[SWS_Dcm_01356] [A timer ([DcmDspSecurityDelayTime](#), [DcmDspSecurityMaxAttemptCounterReadoutTime](#)) which is configured with 0 shall be considered to have timed out instantaneously when it is started, i.e. shall have no delay effect.]

[SWS_Dcm_CONSTR_06074] Dependency for [DcmDspSecurityMaxAttemptCounterReadoutTime](#) [[DcmDspSecurityMaxAttemptCounterReadoutTime](#) shall be a multiple and at minimum equal to [DcmTaskTime](#).]

7.2.4.10.2 AttemptCounter update

[SWS_Dcm_01357] [A successful [sendKey](#) subfunction request shall reset that security level's specific [AttemptCounter](#).]

[SWS_Dcm_01599] [If `DcmDspSecurityResetAttemptCounterOnTimeout` is set to TRUE and `SecurityDelayTimer` expires, the `Dcm` shall reset that security level's specific `AttemptCounter`.]

[SWS_Dcm_01155] [The `Dcm` shall call `Xxx_SetSecurityAttemptCounter()` (in case the configuration parameter `DcmDspSecurityAttemptCounterEnabled` for the according `DcmDspSecurityRow` is set to TRUE) when the `Dcm` has changed the attempt counter to inform the application about the counter change.]

[SWS_Dcm_CONSTR_06078] **Dependency for `DcmDspSecuritySetAttemptCounterFnc`** [`DcmDspSecuritySetAttemptCounterFnc` shall be present only if `DcmDspSecurityUsePort` is set to `USE_ASYNC_FNC` and the `DcmDspSecurityAttemptCounterEnabled` set to TRUE.]

7.2.4.11 Manage session state

[SWS_Dcm_00022]

Upstream requirements: [RS_Diag_04006](#)

[The `DSL` submodule shall save the state of the current active session.]

For accessing this variable, the `DSL` submodule provides interfaces to:

- get the current active session: `Dcm_GetSesCtrlType`
- set a new session: `DslInternal_SetSesCtrlType()`

[SWS_Dcm_00034]

Upstream requirements: [SRS_BSW_00101](#)

[During `Dcm` initialization, the session state is set to the value 0x01 ("DefaultSession").]

[SWS_Dcm_01062] [The call to `Dcm_ResetToDefaultSession` allows the application to reset the current session to Default session and invokes the mode switch of the `ModeDeclarationGroupPrototype DcmDiagnosticSessionControl` by calling `SchM_Switch_<bsnp>_DcmDiagnosticSessionControl(RTE_MODE_DcmDiagnosticSessionControl_DCM_DEFAULT_SESSION)`.]

Example: Automatic termination of an extended diagnostic session upon exceeding of a speed limit.

7.2.4.12 Manage authentication state

The `Dcm` provides means for authenticated diagnostics. The `DSL` sub-module provides an authentication state per diagnostic connection. It initializes this state upon startup and takes care about fallback into non-authenticated states if the connection is idle for some time.

[SWS_Dcm_01477] Authentication state per connection

Upstream requirements: [RS_Diag_04230](#)

[The Dcm shall provide an authentication state per configured DcmDslConnection.]

[SWS_Dcm_01478] Mode declaration group per connection

Upstream requirements: [RS_Diag_04230](#)

[The Dcm shall provide the authentication state of each connection via the ModeDeclarationGroupPrototype DcmAuthenticationState_<ConnectionName>.]

The Dcm maintains an authentication state and mirrors this state to the mode declaration group DcmAuthentication_<ConnectionName>. This mode declaration group is intended to be changed only by the Dcm, however applications changing this state have no influence on the Dcm authentication state.

[SWS_Dcm_01479] Authentication states

Upstream requirements: [RS_Diag_04230](#)

[The Dcm shall support per connection the two authentication states:]

- deauthenticated
- authenticated

Upon startup, the Dcm is in deauthenticated state or restores the persisted state. A transition to authenticated state can only be done after the client successfully executed the authentication sequence. In some use cases as in production, a frequent power-on/power off sequence is performed. To keep the achieved authentication state over the power off, there is a dedicated mode rule requesting the Dcm to persist the authenticated state.

[SWS_Dcm_01480] Initialization of authentication state

Upstream requirements: [RS_Diag_04230](#)

[If [DcmDspAuthenticationPersistStateModeRuleRef](#) is not configured or the mode rule referenced by [DcmDspAuthenticationPersistStateModeRuleRef](#) is evaluated to false, the Dcm shall initialize within [Dcm_Init](#) all authentication states to deauthenticated state.]

[SWS_Dcm_01481] Initialization of persisted authentication states

Upstream requirements: [RS_Diag_04230](#)

[If the mode rule referenced by [DcmDspAuthenticationPersistStateModeRuleRef](#) is evaluated to true, the Dcm shall initialize the persisted authentication state including role and white list on each connection.]

Transitions between authenticated states are controlled by both DSL and DSP sub-modules. The DSL sub-module is in charge for fallback of authenticated state into deauthenticated state. The DSP sub-module is in charge for transition changes triggered from a client by diagnostic services.

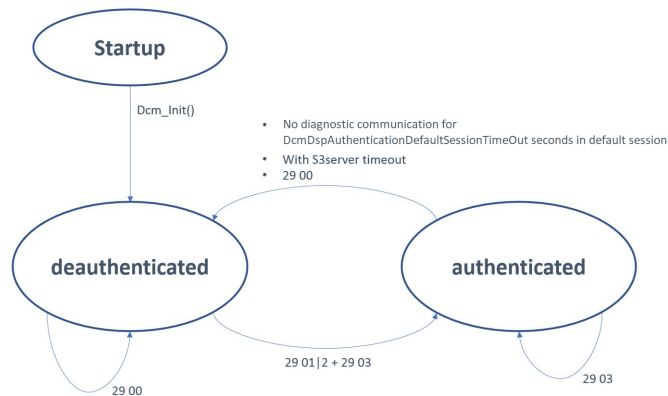


Figure 7.7: Authenticated state transitions without persistent states

[SWS_Dcm_01482] Fallback to deauthenticated state on idle connection

Upstream requirements: [RS_Diag_04230](#)

[The Dcm shall make a transition from authenticated into deauthenticated state for a configured connection if the following conditions apply:

- The Dcm was in default session when the last diagnostic response was sent on that connection and
- `DcmDspAuthenticationDefaultSessionTimeout` is configured and no diagnostic request was received on that connection for `DcmDspAuthenticationDefaultSessionTimeout` seconds after the last `Dcm_TpTxConfirmation` on that connection.

]

[SWS_Dcm_01483] Fallback to deauthenticated session on S3server timeout

Upstream requirements: [RS_Diag_04230](#)

[If the Dcm is in a non-default session and a S3server timeout occurs, the Dcm shall perform a transition from authenticated into deauthenticated state on the authentication state assigned to that connection which was in a non-default session.]

[SWS_Dcm_01484] Clearing persisted authentication state

Upstream requirements: [RS_Diag_04230](#)

[If the authentication state of a connection performs a transition to deauthenticated state, the Dcm shall clear all persisted authentication information on that connection.]

[SWS_Dcm_01485] Reaction of fallback into deauthenticated state

Upstream requirements: [RS_Diag_04230](#)

[Upon a transition from authenticated into deauthenticated state, the Dcm shall discard the current role, white list and use the configured deauthentication role from `DcmDspAuthenticationDeauthenticatedRoleRef`.]

In some use cases, it is desirable that the application set the role instead of using a diagnostic service with its potentially time-consuming certificate parsing. The Dcm provides the API [Dcm_SetDeauthenticatedRole](#) to overwrite the configured deauthentication role. The overwritten role is only valid in deauthenticated state will not be persisted and is overwritten by a role provided by certificates via service 0x29.

[SWS_Dcm_01486] Default authentication role set from SWC

Upstream requirements: [RS_Diag_04230](#)

[If a connection is in deauthenticated state and the API [Dcm_SetDeauthenticatedRole](#) is called, the Dcm shall use the provided deauthenticatedRole as new role per deauthenticated state for this connection.]

[SWS_Dcm_01487] Setting deauthenticated role by SWC only in deauthenticated state

Upstream requirements: [RS_Diag_04230](#)

[The Dcm shall process a call of [Dcm_SetDeauthenticatedRole](#) only if the connection is in deauthenticated state.]

[SWS_Dcm_01488] Lifetime of deauthenticated role by SWC

Upstream requirements: [RS_Diag_04230](#)

[A deauthenticated role set by [Dcm_SetDeauthenticatedRole](#) is discarded when that connection performs a transition to authenticated state.]

[SWS_Dcm_01489] No persistency for deauthenticated roles by SWC

Upstream requirements: [RS_Diag_04230](#)

[If the authentication state is not persisted, at startup the Dcm shall always be in the deauthenticated state and use the default role configured in [DcmDspAuthenticationDeauthenticatedRoleRef](#).]

7.2.4.13 Non-default sessions

The [Dcm](#) supports the ISO14229-1 [1] defined sessions. There are two kind of sessions, the default session and the non-default sessions. In a non-default session the S3 timer is used to detect inactive testers and if the timer elapses the default session is entered.

[SWS_Dcm_01668] S3Server session timer support

Upstream requirements: [RS_Diag_04249](#)

[The [Dcm](#) shall support the S3Server and start and stop this timer according to ISO14229-2 [3].]

[SWS_Dcm_01669] Sessions and connections

Upstream requirements: [RS_Diag_04249](#)

[The [Dcm](#) shall start or stop the S3Server according to ISO14229-2 [3] but only for the client on the [DcmDslConnection](#) that requested the non-default session.]

[SWS_Dcm_01670] Change to default session upon S3Server elapses

Upstream requirements: [RS_Diag_04249](#)

[If the S3Server elapses, the [Dcm](#) shall switch back to default session.]

7.2.4.14 Allow to modify timings

[SWS_Dcm_00027]

Upstream requirements: [RS_Diag_04015](#), [RS_Diag_04249](#)

[The [Dcm](#) module shall handle the following protocol timing parameters in compliance with ISO14229-2 [3]: P2ServerMin, P2ServerMax, P2*ServerMin, P2*ServerMax, S3Server]

[SWS_Dcm_00143]

Upstream requirements: [RS_Diag_04015](#), [RS_Diag_04249](#)

[P2min / P2*min shall be set to defined values: P2min = 0ms, P2*min = 0ms.]

[SWS_Dcm_01679] Default S3 timeout value

Upstream requirements: [RS_Diag_04015](#), [RS_Diag_04249](#)

[If the parameter [DcmS3ServerTimeoutOverwrite](#) is not configured, [Dcm](#) shall take the value of 5000ms as the S3Server timeout.]

[SWS_Dcm_01680] Overwritten S3 timeout value [If the parameter [DcmS3ServerTimeoutOverwrite](#) is configured, the [Dcm](#) shall take the configured value as the S3Server timeout.]

These protocol timing parameters have influence on the session layer timing (no influence on Transport Layer timing). Some of these timing parameters can be modified while protocol is active with the following means:

- [UDS](#) Service DiagnosticSessionControl (0x10)
- [UDS](#) Service AccessTimingParameter (0x83)

The [DSL](#) submodule provides the following functionalities to modify the timing parameters:

- Provide the active timing parameters,
- Set the new timing parameters. Activation of new timing values is only allowed after sending the response.

7.2.4.14.1 Different service tables

For the different protocols a different set of allowed diagnostic services is valid (e.g. the [UDS](#) commands for the enhanced diagnosis, the [OBD](#) mode services for the [OBD](#) protocol). It is possible to create different service tables and link them to the diagnostic protocol.

[SWS_Dcm_00035]

Upstream requirements: [SRS_BSW_00101](#)

[With every protocol initialization, the [DSL](#) submodule sets a link to the corresponding service table (see configuration parameter [DcmDslProtocolSIDTable](#)).]

The [DSD](#) submodule uses this link for further processing of diagnostic requests.

7.2.4.14.2 Prioritization of protocol

The configuration parameter [DcmDslProtocolPriority](#) makes it possible to give each protocol its own relative priority. Possible use case: There are ECUs, communicating with a vehicle-internal diagnostic tester (running on enhanced diagnosis) and a vehicle-external OBD-II/WWH-OBD tester. The OBD-II/WWH-OBD communication must have a higher priority than the enhanced diagnosis.

[SWS_Dcm_00015]

Upstream requirements: [RS_Diag_04021](#)

[A protocol with higher priority is allowed to preempt the already running protocol.]

Differentiation of diagnostic protocols is possible, because of different [DcmDslProtocolRxPduId](#) values (configured per protocol, see configuration parameter [DcmDslProtocolRxPduRef](#)) referenced in the protocol configuration.

7.2.4.14.3 Preemption of protocol

[SWS_Dcm_00079]

Upstream requirements: [RS_Diag_04021](#)

[If a protocol is preempted and this protocol has a running pending response transmission, the Dcm shall call [PduR_DcmCancelTransmit\(\)](#) for this transmission with the following parameters: PduId: the id of the Pdu to be canceled]

[SWS_Dcm_00460]

Upstream requirements: [RS_Diag_04021](#)

[When [PduR_DcmCancelTransmit\(\)](#) returns [E_NOT_OK](#), the [Dcm](#) module shall stop the current protocol.]

[SWS_Dcm_01046]

Upstream requirements: [RS_Diag_04021](#)

[If a running diagnostic request is preempted by a higher priority request, the [Dcm](#) shall cancel all external pending operations on the preempted protocol with `Dcm_OpStatus` set to `DCM_CANCEL`.]

[SWS_Dcm_01047] [In case an operation to the [Dem](#) is pending and the new request also requires an interaction with the Dem, the [Dcm](#) shall accept the new request and call the corresponding [Dem API](#) with the parameters from the new request.]

[SWS_Dcm_00575]

Upstream requirements: [RS_Diag_04021](#)

[If the [Dcm](#) is preempting a protocol with a pending reception, the [Dcm](#) module shall call `cancel` that reception with `PduR_DcmCancelReceive()`.]

[SWS_Dcm_00576]

Upstream requirements: [RS_Diag_04021](#)

[If `PduR_DcmCancelReceive ()` returns `E_NOT_OK`, the [Dcm](#) shall stop the current protocol.]

[SWS_Dcm_00727]

Upstream requirements: [RS_Diag_04021](#)

[If a diagnostic request cannot be processed due to a higher priority protocol and [DcmDslDiagRespOnSecondDeclinedRequest](#) is set to `True`, the [Dcm](#) shall send `NRC 0x21` (`BusyRepeatRequest`) for the not processed request.]

[SWS_Dcm_01605]

Upstream requirements: [RS_Diag_04021](#)

[If a diagnostic cannot be processed due to a higher priority protocol and [DcmDslDiagRespOnSecondDeclinedRequest](#) is set to `False`, the [Dcm](#) shall ignore the request. In this case no response message at all is generated.]

[SWS_Dcm_00729] [In case of multiple clients with different `PduIDs` which are requesting the same protocol, as all the connections of the same protocol are having the same priority, a second request (with the different `RxPduId`) will not be processed. If the configuration parameter [DcmDslDiagRespOnSecondDeclinedRequest](#) is `TRUE`, a negative response with `NRC 0x21` (`BusyRepeatRequest`) shall be issued for the second request. If the configuration parameter is `FALSE`, no response shall be issued.]

Note:

- A multitude of `RxPduIDs` may be configured per [DcmDslProtocol](#)
- These `RxPduIDs` may be themselves connected to different Testers via the `PduR` configuration

- This means that many Testers may be configured for the same Protocol
- And this represents a non-UDS extension/use case. In order to have a UDS-compliant flow, there should be one [DcmDslProtocol](#) instance per Tester.

[SWS_Dcm_01050] [In case of diagnostic parallel requests, with same / lower priority than the active request then the ComM APIs (ComM_DCM_ActiveDiagnostic, ComM_DCM_InactiveDiagnostic) shall not be called.]

7.2.4.14.4 Parallel diagnostic protocol processing

Multiple testers are a common scenario in today's vehicles. In order to reduce the interference between concurrent tester requests to a minimum the [Dcm](#) supports parallel diagnostic service processing. This behavior is according to recommended practice of ISO 14229-1 Appendix J. There are certain restrictions, that in non-default session only diagnostic communication from one tester is allowed. In default session and for OBD-II communication it is possible to process diagnostic requests in parallel. Parallel OBD and UDS communication is particularly important if vehicles are equipped with so called 'OBD dongles' or with electronic logging devices. These devices are installed by the vehicle owner and do diagnostic communication over standardized [OBD](#) services. The presence of such devices shall interfere as little as possible with vehicle internal [UDS](#) communication. Therefore, whenever it is possible, the [Dcm](#) supports parallel processing.

[SWS_Dcm_01602] Processing of parallel requests in default session

Upstream requirements: [RS_Diag_04021](#)

[If the [Dcm](#) receives a request and no further protocol with a higher priority is currently in a non-default session, the [Dcm](#) shall accept the new incoming request and process it.]

[SWS_Dcm_01603] No parallel processing in non-default session

Upstream requirements: [RS_Diag_04021](#)

[If the [Dcm](#) receives a request and a further protocol with a higher priority is currently in a non-default session, the [Dcm](#) shall decline the new received request according to [\[SWS_Dcm_00727\]](#).]

Some [Dcm](#) interfaces provide access to different diagnostic services, e.g. interface RoutineService for subfunctions Start, Stop and Request Result of a RoutineControl (0x31) or the interface DataServices for the Read and Write operations. On these interfaces only a single client shall access to the data at any point in time.

[SWS_Dcm_01604] Delay parallel processing on the same application interface change

Upstream requirements: [RS_Diag_04021](#)

[If the [Dcm](#) receives a request and the service processing of this request requires a call to the same application interface that is currently processing another request, the [Dcm](#) shall delay the call to the interface until the running operation on that application interface has finished.]

If the [Dcm](#) delays the service processing due to [\[SWS_Dcm_01604\]](#) the standard timing behavior with P2 and NRC 0x78 apply. From an outside perspective, the delayed call to the application looks like that the application itself is taking more time for execution.

[SWS_Dcm_01367]

Upstream requirements: [RS_Diag_04163](#)

[The [Dcm](#) shall process incoming OBD-II requests in parallel to a running [UDS](#) request. In this case the protocol priority check according to [\[SWS_Dcm_00015\]](#) is skipped and no protocol pre-emption is done.]

With the container [DcmDslProtocolRow](#), the [Dcm](#) configuration supports multiple protocols. Each protocol has a configured [DcmDemClientRef](#) defining the [Dem](#) client interacting with the [Dem](#). This client Id allows the [Dem](#) to distinguish between concurrent calls of the [Dcm](#) of the same function or set of functions to process a certain request.

[SWS_Dcm_01369]

Upstream requirements: [RS_Diag_04162](#)

[While processing a diagnostic request received from a given protocol, the [Dcm](#) shall determine the [DcmDemClientRef](#) of the [DcmDslProtocolRow](#) of the processed protocol. The [Dcm](#) shall use this value in all [Dem](#) API calls that have a ClientId as parameter.]

[SWS_Dcm_01370] Serialization of multiple calls to the same interface

Upstream requirements: [RS_Diag_04162](#)

[The [Dcm](#) shall internally serialize all asynchronous C/S interface or C function calls to the same port interface or C function during parallel diagnostic services processing and return a pending to the re-entrant caller.]

[SWS_Dcm_01371]

Upstream requirements: [RS_Diag_04162](#)

[If the [Dcm](#) receives a request on a higher priority protocol than the currently processed request and a diagnostic service in a non-default session is currently processed, the [Dcm](#) shall cancel the running diagnostic request, make a transition into default session and process the new received request.]

Integrators will assign OBD protocols the highest priority to meet the legislated response and timing requirements. Therefore, all definitions of 'higher priority protocols' apply to the use case where OBD is used.

[SWS_Dcm_01372]

Upstream requirements: RS_Diag_04162

[If the Dcm processes a request from a high priority protocol in default session and the Dcm is receiving a diagnostic request to change in a non-default session, the Dcm shall delay the session change request until the high priority protocol service is finished according to [SWS_Dcm_01371] and make a transition into the requested non-default session.]

[SWS_Dcm_CONSTR_06102] Limitation to one single OBD protocol [The Dcm shall support only one DcmDslProtocolRow with a configured DcmDslProtocolType set to DCM_OBD_ON_<XYZ>.]

[SWS_Dcm_CONSTR_06103] OBD protocol shall have highest priority [The Dcm shall support a DcmDslProtocolRow with DcmDslProtocolType set to DCM_OBD_ON_<XYZ> as the highest priority.]

[SWS_Dcm_CONSTR_06000] Harmonize the naming between interfaces and modes [The shortname of DcmDspSessionRow shall match names of Dcm_SesCtrlType and of the mode declarations of DcmDiagnosticSessionControl. The "DCM_" prefix is mandatory for all shortnames.]

[SWS_Dcm_CONSTR_06001] Provide standardized names for ISO standardized diagnostic sessions [The following values of DcmDspSessionLevel which represent ISO defined diagnostic sessions shall be used for the shortname of DcmDspSessionRow:

- 1 DCM_DEFAULT_SESSION
 - 2 DCM_PROGRAMMING_SESSION
 - 3 DCM_EXTENDED_DIAGNOSTIC_SESSION
 - 4 DCM_SAFETY_SYSTEM_DIAGNOSTIC_SESSION
-]

7.2.4.15 Manage resources

Due to limited resources, the following points should be considered as hints for the design:

- It is allowed to use and allocate only one diagnostic buffer in the Dcm module. This buffer is then used for processing the diagnostic requests and responses.
- Output of NRC 0x78 (Response pending) responses is done with a separate buffer.
- paged-buffer handling (see [SWS_Dcm_00028]).

7.2.4.16 Communication Mode Handling

Communication Mode Handling is an interface between [Dcm](#) and ComM. The ComM informs the [Dcm](#) about the current communication state of a channel. The [Dcm](#) is calling the ComM about active Diagnostic which shall prevent an Ecu shutdown/sleep.

The status `ActiveDiagnostic` shows if diagnostic requests shall keep the ECU awake (`ActiveDiagnostic == 'DCM_COMM_ACTIVE'`) or if diagnostic requests shall not prevent an Ecu shutdown/sleep (`ActiveDiagnostic == 'DCM_COMM_NOT_ACTIVE'`). Application can change the status `ActiveDiagnostic` regarding to system conditions.

[SWS_Dcm_CONSTR_06027] [The application will inform the [Dcm](#) by calling `Xxx_SetActiveDiagnostic()` about the `ActiveDiagnostic` status.]

[SWS_Dcm_01069] [After [Dcm_Init](#), the [Dcm](#) shall set `ActiveDiagnostic` to `'DCM_COMM_ACTIVE'`.]

[SWS_Dcm_01070] [If `Xxx_SetActiveDiagnostic()` is called with `'false'` the [Dcm](#) set `ActiveDiagnostic` to `'DCM_COMM_NOT_ACTIVE'`.]

[SWS_Dcm_01071] [If `Xxx_SetActiveDiagnostic()` is called with `'true'` the [Dcm](#) set `ActiveDiagnostic` to `'DCM_COMM_ACTIVE'`.]

[SWS_Dcm_01142] [The [Dcm](#) shall wait the Full Communication mode indication from the ComM (call to [Dcm_ComM_FullComModeEntered](#)) before initiating the transmission of the diagnostic answer. The time to wait should be no longer than the `P2ServerMax` calculated from the moment the request was received.]

[SWS_Dcm_01143] [If the [Dcm](#) fails to confirm a response pending transmission (`DCM_E_FORCE_RCRRP`) due to [\[SWS_Dcm_01142\]](#), the [Dcm](#) shall trigger the `Det` error `DCM_E_FORCE_RCRRP_IN_SILENT_COMM`.]

Note : On the reception side a silent communication mode can lead to the lost of the request in case of segmented transmission.

7.2.4.16.1 No Communication

The ComM module will indicate the No Communication Mode to the [Dcm](#) module by calling [Dcm_ComM_NoComModeEntered](#). In response, the [Dcm](#) will immediately disable all transmissions (see the definition of [Dcm_ComM_NoComModeEntered](#) for details).

[SWS_Dcm_00148] [[Dcm_ComM_NoComModeEntered](#) shall disable all kinds of transmissions (receive and transmit) of communication. This means that the message reception and also the message transmission shall be off.]

[SWS_Dcm_00149] [`Dcm_ComM_NoComModeEntered` shall disable the ResponseOnEvent transmissions.]

[SWS_Dcm_00150] [`Dcm_ComM_NoComModeEntered` shall disable the periodicId transmissions (`ReadDataByPeriodicIdentifier`).]

[SWS_Dcm_00151] [`Dcm_ComM_NoComModeEntered` shall disable normal transmissions.]

[SWS_Dcm_00152] [After `Dcm_ComM_NoComModeEntered` has been called, the `Dcm` module shall not call the function `PduR_DcmTransmit()`.]

[SWS_Dcm_01324] [In case `Dcm_ComM_NoComModeEntered` is called with a `NetworkId` for a ComM channel not referenced within the `Dcm` (see configuration parameter `DcmDslProtocolComMChannelRef`), the `Dcm` shall return without performing any further action.]

7.2.4.16.2 Silent Communication

The ComM module will indicate the Silent Communication Mode to the `Dcm` module by calling `Dcm_ComM_SilentComModeEntered`. In response, the `Dcm` will immediately disable all transmissions (see the definition of `Dcm_ComM_SilentComModeEntered` for details).

[SWS_Dcm_00153] [`Dcm_ComM_SilentComModeEntered` shall disable all transmission. This means that the message transmission shall be off.]

[SWS_Dcm_00154] [`Dcm_ComM_SilentComModeEntered` shall disable the ResponseOnEvent transmissions.]

[SWS_Dcm_00155] [`Dcm_ComM_SilentComModeEntered` shall disable the periodicId transmissions (`ReadDataByPeriodicIdentifier`) shall be disabled.]

[SWS_Dcm_00156] [`Dcm_ComM_SilentComModeEntered` shall disable the normal transmissions.]

[SWS_Dcm_01325] [In case `Dcm_ComM_SilentComModeEntered` is called with a `NetworkId` for a ComM channel not referenced within the `Dcm` (see configuration parameter `DcmDslProtocolComMChannelRef`), the `Dcm` shall return without performing any further action.]

7.2.4.16.3 Full Communication

The ComM module will indicate the Full Communication Mode to the Dcm module by calling `Dcm_ComM_FullComModeEntered`. In response, the Dcm will enable all transmissions (see the definition of `Dcm_ComM_FullComModeEntered` for details).

[SWS_Dcm_00157] [`Dcm_ComM_FullComModeEntered` shall enable all kind of communication. This means that the message reception and also the message transmission shall be on.]

[SWS_Dcm_00159] [`Dcm_ComM_FullComModeEntered` shall enable the ResponseOnEvent transmissions.]

[SWS_Dcm_00160] [`Dcm_ComM_FullComModeEntered` shall enable the periodicId transmissions (`ReadDataByPeriodicIdentifier`).]

[SWS_Dcm_00161] [`Dcm_ComM_FullComModeEntered` shall enable the normal transmissions.]

[SWS_Dcm_01326] [In case `Dcm_ComM_FullComModeEntered` is called with a NetworkId for a ComM channel not referenced within the Dcm (see configuration parameter `DcmDslProtocolComMChannelRef`), the Dcm shall return without performing any further action.]

7.2.4.16.4 Diagnostic Activation State

The Dcm notifies the ComM module about the internal diagnostic state for all networks. There are two options for the diagnostic state on a network. In 'active' diagnostic state, the Dcm is processing one or more diagnostic requests from this network or the Dcm is in a non-default session. In 'inactive' diagnostic state, the Dcm is in default session and is not processing a diagnostic request on that network.

When a network has no communication in progress, the Dcm will set the diagnostic activation state to 'inactive'. When there is a diagnostic communication on a network the Dcm sets the diagnostic state to 'active'. In any non-default session, the diagnostic state remains in state 'active'. The communication state can also be controlled by the API `Xxx_SetActiveDiagnostic` according to [SWS_Dcm_01070] and [SWS_Dcm_01071].

[SWS_Dcm_01373]

Upstream requirements: [RS_Diag_04006](#)

[The Dcm shall go into 'active' diagnostic state on a network, if a diagnostic request is received on a network or the diagnostic session is changed to any non-default session.]

[SWS_Dcm_01374]

Upstream requirements: [RS_Diag_04006](#)

[The [Dcm](#) shall go into 'inactive' diagnostic state on a network when the current diagnostic request processing is finished and the [Dcm](#) is not processing a diagnostic request of another protocol on this network and if the [Dcm](#) is in default session.]

[SWS_Dcm_01375]

Upstream requirements: [RS_Diag_04006](#)

[The [Dcm](#) shall go into 'inactive' diagnostic state on all networks if a S3Server timeout occurs and the [Dcm](#) makes a transition into default session.]

[SWS_Dcm_01376]

Upstream requirements: [RS_Diag_04006](#)

[If ActiveDiagnostic is 'DCM_COMM_ACTIVE' and the [Dcm](#) is doing a transition into 'active' diagnostic state of a diagnostic protocol, the [Dcm](#) shall call ComM_DCM_ActiveDiagnostic(NetworkId), with the networkId associated to the received Pdu (see [DcmDslProtocolComMChannelRef](#)), with every request, to inform the ComM module about the need to stay in Full Communication Mode.]

[SWS_Dcm_01377]

Upstream requirements: [RS_Diag_04006](#)

[Upon a diagnostic state transition into 'inactive', the [Dcm](#) shall notify the ComM module about an inactive diagnostic state on a network by calling ComM_DCM_InactiveDiagnostic(NetworkId), with the networkId associated to the received Pdu (see [DcmDslProtocolComMChannelRef](#)).]

[SWS_Dcm_01378]

Upstream requirements: [RS_Diag_04006](#)

[The definition of a finished diagnostic request according to [\[SWS_Dcm_01374\]](#), shall be as follows:

- the [Dcm](#) has sent a positive or negative response unequal to NRC 0x78 by receiving the [Dcm_TpTxConfirmation](#) connected to the response given by the [DSL](#) submodule
- the [Dcm](#) has processed the service with SPRMIB=true and the positive response was suppressed
- in case of functional addressing, the [Dcm](#) has processed the service and the negative response was suppressed.

]

7.3 Diagnostic Service Dispatcher (DSD)

7.3.1 Introduction

The [DSD](#) submodule is responsible to check the validity of an incoming diagnostic request (Verification of Diagnostic Session/Security Access levels/Application permission) and keeps track of the progress of a service request execution.

[SWS_Dcm_00178] [The [DSD](#) submodule shall only process valid requests and shall reject invalid ones.]

7.3.2 Use cases

The following use cases are relevant and are described in detail in the following:

- Receive a request message and transmit a positive response message
- Receive a request message and suppress a positive response
- Receive a request message and suppress a negative response
- Receive a request message and transmit a negative response message
- Send a positive response message without corresponding request
- Segmented Responses

7.3.2.1 Receive a request message and transmit a positive response message

This is the standard use case of normal communication ("ping-pong"). The server receives a diagnostic request message. The [DSD](#) submodule ensures the validity of the request message. In this use case, the request is valid and the response will be positive. The request will be forwarded to the appropriate data processor in the [DSP](#) submodule. When the data processor has finished all actions of data processing, it triggers the transmission of the response message by the [DSD](#) submodule.

If the data processor processes the service immediately as part of the request indication function, the data processor can trigger the transmission inside this indication function ("synchronous"). If the processing takes a longer time (e. g. waiting on EEPROM driver), the data processor defers some processing ("asynchronous"). The response pending mechanism is covered by the [DSL](#) submodule. The data processor triggers the transmission explicitly, but from within the data processor's context.

As soon as a request message is received, the corresponding DcmPduld is blocked by the [DSL](#) submodule (see [\[SWS_Dcm_00241\]](#)). During the processing of this request, no other request of the same protocol type (e.g. an enhanced session can be ended by a [OBD](#) session) can be received, until the corresponding response message is sent and the DcmPduld is released again.

7.3.2.2 Receive a request message and suppress a positive response

This is a sub-use-case of the previous one. Within the UDS protocol it is possible to suppress the positive response by setting a special bit in the request message (see [SWS_Dcm_00200]). This special suppression handling is completely performed within the DSD submodule.

7.3.2.3 Receive a request message and suppress a negative response

In case of functional addressing the DSD submodule shall suppress the negative response for NRC 0x11, 0x12, 0x31, 0x7E and 0x7F (see [SWS_Dcm_00001]).

7.3.2.4 Receive a request message and transmit a negative response message

There are a many different reasons why a request message is rejected and a negative response is to be sent. If a diagnostic request is not valid or if a request may not be executed in the current session, the DSD submodule will reject the processing and a negative response will be returned.

But there are even many reasons to reject the execution of a well-formed request message, e.g. if the ECU or system state does not allow the execution. In this case, the DSP submodule will trigger a negative response including an NRC supplying additional information why this request was rejected.

In case of a request composed of several parameters (e.g. a UDS Service Read-DataByIdentifier (0x22) request with more than one identifier to read), each parameter is treated separately. And each of these parameters can return an error. This kind of request returns a positive response if at least one of the parameters was processed successfully.

[SWS_Dcm_00827] [The DSD sub-module shall check the received diagnostic request in the order given by ISO14229-1 [1]. If one of the computations failed the Dcm shall stop the execution of the NRC check sequence then stop or do not start the execution of the received diagnostic request and finally transmit the NRC for which the computation failed.]

7.3.2.5 Send a positive response message without corresponding request

There are two services within the UDS protocol, where multiple responses are sent for only one request. In general, one service is used to enable (and disable) an event- or time-triggered transmission of another service, which again is sent by the ECU without a corresponding request (see ISO14229-1 [1]). These services are:

- **UDS** Service ReadDataByPeriodicIdentifier (0x2A). This service allows the client to request the periodic transmission of data record values from the server identified by one or more periodicDataIdentifiers.

Type 2 = UUDT message on a separate DcmTxPduld.

- ResponseOnEvent (0x86). This service requests a server to start or stop transmission of responses on a specified event.

Type 1 = USDT messages on the DcmTxPduld already used for normal diagnostic responses,

Type 2 = USDT messages on separate DcmTxPduld.

For Type 1, the outgoing messages must be synchronized with "normal outgoing messages", which have a higher priority.

This handling is especially controlled by the **DSL** submodule. However, the **DSD** submodule also provides the possibility to generate a response without a corresponding request.

7.3.2.6 Segmented Responses (paged-buffer)

Within the diagnostic protocol, some services allow to exchange a significant amount of data, e.g. **UDS** Service ReadDTCInformation (0x19) and **UDS** Service TransferData (0x36).

In the conventional approach, the ECU internal buffer must be large enough to keep the longest data message which is to be exchanged (worst-case) and the complete buffer is filled before the transmission is started.

RAM memory in an ECU often is a critical resource, especially in smaller micros. In a more memory-saving approach, the buffer is filled only partly, transmitted partly and then refilled partly - and so on. This paging mechanism requires only a significantly reduced amount of memory, but demands a well-defined reaction time for buffer refilling.

The user can decide whether to use the "linear buffer" or paged-buffer for diagnostics.

7.3.3 Interaction of the **DSD** with other modules

The **DSD** submodule is called by the **DSL** submodule when receiving a diagnostic message and performs the following operations:

- delegates processing of request to the **DSP** submodule or external modules outside the **Dcm**
- keeps track of request processing (Return the status on <Module>_<DiagnosticService>() and <Module>_<DiagnosticService>_<SubService>() APIs call or "Service Interpreter calls")

- transmits the response of the Application to the **DSL** submodule (Transmit functionality)

7.3.3.1 Interaction of the **DSD** with the **DSL** main functionality

Direction	Explanation
Bidirectional	Exchange of the Diagnostic Messages (receive/transmit).
DSD submodule to DSL submodule	Obtain latest diagnostic session and latest security level.
DSL submodule to DSD submodule	Confirmation of transmission of Diagnostic Message.

Table 7.1: Interaction between the **DSD submodule and the **DSL** submodule**

7.3.3.2 Interaction of the **DSD** with the **DSP**

Direction	Explanation
DSD submodule to DSP submodule	- Delegate processing of request. - Confirmation of transmission of Diagnostic Message.
DSP submodule to DSD submodule	- Signal that processing is finished.

Table 7.2: Interaction of the **DSD with the **DSP****

7.3.4 Functional Description of the **DSD**

7.3.4.1 Support checking the diagnostic service identifier and adapting the diagnostic message

The **DSD** submodule shall be triggered by the **DSL** submodule if a new diagnostic message is recognized. The **DSD** submodule will start processing by analyzing the diagnostic service identifier contained in the received diagnostic message.

[SWS_Dcm_00084] [If configured (configuration parameter **DcmRespondAllRequest**=FALSE), if the **Dcm** module receives a diagnostic request that contains a service **ID** that is in the range from 0x40 to 0x7F or in the range from 0xC0 to 0xFF, the **Dcm** shall not respond to such a request.]

This range corresponds to the diagnostic response identifier.

[SWS_Dcm_00192] [The **DSD** submodule shall analyze the (incoming) diagnostic message for the diagnostic service identifier (based on first byte of the diagnostic message) and shall check the supported services with the newly received diagnostic service identifier.]

[SWS_Dcm_00193] [During this check, the **DSD** submodule shall search the newly received diagnostic service identifier in the "Service Identifier Table".]

For performance reasons it might be necessary that the support check is done with a "lookup table" functionality. In this "Service Identifier Table" all supported Service IDs of the ECU are predefined.

[SWS_Dcm_00195] [The **DSL** submodule shall provide the current "Service Identifier Table"]

Rationale for **[SWS_Dcm_00195]**: The "Service Identifier Table" and the information about the supported services will be generated out of the configuration. More than one Service Identifier Table can be configured for selection. At one time only one Service Identifier Table can be active.

[SWS_Dcm_00196] [For the check, the **DSD** submodule shall scan the active "Service Identifier Table" for a newly received diagnostic service identifier. If this service identifier is supported and if the configuration parameter **DcmDsdSidTabFnc** (see ECUC_Dcm_00777) is not empty, the **DSD** submodule shall call the configured service interface (<Module>_<DiagnosticService>). If the configuration parameter is empty, the **Dcm** shall call the internally implemented service interface.]

The diagnostic service identifier is not supported when it is not included in the "Service Identifier Table".

[SWS_Dcm_00197] [If the newly received diagnostic service identifier is not supported, the **DSD** submodule shall transmit a negative response with **NRC** 0x11 (Service not supported) to the **DSL** submodule.]

[SWS_Dcm_00198] [The **DSD** submodule shall store the newly received diagnostic service identifier for later use.]

For example: WriteDataByIdentifier (for writing VIN number):

1. A new diagnostic message is received by the **DSL** submodule. (Diagnostic Message WriteDataByIdentifier = 0x2E, 0xF1, 0x90, 0x57, 0x30, 0x4C, 0x30, 0x30, 0x30, 0x30, 0x34, 0x33, 0x4D, 0x42, 0x35, 0x34, 0x31, 0x33, 0x32, 0x36)
2. The **DSL** submodule indicates a new diagnostic message with the "Data Indication" functionality to the **DSD** submodule. In the diagnostic message buffer the diagnostic message is stored (buffer = 0x2E,0xF1,0x90,...).
3. The **DSD** submodule executes a check of the supported services with the determined service identifier (first byte of buffer 0x2E) on the incoming diagnostic message.
4. The incoming diagnostic message is stored in the **Dcm** variable **Dcm_MsgContextType**.

[SWS_Dcm_CONSTR_06047] [Id of the Service identifier configured in [DcmDsdSidTabServiceId](#) shall be unique within one [DcmDsdServiceTable](#).]

[SWS_Dcm_00732] [For the first call of <Module>_<DiagnosticService> the opStatus shall be set to DCM_INITIAL.]

[SWS_Dcm_00733] [The [Dcm](#) shall not accept further requests (on same or lower priority) while <Module>_<DiagnosticService>() returns DCM_E_PENDING. Dcm-internal timeout handling (based on RCR-RP limitation) may lead to a cancellation of the external diagnostic service processing.]

[SWS_Dcm_00735] [In case of cancellation the [API](#) <Module>_<DiagnosticService> is called again with the parameter opStatus set to DCM_CANCEL.]

7.3.4.2 Handling of "suppressPosRspMsgIndicationBit"

The "suppressPosRspMsgIndicationBit" is part of the subfunction parameter structure (Bit 7 based on second byte of the diagnostic message, see ISO14229-1 [1] Section 6.5: Server response implementation rules).

[SWS_Dcm_00200]

Upstream requirements: [RS_Diag_04020](#)

[If the "suppressPosRspMsgIndicationBit" is TRUE, the [DSD](#) submodule shall NOT send a positive response message.]

[SWS_Dcm_00201] [The [DSD](#) submodule shall remove the "suppressPosRspMsgIndicationBit" (by masking the Bit) from the diagnostic message.]

[SWS_Dcm_00202] [The [Dcm](#) module shall transport the information on a suppression of a positive response being active (between the layers) via the parameter [Dcm_MsgContextType](#).]

[SWS_Dcm_00203] [In case of responsePending the [Dcm](#) module shall clear the "suppressPosRspMsgIndicationBit."]

Rationale for [\[SWS_Dcm_00203\]](#): In the described case the final response (negative/-positive) is required.

[SWS_Dcm_00204] [The [Dcm](#) module shall only perform the "suppressPosRspMsgIndicationBit" handling when the configuration parameter [DcmDsdSidTabSubfuncAvail](#) is set for the newly received service identifier]

Note: The "suppressPosRspMsgIndicationBit" handling needs to be considered independent of the processing order in the request (like for RoutineControl service).

Rationale for [\[SWS_Dcm_00204\]](#): The "suppressPosRspMsgIndicationBit" is only available if a service has a subfunction.

7.3.4.3 Verification functionality

Prior of execution of a received diagnostic service, the DSD performs a set of verifications. The DSD will only accept a service, if all verifications are successfully passed.

[SWS_Dcm_01535] DSD verifications prior of service execution

Upstream requirements: [RS_Diag_04230](#), [RS_Diag_04005](#), [RS_Diag_04006](#)

[The Dcm shall only accept a diagnostic request, if the following verifications have been passed in the following order:]

1. Verification of Manufacturer permission (Call of the manufacturer interface indication operation)
2. Verification of the SID
3. Verification of the service access control on the current authentication state
4. Verification of the Diagnostic Session
5. Verification of the Service Security Access levels
6. Verification of the Supplier permission (Call of the Supplier interface indication operation)
7. Verification of the Mode rules for service IDs.

[SWS_Dcm_01474]

Upstream requirements: [RS_Diag_04235](#)

[In case the DSD generates a [NRC](#), the Dcm shall only call XXX_Confirmation.]

This means that the Dcm will not call DsplInternal_DcmConfirmation().

7.3.4.3.1 Verification of the diagnostic service access rights

The UDS service Authentication (0x29) is used to change the authentication state of a diagnostic connection and to provide the access rights. Depending on the reached role and provided white list a dynamic set of diagnostic services is available for the tester on that connection. The DSD submodule verifies on service ID (SID) and sub-function (SF) level, if a service can be executed or not.

[SWS_Dcm_01536] Authentication on UDS services only

Upstream requirements: [RS_Diag_04230](#)

[The Dcm shall only verify the authentication for UDS services. A UDS service has a service ID within the range of 0x10 and 0xFF.]

OBD services are explicitly excluded from authentication checks. By legislation the OBD services need to be always available, independent from active authentication

state. If WWH-OBD is used the system engineer must ensure that these services are always accessible.

[SWS_Dcm_01537] Verifying access rights

Upstream requirements: [RS_Diag_04230](#)

[The Dcm shall only verify and check the configured access rights of a diagnostic service, if the container [DcmDspAuthentication](#) is configured.]

If no [DcmDspAuthentication](#) is configured, the Dcm will process all diagnostic services as if the current connection would grant access to execute the current processed service. Checking the access rights for diagnostic services is done at different levels of the service structure. The use of diagnostic service access rights introduces means to allow or to refuse a diagnostic service due to current roles and authentication states. Some services shall always be allowed to be executed, like the service 0x29 (Authentication) to set the current tester access rights. This service and other OEM or supplier specific services should have granted access independent from the authentication state. To realize this, the Dcm uses a default role that is used in all deauthenticated states. In that state, all role based verifications are done as in authenticated state. The active role is provided by the configuration.

[SWS_Dcm_01538] Access rights for services in deauthenticated state

Upstream requirements: [RS_Diag_04230](#)

[If the current connection is in deauthenticated state, the [Dcm](#) shall use the role configured by [DcmDspAuthenticationDeauthenticatedRoleRef](#) as current role for all role based access verification checks.]

[SWS_Dcm_01539] Definition of allowed service execution

Upstream requirements: [RS_Diag_04233](#)

[The Dcm shall allow the service execution, if a role verification was successful or the service is allowed by the white list.]

[SWS_Dcm_01540] Diagnostic service execution rights verification

Upstream requirements: [RS_Diag_04233](#)

[The Dcm shall check if a service execution is permitted in the current authentication check or not. The Dcm shall perform the following checks in the given order below. If a check grants access to a service, the remaining checks are skipped:

1. Checks on service ID level
2. Checks on service ID and sub-function level
3. Checks for services with one or multiple DIDs
4. Check on dynamically defined DIDs
5. Checks on service 0x31 per sub-function
6. Checks on service 0x19 parameter MemorySelection

7. Checks on service 0x14 parameter MemorySelection

]

[SWS_Dcm_01541] Service ID authentication check for UDS service requests*Upstream requirements:* [RS_Diag_04233](#)

[Upon processing a diagnostic service, the Dcm shall grant access to the diagnostic service if:

1. for that service a service role is configured via [DcmDsdServiceRoleRef](#) and the verification according to [\[SWS_Dcm_01522\]](#) was successful or
2. the active white list on that connection has one entry for a SID (1-byte element) which matches that service.

]

[SWS_Dcm_01542] Service with sub-function authentication check for UDS service requests*Upstream requirements:* [RS_Diag_04230](#)

[Upon processing a diagnostic service with sub-function, the Dcm shall grant access to the diagnostic service if:

1. for that service and subfunction a subservice service role is configured via [DcmDsdSubServiceRoleRef](#) and the verification according to [\[SWS_Dcm_01522\]](#) was successful or
2. the active white list on that connection has one entry for a SID with sub-function (2-byte element) that matches that service and sub-function.

]

[SWS_Dcm_01562] White list verification for services with 3 and 4 bytes*Upstream requirements:* [RS_Diag_04233](#)

[For 3 and 4 bytes white list for services entries, the Dcm shall verify on the full length of the configured white list service element. The service is granted access if the first bytes of the received request match the entire white list entry.]

Verification of byte 3 and 4 within the Dsd is beyond the scope of a typical Dsd operation. It provides means to extend the capabilities of white list service verifications and gives means to adapt to legacy authentication solutions.

[SWS_Dcm_01544] Response behavior of services without access rights*Upstream requirements:* [RS_Diag_04230](#)

[If the service execution verification fails due to a failed check in scope of [\[SWS_Dcm_01540\]](#), the Dcm shall send a NRC 0x34 authenticationRequired and stop the service processing.]

7.3.4.3.2 Verification of the Diagnostic Session

The [UDS](#) Service DiagnosticSessionControl (0x10) is used to enable different diagnostic sessions in the ECU (e.g. Default session, Extended session). A diagnostic session enables a specific set of diagnostic services and/or functionality in the ECU. It furthermore enables a protocol-depending data set of timing parameters applicable to the started session.

On receiving a service request, the [DSD](#) module will obtain the current Diagnostic Session with [Dcm_GetSesCtrlType](#) and will verify whether the execution of the requested service (NOT the [UDS](#) Service DiagnosticSessionControl (0x10)) and sub-service is allowed in the current diagnostic session or not.

Note that the handling of the [UDS](#) Service DiagnosticSessionControl (0x10) itself is not part of the [DSD](#) submodule.

[SWS_Dcm_00211] [If the newly received diagnostic service is not allowed in the current Diagnostic Session (according to the configuration parameter [DcmDsdSidTabSessionLevelRef](#)), the [DSD](#) submodule shall transmit a negative response with [NRC](#) 0x7F (serviceNotSupportedInActiveSession) to the [DSL](#) submodule.]

[SWS_Dcm_00616] [If the newly received diagnostic service is allowed in the current Diagnostic Session (see [\[SWS_Dcm_00211\]](#)), but the requested subservice is not allowed in the current Diagnostic Session (according to the configuration parameter [DcmDsdSubServiceSessionLevelRef](#)), the [DSD](#) submodule shall transmit a negative response with [NRC](#) 0x7E (subFunctionNotSupportedInActiveSession) to the [DSL](#) submodule.]

7.3.4.3.3 Verification of the Service Security Access levels

The purpose of the Security Access level handling is to provide a possibility to access data and/or diagnostic services, which have restricted access for security, emissions, or safety reasons. The [DSD](#) submodule shall perform this handling with the [UDS](#) Service SecurityAccess (0x27). The [DSD](#) submodule will perform a verification whether the execution of the requested service (NOT the [UDS](#) Service SecurityAccess (0x27)) is allowed in the current Security level by asking for the current security level, using the [DSL](#) function [Dcm_GetSecurityLevel](#).

The management of the security level is not part of the [DSD](#) submodule.

Note: For some use cases (e.g. [UDS](#) Service ReadDataByIdentifier (0x22), where some DataIdentifier can be secure) it will be necessary for the Application to call also the function [Dcm_GetSecurityLevel](#).

[SWS_Dcm_00217] [If the newly received diagnostic service is not allowed in the current Security level (according to the configuration parameter [DcmDsdSidTabSecurityLevelRef](#)), the [DSD](#) submodule shall transmit a negative response with [NRC](#) 0x33 (Security access denied) to the [DSL](#) submodule.]

[SWS_Dcm_00617] [If the newly received diagnostic service is allowed in the current Security level (see [\[SWS_Dcm_00217\]](#)), but the requested subservice is not allowed in the current Security level (according to the configuration parameter [DcmDsdSubServiceSecurityLevelRef](#)), the [DSD](#) submodule shall transmit a negative response with [NRC](#) 0x33 (Security access denied) to the [DSL](#) submodule.]

7.3.4.3.4 Verification of the Service mode dependencies

[SWS_Dcm_00773] [If the newly received diagnostic service is not allowed in the current mode condition (according to the configuration parameter [DcmDsdSidTabModeRuleRef](#)), the [DSD](#) submodule shall transmit the calculated negative response of the referenced [DcmModeRule](#) to the [DSL](#) submodule.]

[SWS_Dcm_00774] [If the newly received diagnostic service is allowed in the current mode condition [\[SWS_Dcm_00773\]](#), but the requested subservice is not allowed in the current mode condition (according to the configuration parameter [DcmDsdSubServiceModeRuleRef](#)), the [DSD](#) submodule shall transmit the calculated negative response of the referenced [DcmModeRule](#) to the [DSL](#) submodule.]

7.3.4.4 Check format and subfunction support

The [DSD](#) submodule checks whether a specific subfunction is supported before executing the requested command.

[SWS_Dcm_00273] General sub-function supported NRC check [The [DSD](#) shall send the negative response [NRC](#) 0x12 (sub-functionNotSupported), if for the processed service no configured [DcmDsdSubService](#) exists with the [DcmDsdSubServiceId](#) of the processed service. This [NRC](#) check shall not be done for [UDS](#) Service 0x31 (RoutineControl).]

The [DSD](#) submodule will check for the minimum message length before executing the requested command.

[SWS_Dcm_00696] [The [DSD](#) submodule shall trigger a negative response with [NRC](#) 0x13 (Incorrect message length or invalid format), if the length of the request is inferior to the minimum length of the request.]

[SWS_Dcm_01411] [If [DcmDsdSubService](#) is configured for a [DcmDsdService](#), the [Dcm](#) shall support the sub-function configured in [DcmDsdSubServiceId](#) with [SPRMIB](#) set to 0 or 1.]

7.3.4.4.1 Verification of the Manufacturer Application environment/permission

The purpose of this functionality is that, just after receiving the diagnostic request, the Manufacturer Application is requested to check permission/environment.

E.g. in after-run ECU state, it might be not allowed to process [OBD](#) requests.

[SWS_Dcm_00218] [If container [DcmDsdServiceRequestManufacturerNotification](#) exists, the [DSD](#) submodule shall call the operation [Xxx_Indication](#) on all configured [ServiceRequestIndication](#) ports (see configuration parameter [DcmDsdServiceRequestManufacturerNotification](#)).]

[SWS_Dcm_00462] [If at least a single [Xxx_Indication](#) function called according to [\[SWS_Dcm_00218\]](#) returns [E_REQUEST_NOT_ACCEPTED](#), the [DSD](#) submodule shall give no response.]

[SWS_Dcm_01172] [In case of [\[SWS_Dcm_00462\]](#), the [DSD](#) shall only call [Xxx_Confirmation](#) but not [DspInternal_DcmConfirmation](#).]

[SWS_Dcm_00463] [If at least a single [Xxx_Indication](#) function called according to [\[SWS_Dcm_00218\]](#) has returned [E_NOT_OK](#) and no function has returned [E_REQUEST_NOT_ACCEPTED](#), the [DSD](#) submodule shall trigger a negative response with [NRC](#) from the [ErrorCode](#) parameter.]

[SWS_Dcm_01321]

Upstream requirements: [RS_Diag_04011](#)

[If more than one [Xxx_Indication](#) function called, according to [\[SWS_Dcm_00218\]](#), has returned [E_NOT_OK](#) and no function has returned [E_REQUEST_NOT_ACCEPTED](#), the [DSD](#) submodule shall trigger a negative response using the [ErrorCode](#) parameter from the first [Xxx_Indication](#) returning [E_NOT_OK](#).]

7.3.4.4.2 Verification of the Supplier Application environment/permission

The purpose of this functionality is that, right before processing the diagnostic message, the Supplier Application is requested to check permission/environment.

E.g. in after-run ECU state, it might be not allowed to process [OBD](#) requests.

[SWS_Dcm_00516] [If container [DcmDsdServiceRequestSupplierNotification](#) exists, the [DSD](#) submodule shall call the operation [Xxx_Indication](#) on all

configured ServiceRequestIndication ports (see configuration parameter [DcmDsdServiceRequestSupplierNotification](#)).]

[SWS_Dcm_00517] [If at least a single [Xxx_Indication](#) function called according to [\[SWS_Dcm_00516\]](#) returns E_REQUEST_NOT_ACCEPTED, the [DSD](#) submodule shall give no response.]

[SWS_Dcm_00518] [If at least a single [Xxx_Indication](#) function called according to [\[SWS_Dcm_00516\]](#) has returned E_NOT_OK and no function has returned E_REQUEST_NOT_ACCEPTED, the [DSD](#) submodule shall trigger a negative response with [NRC](#) from the [ErrorCode](#) parameter.]

[SWS_Dcm_01322]

Upstream requirements: [RS_Diag_04011](#)

[If more than one [Xxx_Indication](#) function called, according to [\[SWS_Dcm_00516\]](#), has returned E_NOT_OK and no function has returned E_REQUEST_NOT_ACCEPTED, the [DSD](#) submodule shall trigger a negative response using the [ErrorCode](#) parameter from the first [Xxx_Indication](#) returning E_NOT_OK.]

7.3.4.5 Distribution of diagnostic message to [DSP](#) submodule

[SWS_Dcm_00221] [The [DSD](#) submodule shall search for the executable functionality of the [DSP](#) submodule for newly received diagnostic service identifier and shall call the corresponding [DSP](#) service interpreter.]

7.3.4.6 Assemble positive or negative response

[SWS_Dcm_00222] [When the [DSP](#) submodule has finished the execution of the requested Diagnostic Service the [DSD](#) submodule shall assemble the response.]

The execution of the [DSP](#) service interpreter can have the results:

- positive Result or
- negative Result.

Following possible Responses can be assembled:

- positive Response,
- negative Response, or
- no Response (in the case of suppression of responses).

7.3.4.6.1 Positive Response

[SWS_Dcm_00223] [The [DSD](#) submodule shall add the response service identifier and the response data stream (returned by the Application) in the parameter "Dcm_MsgContextType".]

[SWS_Dcm_00224] [The [DSD](#) submodule shall therefore transfer the Dcm_MsgContextType into a (response) buffer and shall add the service identifier at the first byte of the buffer.]

[SWS_Dcm_00225] [The [DSD](#) submodule shall execute the "Initiate transmission" functionality in the next execution step.]

7.3.4.6.2 Negative Response

The [DSP](#) submodule can trigger the transmission of a negative response with a specific [NRC](#) to the [DSD](#) submodule. For the allowed [NRC](#) of the executed Service [ID](#) please refer to the specification of the service in ISO14229-1 [1] (Response code parameter definition Table 18 of ISO 14229-1:2020) and ISO15031-5 [2]. The [DSP](#) and the Application have to take care of the correct use of [NRC](#) of the executed Service ID.

[SWS_Dcm_00228] [The [DSD](#) submodule shall handle all [NRCs](#) supported from the Application and defined in Dcm_NegativeResponseCodeType.]

7.3.4.6.3 Suppression of response

[SWS_Dcm_00231] [In the case that the "suppressPosRspMsgIndicationBit" is indicated in the functionality "Handling of suppressPosRspMsgIndicationBit" (stored in the Variable Dcm_MsgContextType (Element: Dcm_MsgAddInfo)), the [DSD](#) submodule shall activate the suppression of Positive Responses.]

[SWS_Dcm_00001]

Upstream requirements: [RS_Diag_04020](#)

[In the case of a Negative Result of the execution and active [Functional Addressing](#) the [DSD](#) submodule shall activate the suppression of the following Negative Responses:

- [NRC](#) 0x11 (Service not supported),
- [NRC](#) 0x12 (SubFunction not supported),
- [NRC](#) 0x31 (Request out of range),
- [NRC](#) 0x7E (Subfunction not supported in active session),
- [NRC](#) 0x7F (Service not supported in active session)

」

7.3.4.7 Initiate transmission

[SWS_Dcm_00232] [The [DSD](#) submodule shall forward the diagnostic (response) message (positive or negative response) to the [DSL](#) submodule.]

[SWS_Dcm_00237] [The [DSL](#) submodule shall forward the diagnostic (response) message (positive or negative response) further to the PduR module by executing a [DSL](#) transmit functionality.]

The [DSL](#) submodule will receive a confirmation by the PduR module upon forwarding the data.

[SWS_Dcm_00235] [The [DSL](#) submodule shall forward the received confirmation from the PduR module to the [DSD](#) submodule.]

[SWS_Dcm_00236] [The [DSD](#) submodule shall forward the confirmation via the internal function `DspInternal_DcmConfirmation()` to the [DSP](#) submodule.]

[SWS_Dcm_00238] [In the case that no diagnostic (response) message shall be sent (Suppression of Responses) the [DSL](#) submodule shall not transmit any response.]

In this case no Data Confirmation is sent from the [DSL](#) submodule to the [DSD](#) submodule but the [DSD](#) submodule will still call internal function `DspInternal_DcmConfirmation()`.

[SWS_Dcm_00240] [In case the request has been fully processed by the Dcm, The [DSD](#) submodule shall finish the processing of one Diagnostic Message of the Diagnostic Service Dispatcher by calling `DspInternal_DcmConfirmation()`.]

Rationale for [\[SWS_Dcm_00240\]](#): The [DSP](#) submodule is waiting for the execution of the `DspInternal_DcmConfirmation()` functionality. So it has to be sent, also when no Data Confirmation is provided. Altogether this means that in any of the following cases:

- Positive Response,
- Negative Response,
- Suppressed Positive Response, and
- Suppressed Negative Response

The [DSD](#) submodule will finish by calling `DspInternal_DcmConfirmation()` (refer to [8.10.3 DspInternal_DcmConfirmation](#)).

[SWS_Dcm_00741] [The [DSD](#) submodule shall call the operation `Xxx_Confirmation()` on all ports using the `ServiceRequestNotification` interface (see configuration parameter `DcmDsdServiceRequestManufacturerNotification` and `DcmDsdServiceRequestSupplierNotification`)]

[SWS_Dcm_00742] [The call of `Xxx_Confirmation()` shall be done right after the call of `DsplInternal_DcmConfirmation()`]

[SWS_Dcm_00677] [If the operation `Indication()` returns value `E_REQUEST_NOT_ACCEPTED`, the [Dcm](#) module shall not send any diagnostic response and shall end the current diagnostic request management.]

[SWS_Dcm_00678] [If the operation `Indication()` returns value `E_NOT_OK`, the [Dcm](#) module shall send a negative response with [NRC](#) value equal to `ErrorCode` parameter value.]

7.4 Diagnostic Service Processing (DSP)

7.4.1 General

When receiving a function call from the [DSD](#) submodule requiring the [DSP](#) submodule to process a diagnostic service request, the [DSP](#) always carries out following basic process steps:

- analyze the received request message,
- check format and whether the addressed subfunction is supported,
- acquire data or execute the required function call on the DEM, SW-Cs or other BSW modules
- assemble the response

The following sections are some general clarifications.

7.4.1.1 Check format and subfunction support

The [DSP](#) submodule will check for appropriate message length and structure before executing the requested command.

[SWS_Dcm_00272] [The [DSP](#) submodule shall trigger a negative response with [NRC](#) `0x13` (Incorrect message length or invalid format), when the analysis of the request message results in formatting or length failure.]

Note: It is up to the implementation in which detail the format check might be executed and depends on the level of detail the diagnostic data description provides at compile time.

7.4.1.2 Assemble response

[SWS_Dcm_00039] [The DSP submodule shall assemble the response message excluding response service identifier and determine the response message length.]

[SWS_Dcm_00038] [If the paged-buffer mechanism is used, the DSP submodule shall determine the overall response length before any data is passed to the DSD submodule or the DSL submodule respectively.]

Requirement [SWS_Dcm_00038] is needed because of segmented diagnostic data transmission on CAN using ISO15765-2 [15], which requires the provision of the overall length of the complete data stream in the very first CAN frame of the respective data transmission (please refer to Section 7.2.4.8 for details about the paged-buffer mechanism).

7.4.1.3 Negative Response Codes handling

[SWS_Dcm_00271] [Unless another particular NRC is specified, the DSP submodule shall trigger a negative response with NRC 0x10 (generalReject), when the API calls made to execute the service do not return OK.]

[SWS_Dcm_01414] Accepted range of Dcm_NegativeResponseCodeType for negative responses [If the Dcm calls an external application by any of the APIs having the out parameter Dcm_NegativeResponseCodeType ErrorCode, the Dcm shall accept only values in the range 0x01-0xFF in case the return value is E_NOT_OK.]

[SWS_Dcm_01415] Behavior on application returning unexpected return code [If the Dcm calls an API with the out parameter Dcm_NegativeResponseCodeType ErrorCode and the application sets this parameter to DCM_POS_RESP and E_NOT_OK is returned, the Dcm shall report the runtime error DCM_E_INVALID_VALUE.]

[SWS_Dcm_00275] [The DSP submodule shall trigger a negative response with NRC 0x31 (Request out of range), when the analysis of the request message results in other unsupported message parameters.]

7.4.1.4 Diagnostic mode declaration groups

The Dcm shall act as a mode manager for the diagnostic modes:

- 1. DcmDiagnosticSessionControl (service 0x10)
- 2. DcmEcuReset (partly service 0x11)
- 3. DcmSecurityAccess (service 0x27)
- 4. DcmModeRapidPowerShutDown (partly service 0x11)

- 5. DcmCommunicationControl_<symbolic name of ComMChannelId>. (service 0x28)
- 6. DcmControlDTCSetting (service 0x85)
- 7. DcmResponseOnEvent_<RoeEventId> (service 0x86)
- 8. DcmAuthenticationState_<Symbolic Name of DcmDslMainConnection>

Note: The RTE/SchM will prefix the names with "MODE_", wherefore the names do not include the MODE keyword.

[SWS_Dcm_01327] [The [Dcm](#) shall define the ModeDeclarationGroupPrototype DcmSecurityAccess as provided-ModeGroup based on the following ModeDeclarationGroup:

```

1 ModeDeclarationGroup DcmSecurityAccess {
2     {
3         DCM_SEC_LEV_LOCKED
4         DCM_SEC_LEV_1
5         ...
6         DCM_SEC_LEV_63
7     }
8     initialMode = DCM_SEC_LEV_LOCKED
9 };

```

]

[SWS_Dcm_01328] [

```

1 ModeSwitchInterface SchM_Switch_<bsnp>_DcmSecurityAccess {
2     isService = true;
3     SecLevel currentMode;
4 };

```

]

[SWS_Dcm_00806] [The [Dcm](#) shall define the ModeDeclarationGroupPrototype DcmDiagnosticSessionControl as provided-ModeGroup based on the ModeDeclarationGroup DcmDiagnosticSessionControl.]

[SWS_Dcm_00777] [The [Dcm](#) shall define the ModeDeclarationGroupPrototype DcmEcuReset as provided-ModeGroup in its Basic Software Module instance based on the ModeDeclarationGroup DcmEcuReset.]

[SWS_Dcm_00807] [The [Dcm](#) shall define the ModeDeclarationGroupPrototype DcmModeRapidPowerShutDown as provided-ModeGroup in its Basic Software Module instance based on the ModeDeclarationGroup DcmModeRapidPowerShutDown.]

[SWS_Dcm_00780] [The [Dcm](#) shall define for each network which is considered in the CommunicationControl service a separate ModeDeclarationGroupPrototype [DcmCommunicationControl_<symbolic name of ComMChannelId>](#) as provided-ModeGroup in its Basic Software Module instance based on the ModeDeclarationGroup [DcmCommunicationControl](#).]

[SWS_Dcm_00781] [The [Dcm](#) shall define the ModeDeclarationGroupPrototype [DcmControlDTCSetting](#) as provided-ModeGroup in its Basic Software Module instance based on the ModeDeclarationGroup [DcmControlDTCSetting](#).]

[SWS_Dcm_00933] [The [Dcm](#) shall define for each [RoeEvent](#) a separate ModeDeclarationGroupPrototype [DcmResponseOnEvent_<Symbolic name of RoeEventId>](#) as provided-ModeGroup in its Basic Software Module instance based on the ModeDeclarationGroup [DcmResponseOnEvent](#).]

The [Dcm](#) provides a state machine for each [RoeEvent](#). The state for a [RoeEvent](#) is needed by SWC to activate event reporting or report the [Roe](#) status to a [Did](#). Therefore the [Dcm](#) provides for each state of each [RoeEvent](#) a ModeDeclarationGroupPrototype which reports the current state of the state machine as mode.

[SWS_Dcm_00934] [The ModeDeclarationGroupPrototype shall represent the current state of the ROE state machine for this [RoeEvent](#).]

7.4.1.5 Environmental condition dependent execution

The execution of a diagnostic service or the acceptance of certificates can be restricted to a mode condition. This enables the [Dcm](#) to formalize environmental checks. For diagnostic service processing, a further check (see [\[SWS_Dcm_00773\]](#) and [\[SWS_Dcm_00774\]](#)) can be configured to the [Dcm](#). This is like session and security checks. The referenced mode rule is arbitrating on to several mode declarations of a mode declaration groups in which the request can be processed. Otherwise a configurable NRC (see [\[SWS_Dcm_00812\]](#)) is responded. The same mode rule checks can be applied on certificate validation. Certificates can be restricted to certain vehicle properties, such as VIN or a certain version number. Only if all the conditions are valid, the certificate is accepted by the [Dcm](#).

[SWS_Dcm_00808] [The [DcmModeRule](#) shall evaluate all referenced [DcmModeCondition](#)s and/or nested [DcmModeRule](#)s either by a logical AND in case [DcmLogicalOperator](#) is set to DCM_AND or by a logical OR in case the [DcmLogicalOperator](#) is set to DCM_OR. In case only a single [DcmModeCondition](#) or [DcmModeRule](#) is referenced the [DcmLogicalOperator](#) shall not be present and therefore not be used.]

[SWS_Dcm_CONSTR_06028] [[DcmModeCondition](#) shall either have a [DcmBswModeRef](#) or a [DcmSwcModeRef](#) or a [DcmSwcSRDataElementRef](#) as external reference.]

[SWS_Dcm_00810] [The [DcmSwcModeRef](#) and [DcmBswModeRef](#) of [DcmModeConditions](#) shall evaluate if the referenced Mode-Declaration is set in case of [DcmConditionType](#) is set to DCM_EQUALS or is not set in case of [DcmConditionType](#) is set to DCM_EQUALS_NOT.]

[SWS_Dcm_01119] Mode condition evaluation [For each mode condition, the Dcm shall compare a compare value with a S/R data element. The compare value is provided by [DcmSwcSRDataElementValueRef](#) or [DcmModeConditionConnectionCertificateCompareElementRef](#) and the S/R Element is by [DcmSwcSRDataElementRef](#). The mode condition is evaluated to true if the S/R data element value is:

- equal to the compare value in case of [DcmConditionType](#) is set to DCM_EQUALS
- unequal to the compare value in case of [DcmConditionType](#) is set to DCM_EQUALS_NOT
- greater than the compare value in case of [DcmConditionType](#) is set to DCM_GREATER_THAN
- greater or equal than the compare value in case of [DcmConditionType](#) is set to DCM_GREATER_OR_EQUAL
- less than the compare value in case of [DcmConditionType](#) is set to DCM_LESS_THAN
- less or equal than the compare value in case of [DcmConditionType](#) is set to DCM_LESS_OR_EQUAL.

]

[SWS_Dcm_CONSTR_06029] [The values DCM_GREATER_THAN, DCM_GREATER_OR_EQUAL, DCM_LESS_OR_EQUAL and DCM_LESS_THAN shall not used with a Mode reference ([DcmBswModeRef](#) or [DcmSwcModeRef](#)) .]

Note: The current mode of the referenced [ModeDeclarationGroupPrototypes](#) could be read by either the [API SchM_Mode](#) (in case of [DcmBswModeRef](#)) or by the [API Rte_Mode](#) (in case of [DcmSwcModeRef](#)).

[SWS_Dcm_00811] [In case multiple [DcmModeConditions](#) are referenced within a [DcmModeRule](#) they shall be evaluated in order of the index attributes of the [EcucReferenceValues](#) for [DcmArgumentRef](#).]

Note: This implies the priority of NRCs

[SWS_Dcm_00782] [If a [DcmModeRule](#) is not referenced from the [DcmDspAuthenticationConnection](#), the Dcm shall use the optional parameter [DcmModeRuleNrcValue](#) as [NegativeResponseCode](#) in case the mode rule is evaluated to false.]

Mode rules for [DcmDspAuthenticationConnection](#) are not part of the NRC evaluation.

[SWS_Dcm_00812] [In case a nested [DcmModeRule](#) contains also a [DcmModeRuleNrcValue](#) parameter, this [NRC](#) shall be used prior the higher-level [NRC](#).]

[SWS_Dcm_00813] [In case [DcmLogicalOperator](#) is set to DCM_AND, the first failed [DcmModeRule](#) with an explicit configured [NRC](#) ([DcmModeRuleNrcValue](#)) shall be used to define the [NRC](#) for the response message.]

[SWS_Dcm_00814] [In case [DcmLogicalOperator](#) is set to DCM_OR, the last failed [DcmModeRule](#) with an explicit configured [NRC](#) ([DcmModeRuleNrcValue](#)) shall be used to define the [NRC](#) for the response message.]

Note: The difference in the AND and OR logical operation is to allow an optimized implementation.

[SWS_Dcm_00815] [In case the complete evaluation result in no specific [NRC](#) the [NRC](#) 0x22 (ConditionsNotCorrect) shall be used.]

[SWS_Dcm_00942] [The [Dcm](#) shall create for commonly used ModeDeclarationGroupPrototype of each [DcmSwcModeRef](#) of [DcmModeConditions](#) a required mode switch port referencing this ModeDeclarationGroupPrototype. The name pattern of this port prototype shall be [DcmModeUser_<ModeDeclarationGroupPrototype>](#)" in case the ModeDeclarationGroupPrototype shortname is unique. Otherwise the name pattern is implementation specific, except the required prefix "[DcmModeUser_](#)".]

Note: ModeDeclarationGroupPrototypes are not necessarily unique, wherefore the exception is required to avoid name clashes in the [Dcm](#) Service-SWC.

Examples on using mode dependent request execution:

General assumptions:

1. [DcmModeRule1](#) consists of [DcmModeCondition1](#), [DcmModeRule2](#) and [DcmModeRule3](#)
2. [DcmModeRule1](#) defines [NRC](#) 0x22
3. [DcmModeRule2](#) and [DcmModeRule3](#) do not have any sub-rules
4. [DcmModeRule2](#) defines [NRC](#) 0x72
5. [DcmModeRule3](#) does not define a [NRC](#) value

Example 1:

- 1) [DcmModeRule1](#) uses an OR combination ([DcmModeCondition1](#) OR [DcmModeRule2](#) OR [DcmModeRule3](#))
 - a) [DcmModeCondition1](#) is failing
→ [NRC](#) 0x22 is returned
 - b) [DcmModeRule2](#) is failing
→ [NRC](#) 0x72 is returned
 - c) [DcmModeRule3](#) is failing
→ [NRC](#) 0x22 is returned

- d) DcmModeCondition1, DcmModeRule2 and DcmModeRule3 are failing
→ [NRC 0x72](#) is returned
- e) DcmModeCondition1 and DcmModeRule3 are failing
→ [NRC 0x22](#) is returned

Example 2:

- 1) DcmModeRule1 uses an AND combination (DcmModeCondition1 AND DcmModeRule2 AND DcmModeRule3)
 - a) DcmModeCondition1 is failing
→ [NRC 0x22](#) is returned
 - b) DcmModeRule2 is failing
→ [NRC 0x72](#) is returned
 - c) DcmModeRule3 is failing
→ [NRC 0x22](#) is returned
 - d) DcmModeCondition1, DcmModeRule2 and DcmModeRule3 are failing
→ [NRC 0x22](#) is returned
 - e) DcmModeCondition1 and DcmModeRule3 are failing
→ [NRC 0x22](#) is returned
 - e) DcmModeRule2 and DcmModeRule3 are failing
→ [NRC 0x72](#) is returned

[SWS_Dcm_CONSTR_06089] Only one compare element

Upstream requirements: [RS_Diag_04232](#)

[In one [DcmModeCondition](#) only one of the elements [DcmSwcSRDataElementRef](#) or [DcmModeConditionCertificateCompareElementRef](#) shall be configured.]

[SWS_Dcm_CONSTR_06090] Use of certificate compare elements [The [DcmModeConditionCertificateCompareElementRef](#) is only allowed, if the parent [DcmModeRule](#) is referenced from a [DcmDspAuthenticationConnection](#).]

7.4.1.6 Sender/Receiver Communication

[SWS_Dcm_00964] [If [DcmDspDiagnosisScaling](#) is present, the [Dcm](#) shall derive the CompuMethod from the [DcmDspDiagnosisScaling](#) container and add it to the DataType in their respective port interface for S/R port of DataServices_Data [\[SWS_Dcm_01035\]](#).]

7.4.1.7 Passing SwDataDefProps properties from DEXT file to the Dcm Service SW-C

UseCase: Pass the SwDataDefProps details like CompuMethod, DataConstraints and Units to the Dcm Service SW-C and make them there available per DID DataElement / per RoutineControl signal. Two alternative work flows are available.

7.4.1.7.1 DcmDspDiagnosticDataElementRef workflow

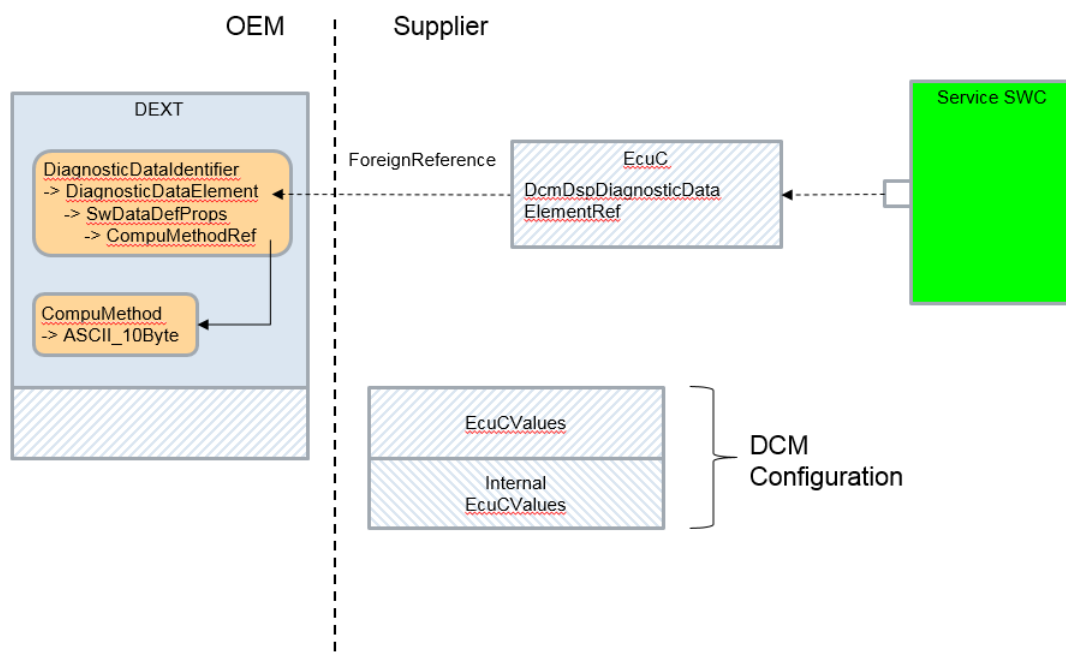


Figure 7.8: DcmDspDiagnosticDataElementRef Workflow

The feature of the `DcmDspDiagnosticDataElementRef` workflow is the use of a `EcucForeignReference` inside the generated EcuC values. While importing the DEXT information, a dedicated EcuC parameter is generated, which holds a `EcucForeignReference` named `DcmDspDiagnosticDataElementRef` to a `DiagnosticDataElement` in the DEXT file. This `EcucForeignReference` enables the access to all SwDataDefProps (BaseType, CompuMethod, DataConstr, etc.) of the corresponding `DiagnosticDataElement`. The container `DcmDspAlternativeDiagnosticDataElement` aggregates this `EcucForeignReference`. In the process step of generating the corresponding Service SWC all needed content will be copied directly based on the `EcucForeignReference` from DEXT to the Service SW-C. In this work flow the existence of the DEXT file while the generation of the Service SW-C is required.

[SWS_Dcm_CONSTR_06053] [The aggregation of `DcmDspTextTableMapping` at `DcmDspAlternativeDataType` is only valid if the category of the `CompuMethod`

of the `DataType` referenced by `DcmDspAlternativeDataType.DcmApplicationDataType` has category set to `TEXTTABLE` or `SCALE_LINEAR_AND_TEXTTABLE`.]

7.4.1.7.2 `DcmDspAlternativeDataType.DcmApplicationDataType` workflow

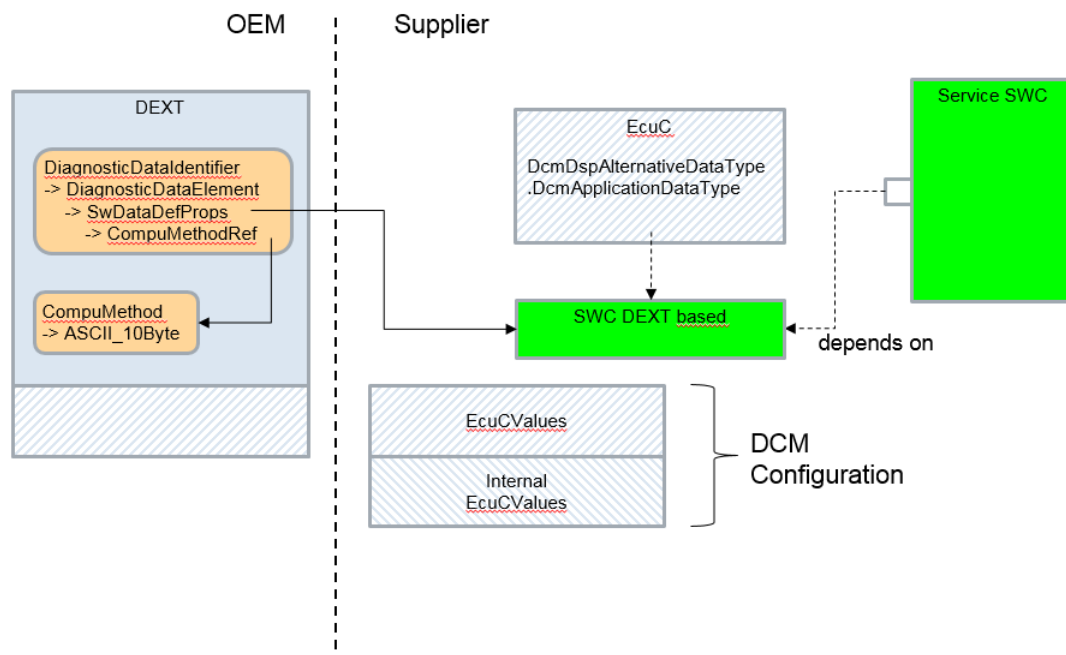


Figure 7.9: `DcmDspAlternativeDataType.DcmApplicationDataType` Workflow

The feature of the `DcmDspAlternativeDataType.DcmApplicationDataType` workflow is that while importing the DEXT information beside the EcuC values also a SW-C fragment is generated. In this SW-C fragment all needed `SwDataDefProps` are directly copied from the DEXT file. Inside the generated EcuC values the EcuC parameter `DcmDspAlternativeDataType.DcmApplicationDataType` refers to the SWC fragment and enables the access to all `SwDataDefProps` (`BaseType`, `CompuMethod`, `DataConstr`, etc.). In the process step of generating the corresponding Service SW-C, all needed content will be included based on the reference from `DcmDspAlternativeDataType.DcmApplicationDataType` to the SW-C fragment. In this work flow the existence of the DEXT file while the generation of the Service SW-C is not required.

7.4.1.8 Asynchronous call behavior

[SWS_Dcm_01412] [If a `Dem` function returns `DEM_PENDING`, the `Dcm` shall call this function again at a later point in time as long as `DEM_PENDING` is returned.]

[SWS_Dcm_00120] [If the number of negative responses for a requested diagnostic tasks (see [SWS_Dcm_00024]) reaches the value defined in the configuration parameter `DcmDslDiagRespMaxNumRespPend`, the `Dcm` module shall stop processing the active diagnostic request, inform the application or BSW (if this diagnostic task implies the call to a `SW-C` interface or a BSW interface) by setting `OpStatus` parameter, of active port interface, to `DCM_CANCEL`, report the runtime error `DCM_E_INTERFACE_TIMEOUT` and shall send a negative response with `NRC 0x10` (General reject).]

[SWS_Dcm_01184] [The `Dcm_SetProgConditions` API shall be called again in the next `Dcm` main function cycle if previous return status was `E_PENDING`.]

[SWS_Dcm_00760] [The return of `DCM_E_PENDING` shall do a re-triggering (e.g. in the next `MainFunction` cycle).]

[SWS_Dcm_01413] [The return values of interfaces called with an `OpStatus` equal to `DCM_CANCEL` shall be ignored.]

7.4.2 UDS Services

[SWS_Dcm_01622] Diagnostic Request Validation

Upstream requirements: `RS_Diag_04196`, `RS_Diag_04203`

[The `Dcm` shall execute the diagnostic request validation, negative response code (`NRC`) determination and processing according to ISO 14229-1 [1].]

[SWS_Dcm_00442] [The `Dcm` module shall implement the services of `UDS` according to [SWS_Dcm_01641].]

[SWS_Dcm_01641] Support of UDS Services [

SID	Service	Subfunction	Supported
0x10	DiagnosticSessionControl		Supported
0x11	ECUReset		Supported
0x14	ClearDiagnosticInformation		Supported
0x19	ReadDTCInformation		Supported
0x22	ReadDataByIdentifier		Supported
0x23	ReadMemoryByAddress		Supported (callout)
0x24	ReadScalingDataByIdentifier		Supported
0x27	SecurityAccess		Supported
0x28	CommunicationControl		Supported
0x29	Authentication		Supported
0x2A	ReadDataByPeriodicIdentifier		Supported
0x2C	DynamicallyDefineDataIdentifier		Supported
0x2E	WriteDataByIdentifier		Supported
0x2F	InputOutputControlByIdentifier		Supported
0x31	RoutineControl		Supported
0x34	RequestDownload		Supported (callout)
0x35	RequestUpload		Supported (callout)
0x36	TransferData		Supported

SID	Service	Subfunction	Supported
0x37	RequestTransferExit		Supported
0x38	RequestFileTransfer		Supported (callout)
0x3D	WriteMemoryByAddress		Supported (callout)
0x3E	TesterPresent		Supported
0x83	AccessTimingParameter		NRC "ServiceNotSupported"
0x84	SecuredDataTransmission		NRC "ServiceNotSupported"
0x85	ControlDTCSetting	On, off	Supported
0x86	ResponseOnEvent		Supported
0x87	LinkControl		User optional

]

7.4.2.1 General behavior using DEM interfaces

Fault memory related diagnostic services are by default processed directly within the [Dcm](#). The [Dcm](#) uses the [Dem](#) module to clear, query, and read diagnostic fault codes, status, and related data. The option to process such a diagnostic via the callout of the DSD sub-module, e.g. [DcmDsdSubServiceFnc](#) for service 0x19, is considered an exceptional use case.

This chapter describes general rules, and how the [Dcm](#) interacts with the [Dem](#) to fulfill the task to process the fault memory related [UDS](#) services.

The [UDS](#) defined [DTC](#) status availability mask is managed by the [Dem](#) module and therefore not configured in the [Dcm](#). When this mask is required the [Dcm](#) uses the [Dem_GetDTCStatusAvailabilityMask](#) API to read this information.

The [UDS](#) defined severity [DTC](#) severity availability mask is also managed by the [Dem](#) and therefore not configured by the [Dcm](#). The [Dcm](#) uses [Dem_GetDTCStatusAvailabilityMask](#) to retrieve this mask information.

To ensure consistent event related data during the reading sequence, the [Dcm](#) module will lock the update of event related data during the read operation. The [Dem](#) provides [Dem_DisableDTCRecordUpdate](#) and [Dem_EnableDTCRecordUpdate](#) to temporarily stop updating the event memory. In particular, this happens during the use of the following [Dem](#) APIs.

- [Dem_SelectExtendedDataRecord\(\)](#)
- [Dem_GetSizeOfExtendedDataRecordSelection\(\)](#)
- [Dem_GetNextExtendedDataRecord\(\)](#)
- [Dem_SelectFreezeFrameData\(\)](#)
- [Dem_GetSizeOfFreezeFrameSelection\(\)](#) and
- [Dem_GetNextFreezeFrameData\(\)](#)

Potentially time consuming API calls in the [Dem](#) are designed asynchronous so that the call can return and the caller can call again at a later time to retrieve the requested data. These APIs have DEM_PENDING as a possible return value. If this return value is detected, the [Dcm](#) will call again with the same parameters. The time of the next call is implementation specific, but mostly this will occur in the next [Dcm_MainFunction](#). Repeated occurring DEM_PENDING as the return value from the [Dem](#) can result in ResponsePending NRC 0x78 sent by the [Dcm](#).

7.4.2.2 Service 0x10 - Diagnostic Session Control

[UDS](#) Service 0x10 allows an external tester to enable different diagnostic sessions in the server. A diagnostic session enables a specific set of diagnostic services and/or functionality in the server. The service request contains the parameter:

- diagnosticSessionType

[SWS_Dcm_00250]

Upstream requirements: [RS_Diag_04006](#)

[The [Dcm](#) module shall implement the [UDS](#) Service 0x10 according to ISO 14229-1:2020 [9]]

[SWS_Dcm_00307] [When responding to [UDS](#) Service 0x10, if the requested subfunction value is not configured in the ECU (configuration parameter [DcmDspSessionLevel](#)), the [DSP](#) submodule shall trigger a negative response with [NRC](#) 0x12 (SubFunction not supported).]

If the requested subfunction value is configured, the following steps are processed even if the requested session type is equal to the already running session type (see ISO14229-1 [1] Section 9.2).

[SWS_Dcm_00311]

Upstream requirements: [RS_Diag_04248](#)

[The send confirmation function shall set the new diagnostic session type with `DslInternal_SetSesCtrlType()` and shall set the new timing parameters (P2ServerMax, P2ServerMax*) (see configuration parameters [DcmDspSessionP2ServerMax](#) and [DcmDspSessionP2StarServerMax](#)) and do the mode switch of the `ModeDeclarationGroupPrototype` [DcmDiagnosticSessionControl](#) by calling `SchM_Switch_<bsnp>_DcmDiagnosticSessionControl()` with the new diagnostic session type (see [\[SWS_Dcm_91019\]](#)).]

[SWS_Dcm_00085] [The [DSP](#) submodule shall manage internally a read access for the dataIdentifier 0xF186 (`ActiveDiagnosticSessionDataIdentifier`) defined in ISO14229-1 [1].]

7.4.2.3 Service 0x11 - ECUReset

UDS Service ECUReset (0x11) allows an external tester to request a server reset. The service request contains parameter:

- resetType

[SWS_Dcm_00260] [The Dcm module shall implement the UDS Service ECUReset (0x11).]

[SWS_Dcm_00373] [On reception of a request for UDS Service 0x11 with the sub functions other than enableRapidPowerShutDown (0x04) or disableRapidPowerShutDown (0x05), the Dcm module shall trigger the mode switch of ModeDeclarationGroup-Prototype DcmEcuReset equal to the received resetType. After the mode switch is requested the Dcm shall trigger the start of the positive response message transmission. Sub function hardReset (0x01) to HARD Sub function keyOffOnReset (0x02) to KEYONOFF Sub function softReset (0x03) to SOFT]

Note: By this mode switch the Dcm informs the BswM to carry out necessary actions for the handling of this individual reset type. These actions can be configured within the BswM action list corresponding to the requested reset type. Here the integrator can also define if an ECU reset will finally be performed or not.

[SWS_Dcm_00594] [On the transmit confirmation (call to Dcm_TpTxConfirmation) of the positive response for the subfunctions other than enableRapidPowerShutDown (0x04) or disableRapidPowerShutDown (0x05), the Dcm module shall trigger the mode switch of ModeDeclarationGroupPrototype DcmEcuReset to the mode EXECUTE (via SchM_Switch_<bsnp>_DcmEcuReset(RTE_MODE_DcmEcuReset_EXECUTE)).c())]

Note: By this mode switch the Dcm requests the BswM to perform the final processing on the reset type according to the configured action list.

[SWS_Dcm_00818] [On reception of a request for UDS Service 0x11 with the sub functions enableRapidPowerShutdown (0x04) or disableRapidPowerShutdown (0x05), the Dcm module shall trigger the mode switch of ModeDeclarationGroupPrototype DcmRapidPowerShutdown: Sub function enableRapidPowerShutdown (0x04) to ENABLE_RAPIDPOWERSHUTDOWN, Sub function disableRapidPowerShutdown (0x05) to DISABLE_RAPIDPOWERSHUTDOWN]

Note: If EnableRapidPowerShutdown is enabled, the ECU should shorten its power-down time.

[SWS_Dcm_00589] [In case the parameter DcmDspPowerDownTime is present, the Dcm shall set the powerDownTime in positive response to sub-service enableRapidPowerShutDown with value set in DcmDspPowerDownTime.]

[SWS_Dcm_00834] [After sending the positive response of EcuReset (call of [DcmTpTxConfirmation](#)) the [Dcm](#) shall ignore all further requests during reset-processing.]

[SWS_Dcm_CONSTR_06080] [DcmDspEcuResetRow](#) container configuration

Upstream requirements: [RS_Diag_04098](#)

[One container [DcmDspEcuResetRow](#) shall be configured for each [DcmDsdSubService](#) ([DcmDspEcuResetId](#) matching to the [DcmDsdSubServiceId](#)) configured for the UDS service ECUReset (0x11) which does not have the corresponding [DcmDsdSubServiceFnc](#) parameter configured.]

7.4.2.4 Service 0x14 - Clear Diagnostic Information

UDS Service ClearDiagnosticInformation (0x14) requests an ECU to clear the error memory. The service request contains the parameter:

- groupOfDTC.

[SWS_Dcm_00247] [The [Dcm](#) module shall implement UDS Service 0x14.]

[SWS_Dcm_01263]

Upstream requirements: [RS_Diag_04058](#)

[Upon reception of a UDS Service ClearDiagnosticInformation (0x14) request with parameter groupOfDTC, the [Dcm](#) module shall call the [API](#) [Dem_SelectDTC\(\)](#) with the following parameter values:

- ClientId: Client Id for this [Dcm](#) instance (see [DcmDemClientRef](#))
- DTC: groupOfDTC from the service request
- DTCFormat: DEM_DTC_FORMAT_UDS
- DTCOrigin: DEM_DTC_ORIGIN_PRIMARY_MEMORY

]

[SWS_Dcm_01400] [After call of [Dem_SelectDTC\(\)](#) the [Dcm](#) shall call [Dem_GetDTCSelectionResultForClearDTC\(\)](#) with the following parameter value:

- ClientId: Client Id for this [Dcm](#) instance (see [DcmDemClientRef](#)).

]

[SWS_Dcm_01265] [In case [Dem_GetDTCSelectionResultForClearDTC\(\)](#) returns DEM_WRONG_DTC, the [Dcm](#) shall send a [NRC](#) 0x31 (RequestOutOfRange).]

[SWS_Dcm_01268] [In case [Dem_GetDTCSelectionResultForClearDTC\(\)](#) returns E_OK, the [Dcm](#) module shall check if application allows to clear the [DTC](#) (according

to the configuration parameter `DcmDspClearDTCCheckFnc`). If not, the `Dcm` module shall send a negative response with `NRC` set to value from the parameter "ErrorCode".]

[SWS_Dcm_01269] [In case application allows to clear the DTC, the `Dcm` module shall check if the `DTC` can be cleared in the current mode condition (according to the configuration parameter `DcmDspClearDTCModeRuleRef`). If not, the `Dcm` module shall send the calculated negative response code of the referenced `DcmModeRule`.]

[SWS_Dcm_00005]

Upstream requirements: [RS_Diag_04058](#)

[If the condition checks are successfully done, the `Dcm` module shall call `Dem_ClearDTC` with the following parameter values:

- `ClientId` = Client Id for this `Dcm` instance (see `DcmDemClientRef`)

]

[SWS_Dcm_00705] [In case `Dem_ClearDTC()` returns `E_OK`, the `Dcm` module shall send a positive response.]

[SWS_Dcm_00707] [In case `Dem_ClearDTC()` returns `DEM_CLEAR_FAILED`, the `Dcm` shall send a negative response `0x22` (`conditionsNotCorrect`).]

[SWS_Dcm_00708] [In case `Dem_ClearDTC()` returns `DEM_WRONG_DTC`, the `Dcm` shall send a negative response `0x31` (`requestOutOfRange`).]

[SWS_Dcm_00966] [In case `Dem_ClearDTC()` returns `DEM_CLEAR_BUSY`, the `Dcm` shall send a negative response `0x22` (`conditionsNotCorrect`).]

[SWS_Dcm_01060] [In case `Dem_ClearDTC()` returns `DEM_CLEAR_MEMORY_ERROR`, the `Dcm` shall trigger a negative response with `NRC` `0x72` (`generalProgrammingFailure`).]

[SWS_Dcm_01408] [In case `Dem_ClearDTC()` returns `DEM_WRONG_DTCORIGIN`, the `Dcm` shall trigger a negative response `0x31` (`requestOutOfRange`).]

7.4.2.5 Service 0x19 - Read `DTC` Information

Service 0x19 allows a client to read the status of server resident Diagnostic Trouble Code (`DTC`) information.

[SWS_Dcm_00248] [The `Dcm` module shall implement the `UDS` Service 0x19.]

To setup the retrieval of specific data from the `Dem` module, the `Dcm` will call different filter `APIs` (`Dem_SetDTCFilter()`, `Dem_SetFreezeFrameRecordFilter()`, `Dem_SelectFreezeFrameData()` and `Dem_SelectExtendedDataRecord()`).

[SWS_Dcm_01043] [In case E_NOT_OK is returned by Dem_SetDTCFilter(), the Dcm module shall send a negative response with NRC 0x31 (requestOutOfRange).]

[SWS_Dcm_01334] [For all sub-functions addressing user defined fault memory, before calling the appropriate Dem API, the Dcm shall add the value 0x0100 to the received selection request parameter MemorySelection in order to match the Dem_DTCOriginType.]

The Dcm service processor for UDS service ReadDTCInformation (0x19) can be used in configurations where SAE J1979-2 is supported, so that reported DTC values of SAE J1979-2 services have different DTC values than used for UDS DTCs.

Some manufacturers don't want to fully change to SAE J1979-2 support with UDS and will only support the J1979-2 functionality on the limited UDS subset defined by J1979-2, wherefore the feature OBD UDS DTC separation was added.

[SWS_Dcm_01618] Use of separated OBD and UDS DTCs

Upstream requirements: RS_Diag_04253

[If DcmDspReadDTCInformationSupportedObdUdsDtcSeparation is set to True, the Dcm service processor for all diagnostic requests according to SAE J1979-2 shall use the Dem API Dem_SelectDTC with the parameter DTCFormat set to DEM_DTC_FORMAT_OBD_3BYTE to query and process DTC related information.]

[SWS_Dcm_01619]

Upstream requirements: RS_Diag_04253

[If DcmDspReadDTCInformationSupportedObdUdsDtcSeparation is set to False, the Dcm service processor for all diagnostic requests according to SAE J1979-2 shall use the Dem API Dem_SelectDTC with the parameter DTCFormat set to DEM_DTC_FORMAT_UDS to query and process DTC related information.]

[SWS_Dcm_01343] [For services with FunctionalGroupIdentifier as parameter in the request, the Dcm shall only process request messages with FunctionalGroupIdentifier equal to 0x33.]

[SWS_Dcm_01344] [For services with FunctionalGroupIdentifier as parameter in the request and FunctionalGroupIdentifier unequal to 0x33, the Dcm shall return NRC 0x31 (RequestOutOfRange).]

7.4.2.5.1 Subfunctions 0x01, 0x07 and 0x12

UDS Service 0x19 with subfunctions 0x01 or 0x12 requests the ECU to report the number of DTCs matching tester-defined criteria. The service request contains the parameter:

- DTCStatusMask

UDS Service 0x19 with subfunction 0x07 requests the ECU to report the number of DTCs matching tester-defined criteria. The service request contains the parameters:

- DTCSeverityMask
- DTCStatusMask

[SWS_Dcm_00376] [When sending a positive response to UDS Service 0x19 with subfunction 0x01, 0x07 or 0x12, the Dcm module shall use the data in the response message according to [\[SWS_Dcm_01642\]](#)]

[SWS_Dcm_01642] Subfunction 0x01, 0x07 and 0x12 response values [

Parameter name	Value
DTCStatusAvailabilityMask	DTCStatusAvailabilityMask
DTCFormatIdentifier	Value returned by Dem_GetTranslationType()
DTCCount	Value calculated according to [SWS_Dcm_00293]

]

[SWS_Dcm_00293]

Upstream requirements: [RS_Diag_04058](#), [RS_Diag_04067](#)

[When responding to UDS Service 0x19 with subfunction 0x01, 0x07 or 0x12, the Dcm module shall calculate the number of DTCs using Dem_GetNumberOfFilteredDTC() after having set the DEM-filter with Dem_SetDTCFilter() using the parameter values according to [\[SWS_Dcm_01643\]](#).]

[SWS_Dcm_01643] Dem_SetDTCFilter() parameters values for subfunctions 0x01, 0x07 and 0x12 [

	reportNumberOfDTC ByStatusMask	reportNumberOfDTC BySeverity- MaskRecord	reportNumberOf EmissionsRelated OBDDTCByStatus- Mask
	0x01	0x07	0x12
ClientId	Client Id for this Dcm instance (see DcmDemClientRef)	Client Id for this Dcm instance (see DcmDemClientRef)	Client Id for this Dcm instance (see DcmDemClientRef)
DTCStatusMask	DTCStatusMask from request	DTCStatusMask from request	DTCStatusMask from request
DTCFormat	DEM_DTC_FORMAT_UDS	DEM_DTC_FORMAT_UDS	DEM_DTC_FORMAT_UDS
DTCOrigin	PRIMARY_MEMORY	PRIMARY_MEMORY	DEM_DTC_ORIGIN_OBD_RELEVANT_MEMORY
FilterWithSeverity	FALSE	TRUE	FALSE
DTCSeverityMask	Not relevant	DTCSeverityMask from request	Not relevant
FilterForFaultDetectionCounter	FALSE	FALSE	FALSE

]

7.4.2.5.2 Subfunctions 0x02, 0x0A, 0x13, 0x15 and 0x17

UDS Service 0x19 with subfunctions 0x02 or 0x13 requests the DTCs (and their associated status) that match certain conditions. The service request contains the parameter:

- DTCStatusMask

UDS Service 0x19 with subfunction 0x0A requests all supported DTCs and their associated status. UDS Service 0x19 with subfunction 0x15 requests all DTCs with permanent status.

[SWS_Dcm_00377] [When sending a positive response to UDS Service 0x19 with subfunction 0x02, 0x0A, 0x13, 0x15 or 0x17, the Dcm module shall use the data in the response message according to [\[SWS_Dcm_01644\]](#).]

[SWS_Dcm_01644] Subfunction 0x02, 0x0A, 0x13, 0x15 and 0x17 response values [

Parameter name	Value
DTCStatusAvailabilityMask	DTCStatusAvailabilityMask
DTCAndStatusRecord	As defined in [SWS_Dcm_00008] and [SWS_Dcm_00378]
MemorySelection (subservice 0x17 only)	From request

]

[SWS_Dcm_01545] Read user defined memory by status mask authentication check

Upstream requirements: [RS_Diag_04233](#)

[On reception of the UDS Service ReadDTCInformation (0x19) with sub-function reportUserDefMemoryDTCByStatusMask (0x17), the Dcm shall check if the access to the selected user defined memory in parameter MemorySelection is authenticated and read the DTC information only if:

- a [DcmDspReadDTCInformationUserDefinedFaultMemoryRoleRef](#) is configured with [DcmDspReadDTCInformationUserDefinedFaultMemoryId](#) matching the MemorySelection and the verification according to [\[SWS_Dcm_01479\]](#) was successful or
- the active white list on that connection has an entry for that requested user defined memory selection one entry.

]

According to [SWS_Dcm_01545] the authentication checks are only executed if DcmDspAuthentication is configured. In case of a failed authentication the NRC handling is according to [SWS_Dcm_01544] and [SWS_Dcm_01551] applies.

[SWS_Dcm_00008] [On reception of a UDS Service 0x19 request with subfunction 0x02 and 0x13 and if the result of the bitwise AND operation between the DTCStatusMask received within the request message and the DTCStatusAvailabilityMask reported by the DEM is equal to 0, the Dcm module shall answer positively with 0 DTC.]

[SWS_Dcm_00378]

Upstream requirements: RS_Diag_04058, RS_Diag_04067

[When responding to UDS Service 0x19 with subfunctions 0x02, 0x0A, 0x13, 0x15 or 0x17, the Dcm module shall obtain the records with DTCs (and their associated status) by repeatedly calling Dem_GetNextFilteredDTC() after having configured the filter with Dem_SetDTCFilter() using the parameter values according to [SWS_Dcm_01645].]

[SWS_Dcm_01645] Dem_SetDTCFilter() parameters values for subfunctions 0x02, 0x0A, 0x13, 0x15 and 0x17 [

	reportDTC ByStatus Mask	report Supported DTCs	report Emissions Related OBDDTC ByStatus Mask	report DTCWith Permanent Status	report UserDef Memory DTCBy StatusMask
	0x02	0x0A	0x13	0x15	0x17
ClientId	Client Id for this Dcm instance (see DcmDem-ClientRef)	Client Id for this Dcm instance (see DcmDem-ClientRef)	Client Id for this Dcm instance (see DcmDem-ClientRef)	Client Id for this Dcm instance (see DcmDem-ClientRef)	Client Id for this Dcm instance (see DcmDem-ClientRef)
DTCStatus Mask	DTCStatus Mask from request	0x00	DTCStatus Mask from request	0x00	DTCStatus Mask from request
DTCFormat	DEM_DTC_FOR-MAT_ UDS	DEM_DTC_FOR-MAT_ UDS	DEM_DTC_FOR-MAT_ UDS	DEM_DTC_FOR-MAT_ UDS	DEM_DTC_FOR-MAT_ UDS
DTCOrigin	PRIMARY_MEMORY	PRIMARY_MEMORY	DEM_DTC_ORIGIN_OBD_RELEVANT_MEMORY	PERMANENT_MEMORY	Memory Selection from request + 0x0100
FilterWith Severity	NO	NO	NO	NO	NO
DTCSeverity Mask	Not relevant	Not relevant	Not relevant	Not relevant	Not relevant
FilterForFault Detection Counter	NO	NO	NO	NO	NO

]

Note:

- The `Dcm` module can get an indication of the number of records that will be found using `Dem_GetNextFilteredDTC()` by using `Dem_GetNumberOfFilteredDTC()`. This allows the implementation to calculate the total size of the response before cycling through the DTCs.
- The value `0x00` used as `DTCStatusMask` for the subfunctions `0x0A` and `0x15` disables the status byte filtering in `Dem_SetDTCFilter()`.

[SWS_Dcm_00828] [In case of paged buffer support is disabled, the `Dcm` module shall not insert zero-padded DTCs to the response of `UDS` Service `0x19` with subfunctions `0x02`, `0x0A`, `0x13`, `0x15` or `0x17`.]

When using paged buffer mechanism, in some case, it's possible that the number of `DTC` matching the filter change between the calculation of the total size, needed for the first page transmission, and the sending of the further pages. For this reason, the following requirement apply :

[SWS_Dcm_00587] [In case of paged buffer support is enabled, The `Dcm` shall limit the response size to the size calculated when sending the first page. If more DTCs match the filter after this sending, the additional DTCs shall not be considered.]

[SWS_Dcm_00588] [In case of paged buffer support is enabled, The `Dcm` shall pad the response with the size calculated when sending the first page. If less `DTC` match the filter after this sending, the missing DTCs shall be padded with `0` value as defined in 15031-6 [16].]

[SWS_Dcm_01229] [If `Dem_GetNextFilteredDTC()` returns `DEM_NO_SUCH_ELEMENT` and at least one matching element could be retrieved before, the `Dcm` shall send a positive response including these data elements.]

[SWS_Dcm_01230] [If `Dem_GetNextFilteredDTC()` returns `DEM_NO_SUCH_ELEMENT` and at no matching element could be retrieved before, the `Dcm` shall send a positive response only for service and subservice and additional parameters required within a positive response.]

7.4.2.5.3 Subfunction 0x08

`UDS` Service `0x19` with subfunction `0x08` requests the DTCs and the associated status that match a tester-defined severity mask record. The service request contains the following parameters:

- `DTCSeverityMask`
- `DTCStatusMask`

[SWS_Dcm_00379] [When sending a positive response to UDS Service 0x19 with subfunction 0x08, the Dcm module shall use the data in the response message according to [SWS_Dcm_01646].]

[SWS_Dcm_01646] Subfunction 0x08 response values [

Parameter name	Value
DTCStatusAvailabilityMask	DTCStatusAvailabilityMask
DTCAndSeverityRecord	As defined in [SWS_Dcm_00380]

]

[SWS_Dcm_00380] [When responding to UDS Service 0x19 with subfunction 0x08, the Dcm module shall obtain the DTCAndSeverityRecords by repeatedly calling Dem_GetNextFilteredDTCAndSeverity() after having configured the filter with Dem_SetDTCFilter() using the parameter values according to [SWS_Dcm_01647].]

[SWS_Dcm_01647] Dem_SetDTCFilter() parameters values for Subfunction 0x08 [

	reportDTCBySeverityMaskRecord
ClientId	Client Id for this Dcm instance (see DcmDemClientRef)
DTCStatusMask	DTCStatusMask from request
DTCFormat	DEM_DTC_FORMAT_UDS
DTCOrigin	PRIMARY_MEMORY
FilterWithSeverity	TRUE
DTCSeverityMask	DTCSeverityMask from request
FilterForFaultDetectionCounter	FALSE

]

Note: The Dcm module can get an indication of the number of records that will be found using Dem_GetNextFilteredDTCAndSeverity() by using Dem_GetNumberOfFilteredDTC().

7.4.2.5.4 Subfunction 0x09

UDS Service 0x19 with subfunction 0x09 requests the severity information of a DTC. The service request contains the parameter:

- DTCMaskRecord

[SWS_Dcm_00381] [When sending a positive response to UDS Service 0x19 with subfunction 0x09, the Dcm module shall use the data in the response message according to [SWS_Dcm_01648].]

[SWS_Dcm_01648] Subfunction 0x09 response values [

Parameter name	Value
DTCStatusAvailabilityMask	DTCStatusAvailabilityMask
DTCAndSeverityRecord	DTCSeverityMask: see [SWS_Dcm_01402] DTCFunctionalUnit: see [SWS_Dcm_01403] DTC: the given DTC of the request statusOfDTC : see [SWS_Dcm_01404]

]

[SWS_Dcm_01402] [To select the DTC, the Dcm module shall call the API Dem_SelectDTC() with the following parameter values:

- ClientId: Client Id for this Dcm instance (see DcmDemClientRef)
- DTC: DTC from the service request
- DTCFormat: DEM_DTC_FORMAT_UDS
- DTCOrigin: DEM_DTC_ORIGIN_PRIMARY_MEMORY

]

[SWS_Dcm_01403] [To retrieve the DTCSeverityMask of the selected DTC, the Dcm shall call Dem_GetSeverityOfDTC() with the following parameter value:

- ClientId: Client Id for this Dcm instance (see DcmDemClientRef)

]

[SWS_Dcm_01404] [To retrieve the DTCFunctionalUnit of the selected DTC, the Dcm shall call Dem_GetFunctionalUnitOfDTC() with the following parameter value:

- ClientId: Client Id for this Dcm instance (see DcmDemClientRef)

]

[SWS_Dcm_01405] [To retrieve the statusOfDTC of the selected DTC, the Dcm shall call Dem_GetStatusOfDTC() with the following parameter value:

- ClientId: Client Id for this Dcm instance (see DcmDemClientRef)

]

[SWS_Dcm_01226] [If Dem_GetFunctionalUnitOfDTC() returns DEM_WRONG_DTC or DEM_WRONG_DTCORIGIN, the Dcm shall send a NRC 0x31 (requestOutOfRange).]

[SWS_Dcm_01240] [If Dem_GetSeverityOfDTC() returns DEM_WRONG_DTC, the Dcm shall send a NRC 0x31 (requestOutOfRange)]

[SWS_Dcm_01406] [If Dem_GetStatusOfDTC() returns DEM_WRONG_DTC or DEM_WRONG_DTCORIGIN, the Dcm shall send a NRC 0x31 (requestOutOfRange).]

7.4.2.5.5 Subfunctions 0x06/0x19

The UDS Service 0x19 with subfunction 0x06 or 0x19 requests a specific Extended Data Records for a specific DTC. The service request contains the parameters:

- DTCMaskRecord
- DTCExtendedDataRecordNumber

[SWS_Dcm_01547] Read user defined memory extended data record authentication check

Upstream requirements: [RS_Diag_04233](#)

[On reception of the UDS Service ReadDTCInformation (0x19) with sub-function reportUserDefMemoryDTCExtDataRecordByDTCNumber (0x19), the Dcm shall check if the access to the selected user defined memory in parameter MemorySelection is authenticated and read the DTC information only if:

- a [DcmDspReadDTCInformationUserDefinedFaultMemoryRoleRef](#) is configured with [DcmDspReadDTCInformationUserDefinedFaultMemoryId](#) matching the MemorySelection and the verification according to [\[SWS_Dcm_01522\]](#) was successful or
- the active white list on that connection has an entry for that requested user defined memory selection.

]

According to [\[SWS_Dcm_01537\]](#) the authentication checks are only executed if DcmDspAuthentication is configured. In case of a failed authentication the NRC handling is according to [\[SWS_Dcm_01485\]](#) and [\[SWS_Dcm_01544\]](#) applies.

[SWS_Dcm_00386] [Upon reception of UDS Service 0x019 with subfunction 0x06 or 0x19, the Dcm shall retrieve from the Dem the stored extended data records for the requested DTC and origin.]

[SWS_Dcm_00295]

Upstream requirements: [RS_Diag_04058](#)

[When responding to UDS Service 0x19 with subfunction 0x06 or 0x19, the Dcm module shall calculate the statusOfDTC by first calling Dem_SelectDTC() with the parameters values set according to [\[SWS_Dcm_01649\]](#) and then Dem_GetStatusOfDTC() with ClientId = Client Id for this Dcm instance (see DcmDemClientRef).]

[SWS_Dcm_01649] Dem_SelectDTC() parameters values for subfunctions 0x06 and 0x19 [

	reportDTCExtendedData RecordByDTCNumber	reportUserDefMemoryDTC ExtDataRecord ByDTCNumber
	0x06	0x19

	reportDTCExtendedDataRecordByDTCNumber	reportUserDefMemoryDTCExtDataRecordByDTCNumber
ClientId	Client Id for this Dcm instance (see DcmDemClientRef)	Client Id for this Dcm instance (see DcmDemClientRef)
DTC	DTCMaskRecord from request	DTCMaskRecord from request
DTCOrigin	PRIMARY_MEMORY	MemorySelection from request + 0x0100

]

[SWS_Dcm_00841] [If `Dem_GetNextExtendedDataRecord()` returns `E_OK` and Buf-Size 0 (empty buffer), the [Dcm](#) module shall omit the `DTCExtendedDataRecordNumber` for the related record in the response of service 0x19 0x06/0x19.]

[SWS_Dcm_00382] [When responding to [UDS](#) Service 0x19 with subfunction 0x06 or 0x19, the [Dcm](#) module shall calculate the `DTCExtendedDataRecord` by first calling `Dem_SelectExtendedDataRecord()` with the parameter values set according to [\[SWS_Dcm_01650\]](#) and then call `Dem_GetNextExtendedDataRecord()` repeatedly until `DEM_NO_SUCH_ELEMENT` is returned.]

[SWS_Dcm_01650] `Dem_SelectExtendedDataRecord()` parameters values for subfunctions 0x06 and 0x19 [

	reportDTCExtendedDataRecordByDTCNumber	reportUserDefMemoryDTCExtDataRecordByDTCNumber
	0x06	0x19
ClientId	Client Id for this Dcm instance (see DcmDemClientRef)	Client Id for this Dcm instance (see DcmDemClientRef)
DTC	DTCMaskRecord from request	DTCMaskRecord from request
DTCOrigin	PRIMARY_MEMORY	Memory Selection from request + 0x0100
ExtendedDataNumber	DTCExtendedDataRecordNumber from request	DTCExtendedDataRecordNumber from request

]

The [Dcm](#) module shall obtain the size of the extended data record by using `Dem_GetSizeOfExtendedDataRecordSelection()`.

7.4.2.5.6 Subfunction 0x03

[UDS](#) Service 0x19 with subfunction 0x03 allows an external tester to request all stored snapshot records for all DTCs in an ECU.

[SWS_Dcm_01640] Support of UDS Service 0x19 with subfunction 0x03

Upstream requirements: [RS_Diag_04250](#)

[The [Dcm](#) shall support UDS Service 0x19 with subfunction 0x03 according to ISO 14229-1:2020 [\[9\]](#).]

When using paged buffer mechanism, in some case, it's possible that the number of DTC matching the filter change between the calculation of the total size, needed for the first page transmission, and the sending of the further pages. For this reason, the requirement [\[SWS_Dcm_00587\]](#) and [\[SWS_Dcm_00588\]](#) define the behavior in these situations.

7.4.2.5.7 Subfunctions 0x04 and 0x18

Using UDS Service 0x19 with subfunction 0x04 or 0x18, an external tester can request snapshot record data for one or all snapshot records of a specific DTC.

To process the UDS Service 0x19 04 or 0x19 18, the [Dcm](#) supports the option to process the entire service BSW internally while interacting with the [Dem](#) and the option to leave the service processing to an external service processor.

[SWS_Dcm_01623] Use of internal service processor to read snapshot records

Upstream requirements: [RS_Diag_04180](#), [RS_Diag_04205](#)

[If a [DcmDsdService](#) for service 0x19 with a [DcmDsdSubService](#) for subfunction 0x04 or 0x18 is configured and [DcmDsdSubServiceFnc](#) is not configured for that service, the [Dcm](#) shall process the diagnostic request according to ISO 14229-1 [\[1\]](#).]

[SWS_Dcm_01624] Use of external service processor to read snapshot records

Upstream requirements: [RS_Diag_04205](#)

[If a [DcmDsdService](#) for service 0x19 with a [DcmDsdSubService](#) for subfunction 0x04 or 0x18 is configured and [DcmDsdSubServiceFnc](#) is configured for that service, the [Dcm](#) shall leave the processing of the diagnostic request to the external service processor provided in [DcmDsdSubServiceFnc](#).]

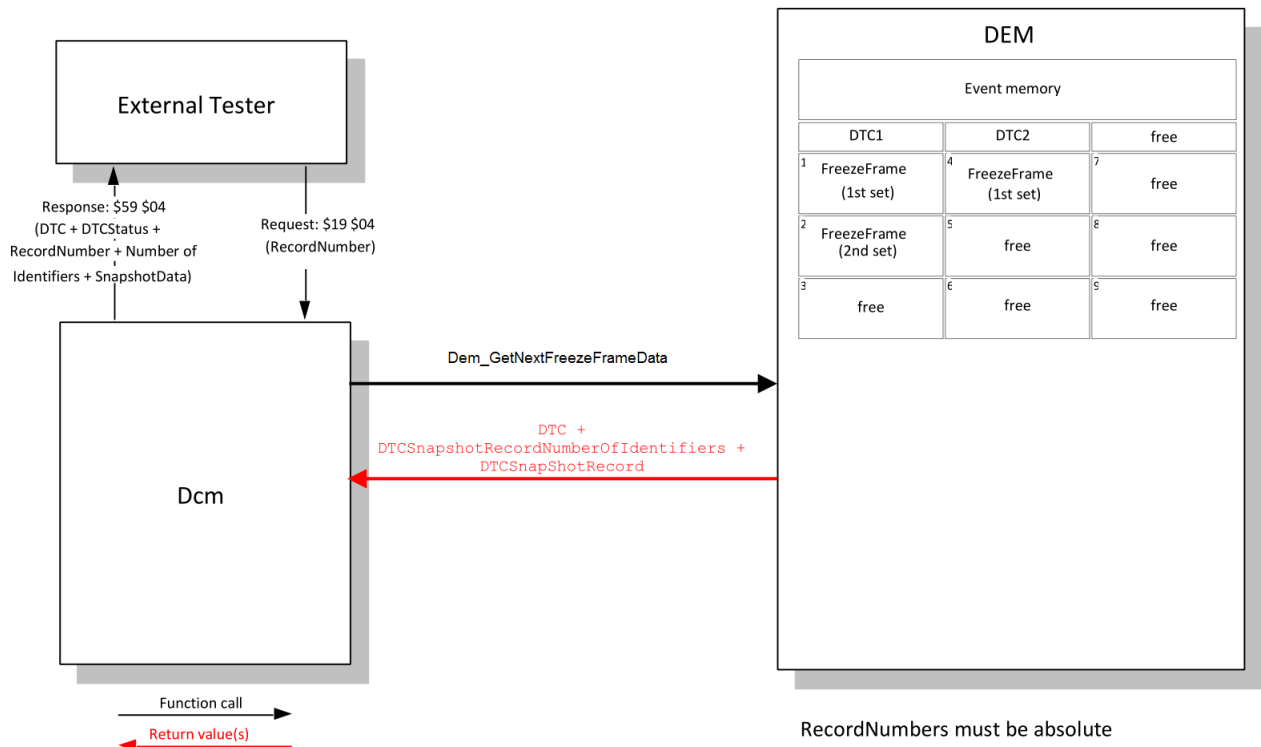


Figure 7.10: Request DTC Snapshot Record by Snapshot Record Number

[SWS_Dcm_01546] Read user defined memory snapshot record authentication check

Upstream requirements: [RS_Diag_04233](#)

[On reception of the UDS Service ReadDTCInformation (0x19) with sub-function reportUserDefMemoryDTCsSnapshotRecordByDTCNumber (0x18), the Dcm shall check if the access to the selected user defined memory in parameter MemorySelection is authenticated and read the DTC information only if:

- a `DcmDspReadDTCInformationUserDefinedFaultMemoryRoleRef` is configured with `DcmDspReadDTCInformationUserDefinedFaultMemoryId` matching the MemorySelection and the verification according to [SWS_Dcm_01522] was successful or
- the active white list on that connection has an entry for that requested user defined memory selection.

]

According to [SWS_Dcm_01546] the authentication checks are only executed if `DcmDspAuthentication` is configured. In case of a failed authentication the NRC handling is according to [SWS_Dcm_01485] and [SWS_Dcm_01551] applies.

7.4.2.5.8 Subfunction 0x05

UDS Service 0x19 with subfunction 0x05 allows an external tester to request Freeze-Frame information for a specific FreezeFrame record number. The service request contains parameter:

- DTCSStoredDataRecordNumber

Due to Dem limitation, the diagnostic service \$19 05 is limited to the OBD legislative freeze frame.

[SWS_Dcm_00632] [On reception of service 0x19 with subfunction 0x05, if the record number of the diagnostic request is different from 0x00, the Dcm module shall send a negative response with NRC 0x31 (request out of range).]

[SWS_Dcm_00574] [When sending a positive response to UDS Service 0x19 with subfunction 0x05 and DTCSStoredDataRecordNumber is 0x00, the Dcm module shall use the data in the response message according to [SWS_Dcm_01651].]

[SWS_Dcm_01651] Subfunction 0x05 response values [

Parameter name	Value
DTCSStoredDataRecordNumber	DTCSStoredDataRecordNumber from request (0x00)
DTCAndStatusRecord	DTC according to [SWS_Dcm_01193], statusOfDTC according to [SWS_Dcm_00389]
DTCSStoredDataRecordNumberOfIdentifiers / DTCSStoredDataRecord	As defined in [SWS_Dcm_00388]

]

[SWS_Dcm_00388]

Upstream requirements: RS_Diag_04058

[When responding to UDS Service 0x19 with subfunction 0x05 and DTCSStoredDataRecordNumber is 0x00, the Dcm shall compose the OBD FreezeFrame by looping all DcmDspPid and collecting all DcmDspPidData which are configured for service 0x02 by calling Dem_DcmReadDataOfOBDFreezeFrame() for the Data Element. The Dcm shall compose the DidId by adding 0xF400 to the Pid, and calculate padding and supported informations.]

[SWS_Dcm_01193] [When responding to UDS Service 0x19 with subfunction 0x05 and DTCSStoredDataRecordNumber is 0x00, the Dcm shall call Dem_DcmGetDTCOfOBDFreezeFrame() with FrameNumber 0x00 and DTCFormat DEM_DTC_FORMAT_UDS to retrieve the DTC of the provided FreezeFrame.]

[SWS_Dcm_00389]

Upstream requirements: [RS_Diag_04058](#)

[When responding to [UDS](#) Service 0x19 with subfunction 0x05 and DTCSaved-DataRecordNumber is 0x00, the [Dcm](#) module shall obtain the status of the [DTC](#) by first calling Dem_SelectDTC() with the following parameters:

- ClientId: Client Id for this [Dcm](#) instance (see [DcmDemClientRef](#))
- DTC: [DTC](#) as defined in [\[SWS_Dcm_00388\]](#)
- DTCOrigin: DEM_DTC_ORIGIN_PRIMARY_MEMORY

and then Dem_GetStatusOfDTC() with the following parameter:

- ClientId: Client Id for this [Dcm](#) instance (see [DcmDemClientRef](#))

]

7.4.2.5.9 Subfunctions 0x0B, 0x0C, 0x0D and 0x0E

An external test tool can request the first occurred or most recent failed or confirmed [DTC](#) and associated status, by sending the [UDS](#) Service request 0x19 including one of the following sub-functions 0x0B, 0x0C, 0x0D, 0x0E

[SWS_Dcm_00392] [When sending a positive response to [UDS](#) Service 0x19 with subfunction 0x0B, 0x0C, 0x0D or 0x0E, the [Dcm](#) module shall use the data in the response message according to [\[SWS_Dcm_01652\]](#).]

[SWS_Dcm_01652] Subfunctions 0x0B, 0x0C, 0x0D and 0x0E response values [

Parameter name	Value
DTCSavedAvailabilityMask	DTCSavedAvailabilityMask
DTCAndStatusRecord	The DTC is obtained according to [SWS_Dcm_00466] , the StatusOfDtc is obtained according to [SWS_Dcm_00393]

]

[SWS_Dcm_00393]

Upstream requirements: [RS_Diag_04058](#)

[For the purpose of responding to [UDS](#) Service 0x19 with subfunctions 0x0B, 0x0C, 0x0D or 0x0E, the [Dcm](#) module shall obtain the StatusOfDtc by calling Dem_GetStatusOfDTC() with the following parameter values:

- ClientId :Client Id for this [Dcm](#) instance (see [DcmDemClientRef](#))
- [DTC](#): the [DTC](#) value as defined in [\[SWS_Dcm_00466\]](#)
- DTCOrigin: DEM_DTC_ORIGIN_PRIMARY_MEMORY

]

[SWS_Dcm_00466] [For the purpose of responding to UDS Service 0x19 with subfunctions 0x0B, 0x0C, 0x0D or 0x0E, the Dcm shall obtain the DTC with Dem_GetDTCByOccurrenceTime() using the parameter values according to Table [SWS_Dcm_01653].]

[SWS_Dcm_01653] Dem_GetDTCByOccurrenceTime() parameters values for subfunctions 0x0B, 0x0C, 0x0D and 0x0E [

	reportFirstTestFailedDTC	reportFirstConfirmedDTC	reportMostRecentTestFailedDTC	reportMostRecentConfirmedDTC
	0x0B	0x0C	0x0D	0x0E
ClientId	Client Id for this Dcm instance (see DcmDem-ClientRef)	Client Id for this Dcm instance (see DcmDem-ClientRef)	Client Id for this Dcm instance (see DcmDem-ClientRef)	Client Id for this Dcm instance (see DcmDem-ClientRef)
DTCRequest	DEM_FIRST_FAILED_DTC	DEM_FIRST_DET_CONFIRMED_DTC	DEM_MOST_RECENT_FAILED_DTC	DEM_MOST_REC_DET_CONFIRMED_DTC

]

[SWS_Dcm_00766] [If the Dcm received DEM_NO_SUCH_ELEMENT by calling Dem_GetDTCByOccurrenceTime it shall reply with a positive response and empty DT-CAndStatusRecord.]

7.4.2.5.10 Subfunction 0x14

An external test tool may request an ECU to report the FaultDetectionCounter for all DTCs with a "Prefailed" status, by sending a UDS Service request 0x19 with subfunction 0x14.

[SWS_Dcm_00464] [When sending a positive response to UDS Service 0x19 with subfunction 0x14, the Dcm module shall use the data in the response message according to [SWS_Dcm_01654].]

[SWS_Dcm_01654] Subfunction 0x14 response values [

Parameter name	Value
DTC	The DTC is obtained according from the call to Dem_GetNextFilteredDTCAndFDC()
DTCFaultDetectionCounter	The DTCFaultDetectionCounter is obtained according from the call to Dem_GetNextFilteredDTCAndFDC()

]

[SWS_Dcm_00465]

Upstream requirements: [RS_Diag_04058](#)

[When responding to [UDS](#) Service 0x19 with subfunctions 0x14, the [Dcm](#) module shall obtain the DTCFaultCounter of every DTCs with status "prefailed" by repeatedly calling `Dem_GetNextFilteredDTCAndFDC()` after having configured the filter with `Dem_SetDTCFilter()` using the parameter values according to [\[SWS_Dcm_01655\]](#).]

[SWS_Dcm_01655] Dem_GetNextFilteredDTCAndFDC() parameters values for subfunctions 0x14 [

Parameter name	Value
ClientId	Client Id for this Dcm instance (see DcmDemClientRef)
DTCStatusMask	0x00
DTCFormat	DEM_DTC_FORMAT_UDS
DTCOrigin	PRIMARY_MEMORY
FilterWithSeverity	FALSE
DTCSeverityMask	Not relevant
FilterForFaultDetectionCounter	TRUE

]

[SWS_Dcm_00681] [The [Dcm](#) module shall obtain the number of records that will be found using `Dem_GetNextFilteredDTCAndFDC()` by using `Dem_GetNumberOfFilteredDTC()`.]

[SWS_Dcm_00519] [The calls to `Dem_SetDTCFilter()` with parameter `FilterForFaultDetectionCounter` set to YES shall be done in the context of the [Dcm_MainFunction](#)]

This allows the implementation to calculate the total size of the response before cycling through the DTCs.

When using paged buffer mechanism, in some case, it's possible that the number of [DTC](#) matching the filter change between the calculation of the total size, needed for the first page transmission, and the sending of the further pages. For this reason, the requirement [\[SWS_Dcm_00587\]](#) and [\[SWS_Dcm_00588\]](#) shall be considered for the implementation of this subservice.

[SWS_Dcm_01232] [If `Dem_GetNextFilteredDTCAndFDC()` returns `DEM_NO_SUCH_ELEMENT` and at least one matching element could be retrieved before, the [Dcm](#) shall send a positive response including these data elements.]

[SWS_Dcm_01233] [If `Dem_GetNextFilteredDTCAndFDC()` returns `DEM_NO_SUCH_ELEMENT` and no matching element could be retrieved before, the [Dcm](#) shall send a positive response only for service and subservice.]

7.4.2.5.11 Subfunction 0x1A

UDS Service 0x19 with subfunction 0x1A request the Dcm to retrieve all DTCs that support an extended data record.

[SWS_Dcm_01607] Support of UDS Service 0x19 with subfunction 0x1A

Upstream requirements: RS_Diag_04250

[The Dcm shall support UDS Service 0x19 with subfunction 0x1A according to ISO 14229-1:2020 [9].]

To retrieve the DTCs that support an extended data record, the Dem provides the API Dem_SetDTCFilterByExtendedDataRecordNumber that allows to set a filter for an extended data record. Subsequent calls to Dem_GetNextFilteredDTC provide the requested data.

The positive response consists of an enumeration of DTC number and DTC status. The Dcm does not define any specific order in which the DTCs are reported in the positive response message.

[SWS_Dcm_01608] Positive response of UDS Service 0x19 with subfunction 0x1A

Upstream requirements: RS_Diag_04250

[If one or more extended data record is/are supported for the requested DTCEst-DataRecordNumber, the Dcm shall send a positive response including all DTCs and the DTC status.]

[SWS_Dcm_01609] Negative response of UDS Service 0x19 with subfunction 0x1A

Upstream requirements: RS_Diag_04250

[If no extended data record is supported for the requested DTCEstDataRecordNumber, the Dcm shall send a negative response with NRC 0x31 (RequestOutOfRange).]

7.4.2.5.12 Subfunction 0x42

UDS Service 0x19 with subfunction 0x42 requests WWH OBD DTCs matching a DTC status mask a severity mask record. The service request contains the following parameters:

- FunctionalGroupIdentifier
- DTCSeverityMask
- DTCStatusMask

[SWS_Dcm_01129] [When sending a positive response to UDS Service 0x19 with subfunction 0x42, the Dcm module shall use the data in the response message according to [SWS_Dcm_01656].]

[SWS_Dcm_01656] Subfunction 0x42 response values [

Parameter name	Value
FunctionalGroupIdentifier	0x33
DTCStatusAvailabilityMask	Dem_GetDTCStatusAvailabilityMask
DTCSeverityAvailabilityMask	Dem_GetDTCSeverityAvailabilityMask
DTCFormatIdentifier	Dem_GetTranslationType (limited to values 0x04 and 0x02)
DTCAndSeverityRecord	As defined in [SWS_Dcm_01130]

]

[SWS_Dcm_01130] [When responding to [UDS](#) Service 0x19 with subfunction 0x42, the [Dcm](#) module shall obtain the DTCAndSeverityRecords by repeatedly calling Dem_GetNextFilteredDTCAndSeverity() after having configured the filter with Dem_SetDTCFilter() using the parameter values according to [SWS_Dcm_01657].]

[SWS_Dcm_01657] Dem_GetNextFilteredDTCAndSeverity() parameters values for subfunctions 0x42 [

Parameter name	Value
ClientId	Client Id for this Dcm instance (see DcmDemClientRef)
DTCStatusMask	DTCStatusMask from request
DTCFormat	DEM_DTC_FORMAT_UDS
DTCOrigin	DEM_DTC_ORIGIN_OBD_RELEVANT_MEMORY
FilterWithSeverity	TRUE
DTCSeverityMask	DTCSeverityMask from request
FilterForFaultDetectionCounter	FALSE

]

[SWS_Dcm_01131] [The return values of Dem_GetNextFilteredDTCAndSeverity shall be filled according to [SWS_Dcm_01658].]

[SWS_Dcm_01658] Dem_GetNextFilteredDTCAndSeverity return values [

Parameter name	Value
DTCSeverity	DTCSeverity
DTCHighByte (MSB)	DTC (high byte)
DTCMiddleByte	DTC (middle byte)
DTCLowByte	DTC (low byte)
statusOfDTC	DTCStatus

]

Note: The [Dcm](#) module can get an indication of the number of records that will be found using Dem_GetNextFilteredDTCAndSeverity() by using Dem_GetNumberOfFilteredDTC().

7.4.2.5.13 Subfunction 0x55

With UDS Service 0x19 with sub-function 0x55 a client can retrieve a list of OBD DTCs with the "permanent DTC" status. The service request contains the following parameter:

- FunctionalGroupIdentifier

[SWS_Dcm_01345] [When sending a positive response to UDS Service 0x19 with sub-function 0x55, the Dcm module shall use the following data in the response message according to [SWS_Dcm_01659].]

[SWS_Dcm_01659] Subfunction 0x55 response values [

Parameter name	Value
FunctionalGroupIdentifier	0x33
DTCStatusAvailabilityMask	Dem_GetDTCStatusAvailabilityMask
DTCFormatIdentifier	Dem_GetTranslationType (limited to values 0x04 and 0x02)
DTCAndStatusRecord	As returned by Dem_GetNextFilteredDTC()

]

Note : When responding to UDS Service 0x19 with sub-function 0x55, the Dcm module could obtain the DTCAndStatusRecords by repeatedly calling Dem_GetNextFilteredDTC() after having configured the filter with Dem_SetDTCFilter() using the parameter values according to [SWS_Dcm_01660].

[SWS_Dcm_01660] Dem_SetDTCFilter() parameters values for subfunction 0x55

[

Parameter name	Value
ClientId	See DcmDemClientRef
DTCStatusMask	0x00
DTCFormat	DEM_DTC_FORMAT_UDS
DTCOrigin	DEM_DTC_ORIGIN_PERMANENT_MEMORY
FilterWithSeverity	FALSE
DTCSeverityMask	Not relevant
FilterForFaultDetectionCounter	FALSE

]

The Dcm module can get an indication of the number of records that will be found using Dem_GetNextFilteredDTC() by using Dem_GetNumberOfFilteredDTC().

[SWS_Dcm_01346] [When responding to UDS Service 0x19 with sub-function 0x55 and Dem_GetTranslationType returns a Dem_DTCTranslationFormatType different to 0x02 (DEM_DTC_TRANSLATION_SAEJ1939_73) or 0x04 (DEM_DTC_TRANSLATION_J2012DA_FORMAT_04), the Dcm module shall return NRC 0x10 (generalReject).]

7.4.2.5.14 Subfunction 0x56

UDS Service 0x19 with subfunction 0x56 request the [Dcm](#) to retrieve all [DTCs](#) that are assigned to an [DTC](#) readiness group identifier.

[SWS_Dcm_01610] Support of UDS Service 0x19 with subfunction 0x56

Upstream requirements: [RS_Diag_04250](#)

[The [Dcm](#) shall support UDS Service 0x19 with subfunction 0x56 according to ISO 14229-1:2020 [9].]

To retrieve the [DTCs](#) that are assigned to an [DTC](#) readiness group, the [Dem](#) provides the API `Dem_SetDTCFilterByReadinessGroup` that allows to set a filter for a readiness group. Subsequent calls to `Dem_GetNextFilteredDTC` provide the requested data.

The positive response consists of an enumeration of [DTC](#) number and [DTC](#) status. The [Dcm](#) does not define any specific order in which the [DTCs](#) are reported in the positive response message.

[SWS_Dcm_01611] Positive response of UDS Service 0x19 with subfunction 0x56

Upstream requirements: [RS_Diag_04250](#)

[If one or more [DTCs](#) support the requested [DTC](#) readiness group identifier, the [Dcm](#) shall send a positive response including all [DTCs](#) and the [DTC](#) status.]

[SWS_Dcm_01612] Negative response of UDS Service 0x19 with subfunction 0x56

Upstream requirements: [RS_Diag_04250](#)

[If no [DTC](#) supports the requested [DTC](#) readiness group, the [Dcm](#) shall send a negative response with [NRC](#) 0x31 (requestOutOfRange).]

7.4.2.6 Service 0x22 - ReadDataByIdentifier

[SWS_Dcm_00253] Support of UDS Service ReadDataByIdentifier (0x22) [The [Dcm](#) module shall support the UDS Service ReadDataByIdentifier (0x22) according to ISO 14229-1:2020 [9].]

With UDS Service 0x22, the tester can request the value of one or more DIDs.

[SWS_Dcm_01548] Read UDS DID authentication check

Upstream requirements: [RS_Diag_04233](#)

[On reception of the UDS Service ReadDataByIdentifier (0x22), the [Dcm](#) shall check if the access to all requested DIDs outside the range 0xF200-0xF8FF is authenticated and read the data identifiers only if:

- for that read [DID](#) a role is configured via [DcmDspDidReadRoleRef](#) and the verification according to [SWS_Dcm_01522] was successful or

- the active white list on that connection has for each requested DID one entry with read access that matches that DID.

]

[SWS_Dcm_01549] Read Dynamically defined DID authentication check*Upstream requirements:* [RS_Diag_04233](#)

[On reception of the UDS Service ReadDataByIdentifier (0x22), the Dcm shall check for all requested DIDs inside the range 0xF200-0xF3FF if the content is based of DIDs or parts of DIDs that are authenticated and read the data identifiers only if:

- for those read [DIDs](#) a role is configured via [DcmDspDidReadRoleRef](#) and the verification according to [\[SWS_Dcm_01522\]](#) was successful or
- the active white list on that connection has for each requested DID one entry with read access that matches those DIDs.

]

According to [\[SWS_Dcm_01537\]](#) the authentication checks are only executed if [DcmDspAuthentication](#) is configured. In case of a failed authentication the NRC handling is according to [\[SWS_Dcm_01544\]](#) and [\[SWS_Dcm_01551\]](#) applies.

[SWS_Dcm_00438] [On reception of the [UDS](#) Service ReadDataByIdentifier (0x22), for every requested [DID](#) the [Dcm](#) module shall check if the [DID](#) is supported (see configuration parameter [DcmDspDid](#) and [DcmDspDidRange](#)) If none of the requested DIDs is supported, the [Dcm](#) module shall send [NRC](#) 0x31 (Request out of range).]

[SWS_Dcm_00651] [On reception of the [UDS](#) Service ReadDataByIdentifier (0x22) with [DID](#) in the range 0xF200 to 0xF3FF, the [Dcm](#) module shall check if the [DID](#) can be dynamically defined (the [DcmDspDidInfo](#) it references has the [DcmDspDidDynamicallyDefined](#) set to true). If yes, if this [DID](#) has not been dynamically defined yet by calls to the DynamicallyDefineDataIdentifier (0x2C) service, i.e. it has no data sources defined, the [Dcm](#) module shall send [NRC](#) 0x31 (Request out of range)]

[SWS_Dcm_00652] [On reception of the [UDS](#) Service ReadDataByIdentifier (0x22) with [DID](#) in the range 0xF200 to 0xF3FF, if verification has been successfully done (see [\[SWS_Dcm_00651\]](#)) and the dynamic [DID](#) has been defined with a [DID](#) source (see [\[SWS_Dcm_00646\]](#)), the [Dcm](#) module shall use the configuration of this [DID](#) source to read the data.]

[SWS_Dcm_00864] [On reception of the [UDS](#) Service ReadDataByIdentifier (0x22) with [DID](#) in the range 0xF200 to 0xF3FF, if verification has been successfully done (see [\[SWS_Dcm_00651\]](#)) and the dynamic [DID](#) has been defined with a [DID](#) source (see [\[SWS_Dcm_00646\]](#)), the [Dcm](#) module shall do the session, security and mode dependencies checks for all source DIDs in case the configuration parameter [DcmDspDDDIDcheckPerSourceDID](#) is set to TRUE.]

[SWS_Dcm_00865] [In case the configuration parameter `DcmDspDDDIDcheckPerSourceDID` is set to FALSE, there is no session, security or mode dependencies check for the source DIDs.]

Note: In case there is a need to validate the session or security dependencies always, the DDDID should be cleared by any security and session transitions.

[SWS_Dcm_00653] [On reception of the UDS Service ReadDataByIdentifier (0x22) with `DID` in the range 0xF200 to 0xF3FF, if verification has been successfully done (see [\[SWS_Dcm_00651\]](#)) and the dynamic `DID` has been defined with a memory address (see [\[SWS_Dcm_00646\]](#)), the `Dcm` module shall use the callout `Dcm_ReadMemory` to read the data.]

[SWS_Dcm_00561] [If a `DID` is set as unused (`DcmDspDidUsed` set to FALSE), the `Dcm` shall consider the `DID` as not supported (according to [\[SWS_Dcm_00438\]](#))]

[SWS_Dcm_00433] [On reception of the UDS Service ReadDataByIdentifier (0x22), for every requested `DID` the `Dcm` module shall check if the `DID` has a Read access configured (see configuration parameter `DcmDspDidRead` in `DcmDspDidInfo`). If none of the `DID` has a Read access, the `Dcm` module shall send `NRC 0x31` (Request out of range).]

[SWS_Dcm_00434] [On reception of the UDS Service ReadDataByIdentifier (0x22), for every requested `DID` the `Dcm` module shall check if the `DID` can be read in the current session (see configuration parameter `DcmDspDidReadSessionRef`). If none of the `DID` can be read in the current session, the `Dcm` module shall send a `NRC 0x31` (RequestOutOfRange).]

[SWS_Dcm_00435] [On reception of the UDS Service ReadDataByIdentifier (0x22), for every requested `DID` the `Dcm` module shall check if the `DID` can be read in the current security level (see configuration parameter `DcmDspDidReadSecurityLevelRef`). If not, the `Dcm` module shall send `NRC 0x33` (Security access denied).]

[SWS_Dcm_00819] [On reception of the UDS Service ReadDataByIdentifier (0x22), for every requested `DID` the `Dcm` module shall check if the `DID` can be read in the current mode condition (according to the configuration parameter `DcmDspDidReadModeRuleRef`). If not, the `Dcm` module shall send the calculated negative response of the referenced `DcmModeRule`.]

In a single ReadDataByIdentifier request, it is possible to request multiple DIDs, referred to as a multi-DID request. The maximum number of DIDs that can be retrieved in one request is configurable through the parameter `DcmDspMaxDidToRead`.

There is a special situation in case of ReadDataByIdentifier. The service is used to read UDS DIDs and regulated DIDs. Local regulations mandate support for a certain minimum number of DIDs (for example, at least 6 as specified in SAE J1979-2). Therefore the `Dcm` supports for OBD related services always the smallest value 6, while `DcmDspMaxDidToRead` is only controlling how many UDS related DIDs can be read out.

OBD-related DIDs fall within the range of 0xF400 to 0xF8FF. Any DID outside these ranges are classified as UDS-only DIDs or do not fall under external regulations that specify a maximum number of DIDs that can be retrieved in a single request. It is therefore possible to have a different value of multiple DIDs supported, depending if the read DIDs are OBD DIDs or UDS DIDs.

[SWS_Dcm_01335] Maximum number of DIDs in non-OBD DID request

Upstream requirements: RS_Diag_04275

[On reception of the UDS Service ReadDataByIdentifier (0x22) with more than one requested DID and at least one DID is outside the OBD-range within 0xF400 and 0xF8FF, the Dcm shall support a request with DcmDspMaxDidToRead DIDs.]

[SWS_Dcm_00439] [On reception of the UDS Service ReadDataByIdentifier (0x22), for every requested DID outside the OBD range (F400-F8FF), the Dcm module shall request the application if the DID can be read by calling the configured function (if parameter DcmDspDataUsePort set to USE_DATA_SYNCH_FNC or USE_DATA_ASYNCH_FNC or USE_DATA_ASYNCH_FNC_ERROR or USE_DATA_SYNCH_FNC_PROXY or USE_DATA_ASYNCH_FNC_PROXY; see configuration parameter DcmDspDataConditionCheckReadFnc) on each data of the DID or call the associated ConditionCheckRead operation (if parameter DcmDspDataUsePort set to USE_DATA_SYNCH_CLIENT_SERVER or USE_DATA_ASYNCH_CLIENT_SERVER or USE_DATA_ASYNCH_CLIENT_SERVER_ERROR). If not (one function returns E_NOT_OK), the Dcm module shall send a negative response with NRC set to value from the parameter "ErrorCode" of DcmDspDataConditionCheckReadFnc function or ConditionCheckRead operation.]

Note: This requirement exceeds the standard ConditionCheck behavior as described by ISO 14229-1 "NRC handling for ReadDataByIdentifier service", because it is not restricted to use NRC 0x22 in a negative response.

[SWS_Dcm_00440] [If the requested DID references other DID using DcmDspDidRef, the Dcm module shall process the verification and the reading of every referenced DID and concatenate the response data without any gaps based on the sequence in the configuration.]

[SWS_Dcm_CONSTR_06023] DcmDspDidRef shall not reference the same DID reference twice [DcmDspDid container shall not include the same DcmDspDidRef parameters more than once.]

[SWS_Dcm_CONSTR_06057] Dependency for DcmDspDataEcuSignal [DcmDspDataEcuSignal shall be only present if DcmDspDataUsePort is set to USE_ECU_SIGNAL.]

[SWS_Dcm_CONSTR_06058] Dependency for DcmDspDataEndianness [In case DcmDspDataEndianness is not configured, the DcmDspDataDefaultEndianness shall be used instead.]

[SWS_Dcm_CONSTR_06061] Dependency for `DcmDspDataReadDataLengthFnc` [`DcmDspDataReadDataLengthFnc` shall be only present if:

- `DcmDspDataUsePort` is set to `USE_DATA_SYNC_FNC` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_ERROR` or
- `DcmDspDataUsePort` is set to `USE_DATA_SYNC_FNC_PROXY` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_PROXY`

]

[SWS_Dcm_CONSTR_06062] Dependency for `DcmDspDataReadFnc` [`DcmDspDataReadFnc` shall be only present if:

- `DcmDspDataUsePort` is set to `USE_DATA_SYNC_FNC` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_ERROR` or
- `DcmDspDataUsePort` is set to `USE_DATA_SYNC_FNC_PROXY` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_PROXY`

]

[SWS_Dcm_01385] [If the `DID` dataRecord has bytes, not defined by any data element, the `Dcm` shall fill these bytes with the value 0x00.]

[SWS_Dcm_01431] [If the configuration parameter `DcmDspDidSize` is configured, the `Dcm` shall enforce the `DID` data to have the configured size.]

[SWS_Dcm_01587] Atomic BndM read operation

Upstream requirements: [RS_Diag_04243](#)

[After all verification (see [\[SWS_Dcm_00433\]](#), [\[SWS_Dcm_00434\]](#), [\[SWS_Dcm_00435\]](#), [\[SWS_Dcm_00436\]](#)) the `Dcm` module shall get for every requested `DID` with `DcmDspDidUsePort` set to `USE_ATOMIC_BNDM` and outside the `OBD` range (F400-F8FF) the data values by reading the data from the associated `BlockId` from the `BndM` ([DcmDspDidBndMBlockIdRef](#)) using the block specific reading function `BndM_GetBlockPtr_<BlockId.Shortname>`.]

7.4.2.6.1 UDS DID

[SWS_Dcm_00578] [On reception of the `UDS` Service `ReadDataByIdentifier` (0x22), for every requested `DID` outside the `OBD` range (F400-F8FF), after all verification (see [\[SWS_Dcm_00433\]](#), [\[SWS_Dcm_00434\]](#) and [\[SWS_Dcm_00435\]](#)), If the

data is configured as a "ECU signal" of the IoHwAb (parameter `DcmDspDataUsePort`), the `Dcm` shall call the Api `IoHwAb_Dcm_Read<EcuSignalName>()` (parameter `DcmDspDataReadEcuSignal`) to get the Data. In this case, the requirements [SWS_Dcm_00439], [SWS_Dcm_00436] and [SWS_Dcm_00437] shall not apply.]

[SWS_Dcm_00436] [On reception of the UDS Service `ReadDataByIdentifier` (0x22), for every requested `DID` outside the OBD range (F400-F8FF), the `Dcm` module shall for each signal (`DcmDspDidSignal`) with a dynamic data length (`DcmDspDataType` is set to `UINT8_DYN`): call either the configured function `DcmDspDataReadDataLengthFnc` (if parameter `DcmDspDataUsePort` set to `USE_DATA_SYNCH_FNC` or `USE_DATA_ASYNCH_FNC` or `USE_DATA_ASYNCH_FNC_ERROR` or `USE_DATA_SYNCH_FNC_PROXY` or `USE_DATA_ASYNCH_FNC_PROXY`) or the associated `ReadDataLength` operation (if parameter `DcmDspDataUsePort` set to `USE_DATA_SYNCH_CLIENT_SERVER` or `USE_DATA_ASYNCH_CLIENT_SERVER` or `USE_DATA_ASYNCH_CLIENT_SERVER_ERROR`) to get the data length in byte.]

[SWS_Dcm_00437] [After all verification (see [SWS_Dcm_00433], [SWS_Dcm_00434], [SWS_Dcm_00435] and [SWS_Dcm_00436]) the `Dcm` module shall get for every requested `DID` outside the OBD range (F400-F8FF), all the data values by calling all the configured function (if parameter `DcmDspDataUsePort` set to `USE_DATA_SYNCH_FNC` or `USE_DATA_ASYNCH_FNC` or `USE_DATA_ASYNCH_FNC_ERROR` or `USE_DATA_SYNCH_FNC_PROXY` or `USE_DATA_ASYNCH_FNC_PROXY`; see configuration parameter `DcmDspDataReadFnc`) or call all the associated `ReadData` operations (if parameter `DcmDspDataUsePort` set to `USE_DATA_SYNCH_CLIENT_SERVER` or `USE_DATA_ASYNCH_CLIENT_SERVER` or `USE_DATA_ASYNCH_CLIENT_SERVER_ERROR`) or read all the associated `SenderReceiver` interfaces (if parameter `DcmDspDataUsePort` set to `USE_DATA_SENDER_RECEIVER` or to `USE_DATA_SENDER_RECEIVER_AS_SERVICE`).]

[SWS_Dcm_01432] [After all verification (see [SWS_Dcm_00433], [SWS_Dcm_00434], [SWS_Dcm_00435] and [SWS_Dcm_00436]) the `Dcm` module shall get for every requested `DID` with `DcmDspDidUsePort` set to `USE_ATOMIC_SENDER_RECEIVER_INTERFACE`, `USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE` or `USE_ATOMIC_NV_DATA_INTERFACE` and outside the OBD range (F400-F8FF) the data values by reading the associated sender-receiver or `NvDataInterface` `DataServices_{DID}`.]

[SWS_Dcm_00638] [To serialize the required AUTOSAR data types (signed- and unsigned integer) into the response message of `ReadDataByIdentifier` responses, the target endianness configured in `DcmDspDataEndianness` shall be considered for `DcmDspData` elements having `DcmDspDataUsePort` set to `USE_DATA_SENDER_RECEIVER`, `USE_DATA_SENDER_RECEIVER_AS_SERVICE`, `USE_ECU_SIGNAL`.]

[SWS_Dcm_CONSTR_06070] Dependency for `DcmDspDataEndianness` [In case `DcmDspDataEndianness` is not present, the `DcmDspDataDefaultEndianness` shall be used instead.]

7.4.2.6.2 OBD DID

[SWS_Dcm_00481] [On reception of the [UDS Service ReadDataByIdentifier \(0x22\)](#), for every requested [DID](#) inside the [OBD](#) range (F400-F4FF), the [Dcm](#) module shall get the [DID](#) value as defined for [OBD Service \\$01](#) (see [\[SWS_Dcm_00407\]](#), [\[SWS_Dcm_00408\]](#), [\[SWS_Dcm_00621\]](#), [\[SWS_Dcm_00622\]](#), [\[SWS_Dcm_00623\]](#), [\[SWS_Dcm_00944\]](#) and [\[SWS_Dcm_00718\]](#)), if [DcmDspEnableObdMirror](#) is set to true.]

[SWS_Dcm_00482] [On reception of the [UDS Service ReadDataByIdentifier \(0x22\)](#), for every requested [DID](#) inside the [OBD Monitor](#) range (F600-F6FF), the [Dcm](#) module shall get the [DID](#) value as defined for [OBD Service \\$06](#) (see [\[SWS_Dcm_00957\]](#), [\[SWS_Dcm_00958\]](#), and [\[SWS_Dcm_00956\]](#))]

[SWS_Dcm_00483] [On reception of the [UDS Service ReadDataByIdentifier \(0x22\)](#), for every requested [DID](#) inside the [OBD InfoType](#) range (F800-F8FF), the [Dcm](#) module shall get the [DID](#) value as defined for [OBD Service \\$09](#) (see [\[SWS_Dcm_00422\]](#) and [\[SWS_Dcm_00423\]](#) without including the number of data items within the response, if [DcmDspEnableObdMirror](#) is set to true.]

[SWS_Dcm_00484] Maximum number of DIDs in a OBD DID request

Upstream requirements: [RS_Diag_04275](#), [RS_Diag_04252](#)

[On reception of the [UDS Service ReadDataByIdentifier \(0x22\)](#) with more than one requested [DID](#), and if all DIDs are within the [OBD](#) range of 0xF400 to 0xF8FF, the [Dcm](#) shall support a request containing at least 6 DIDs.]

Autosar does not define the situation if the parameter [DcmDspMaxDidToRead](#) is configured to a higher value than 6 and [OBD](#) DIDs are requested if a value of 6 or [DcmDspMaxDidToRead](#) or any other behavior of the [Dcm](#) is used.

[SWS_Dcm_01195] [If [DcmDspEnableObdMirror](#) is set to true, an explicitly configured [DID](#) inside the [OBD](#) range (F400-F4FF) and the [OBD InfoType](#) range (F800-F8FF) shall use the [UDS](#) interface.]

[SWS_Dcm_01197] [If [DcmDspEnableObdMirror](#) is set to FALSE, all requests within the [OBD DID](#) range shall use the [UDS](#) interface.]

If [DcmDspEnableObdMirror](#) is set to FALSE ([\[SWS_Dcm_01197\]](#)) or the [DID](#) is explicitly configured inside the [OBD PID](#) range (F400-F4FF) ([\[SWS_Dcm_01195\]](#)), the access to the [OBD](#) data shall be given in the following way:

[SWS_Dcm_01379] [On reception of an [UDS Service ReadDataByIdentifier \(0x22\)](#) request with only "availability OBDDataIdentifier" as parameter, the [Dcm](#) shall respond with the corresponding supported (=configured) [DIDs](#) in the [OBD](#) range (F400-F4FF).]

[SWS_Dcm_01380] [On reception of an [UDS Service ReadDataByIdentifier \(0x22\)](#) request with only OBDDataIdentifier that are not "availability OBDDataIdentifier", the

`Dcm` shall obtain the current value of these `OBDDataIdentifier` by invoking the configured `Xxx_ReadData()` functions for every data of the `OBDDataIdentifier` and shall return these values as response to Service 0x22.]

[SWS_Dcm_01382] [If an `OBDDataIdentifier` contains support information (presence of `DcmDspDidDataSupportInfo` container), the `Dcm` shall add the support information in the diagnostic response.]

[SWS_Dcm_01383] [If an `OBDDataIdentifier` contains support information (presence of `DcmDspDidDataSupportInfo` container), the `Dcm` shall calculate the support information value according to the available data for this `DID`: for every `DcmDspData` container existing for this `DID`, the associated support information bits, referenced in `DcmDspDidDataSupportInfo`, shall be set to one.]

[SWS_Dcm_01384] [When responding to `UDS` Service `ReadDataByIdentifier` (0x22) with `OBDDataIdentifier`, the `Dcm` shall put fill-bytes between `DcmDspData` in the `OBDDataIdentifier` whenever content bytes are missing, in order to fit to the `DID` size (see configuration parameter `DcmDspDidSize`).]

[SWS_Dcm_01386] [To serialize the required AUTOSAR data types (signed and unsigned integer) into the response message of `ReadDataByIdentifier` (0x22) `OBDDataIdentifier` responses the target endianness configured in `DcmDspDataEndianness` shall be considered for `DcmDspData` elements having `DcmDspDataUsePort` set to {`USE_DATA_SENDER_RECEIVER`, `USE_DATA_SENDER_RECEIVER_AS_SERVICE`, `USE_ECU_SIGNAL`}. In case `DcmDspDataEndianness` is not present, the `DcmDspDataDefaultEndianness` shall be used instead.]

If `DcmDspEnableObdMirror` is set to `FALSE` or the `DID` is explicitly configured inside the `OBD` InfoType range (F800-F8FF), the access to the `OBD` data shall be given in the following way:

[SWS_Dcm_01387] [On reception of an `UDS` Service `ReadDataByIdentifier` (0x22) request with one or more "availability `OBDInfoTypeDataIdentifier`" as parameter, the `Dcm` module shall respond with the corresponding supported (=configured) `DIDs` in the `OBD` range (F800-F8FF).]

[SWS_Dcm_01389] [On reception of an `UDS` Service `ReadDataByIdentifier` (0x22) request with an `OBDInfoTypeDataIdentifier` that is not an "availability `OBDInfoTypeDataIdentifier`", the `Dcm` module shall obtain the value of this `OBDInfoTypeDataIdentifier` by invoking all the configured `Xxx_ReadData()` function for every data of this `OBDInfoTypeDataIdentifier` and shall return the value as response to Service 0x22.]

7.4.2.7 Service 0x24 - ReadScalingDataByIdentifier

[SWS_Dcm_00258] [The [Dcm](#) module shall implement the [UDS](#) Service ReadScalingDataByIdentifier (0x24)]

To obtain scaling information, the tester can invoke [UDS](#) Service 0x24 with the 2-byte [DID](#) as parameter. The configuration of the [Dcm](#) contains for each configured DID:

- The 2-byte [DID](#) (see configuration parameter [DcmDspDidIdentifier](#))
- For every data of the [DID](#) :
- The function [GetScalingInformation](#) (see configuration parameters [DcmDspDataGetScalingInfoFnc](#) and [DcmDspDataUsePort](#))
- The length of the [ScalingInfo](#) returned by the [GetScalingInformation](#) function (see configuration parameter [DcmDspDataScalingInfoSize](#))

[SWS_Dcm_00394] [On reception of a request for [UDS](#) Service ReadScalingByIdentifier, the [Dcm](#) module shall call every function [Xxx_GetScalingInformation\(\)](#) configured for every data of the [DID](#) received in the request and return the data received in the response]

[SWS_Dcm_01601] [In the context of [\[SWS_Dcm_00490\]](#), the [Dcm](#) shall place the received scaling information for every data of a [DcmDspDid](#) in the same order as specified by [DcmDspDidByteOffset](#).]

[SWS_Dcm_CONSTR_06060] Dependency for [DcmDspDataGetScalingInfoFnc](#)
[[DcmDspDataGetScalingInfoFnc](#) shall be only present if:

- [DcmDspDataUsePort](#) is set to [USE_DATA_SYNCH_FNC](#) or
- [DcmDspDataUsePort](#) is set to [USE_DATA_ASYNC_FNC](#) or
- [DcmDspDataUsePort](#) is set to [USE_DATA_ASYNC_FNC_ERROR](#) or
- [DcmDspDataUsePort](#) is set to [USE_DATA_SYNCH_FNC_PROXY](#) or
- [DcmDspDataUsePort](#) is set to [USE_DATA_ASYNC_FNC_PROXY](#)

]

7.4.2.8 Service 0x27 - SecurityAccess

[SWS_Dcm_00252]

Upstream requirements: [RS_Diag_04005](#)

[The [Dcm](#) module shall implement the [UDS](#) Service SecurityAccess (0x27)]

The purpose of this service is to provide a means to access data and/or diagnostic services, which have restricted access for security, emissions, or safety reasons.

[SWS_Dcm_00321] [If the request length is correct, the [DSP](#) submodule shall check if the requested subfunction value (access type) is configured in the ECU (see configuration parameter [DcmDspSecurityLevel](#)). If the requested subfunction value is not configured, the [DSP](#) submodule shall trigger a negative response with [NRC](#) 0x12 (SubFunction not supported).]

[SWS_Dcm_00323] [If the requested subfunction value is configured and a service with subfunction type "requestSeed" (= odd value) has been received and if the requested access type is already active (see [Dcm_GetSecurityLevel](#)), the [DSP](#) submodule shall set the seed content to 0x00.]

[SWS_Dcm_00324] [In the other case than the one described in [\[SWS_Dcm_00323\]](#) (access type is not active or "send key" request), if [DcmDspSecurityUsePort](#) is set to [USE_ASYNC_CLIENT_SERVER](#), the [DSP](#) submodule shall call the configured operation [Xxx_GetSeed\(\)](#) (in case "request seed" is received) or [Xxx_CompareKey\(\)](#) (in case "send key" is received).]

[SWS_Dcm_00862] [On reception of the [UDS](#) Service [SecurityAccess](#) (0x27) with subfunction type "requestSeed" and if the requested access type is not already active, the [Dcm](#) module shall request a seed by calling the configured [Xxx_GetSeed\(\)](#) function (if the configuration parameter [DcmDspSecurityUsePort](#) is set to [USE_ASYNC_FNC](#), refer to configuration parameter [DcmDspSecurityGetSeedFnc](#)).]

Note : If the static seed mechanism is used, the processing needs to be done by the application implementing the [Xxx_GetSeed\(\)](#) and [Xxx_CompareKey\(\)](#) functions.

[SWS_Dcm_CONSTR_06077] Dependency for [DcmDspSecurityGetSeedFnc](#) [[DcmDspSecurityGetSeedFnc](#) shall be present only if [DcmDspSecurityUsePort](#) is set to [USE_ASYNC_FNC](#).]

[SWS_Dcm_00863] [On reception of the [UDS](#) Service [SecurityAccess](#) (0x27) with subfunction type "sendKey", if the requested access type is not already active and if the "request seed" for the related access type was executed successfully, the [Dcm](#) module shall request the result of a key comparison by calling the configured [Xxx_CompareKey\(\)](#) function (if the configuration parameter [DcmDspSecurityUsePort](#) is set to [USE_ASYNC_FNC](#), refer to configuration parameter [DcmDspSecurityCompareKeyFnc](#)).]

[SWS_Dcm_CONSTR_06075] Dependency for [DcmDspSecurityCompareKeyFnc](#) [[DcmDspSecurityCompareKeyFnc](#) shall be configured only if [DcmDspSecurityUsePort](#) is set to [USE_ASYNC_FNC](#).]

The following list gives as an example, which errors can be detected by the security access service and stored in the error code information:

- RequestSequenceError (NRC 0x24), when invalid access type is send at "send key",
- RequiredTimeDelayNotExpired (NRC 0x37), when time delay is active (see configuration parameter [DcmDspSecurityDelayTime](#)),
- ExceededNumberOfAttempts (NRC 0x36), when number of attempts to get security access reaches or exceeds the configured limit (see configuration parameter [DcmDspSecurityNumAttDelay](#)), and
- InvalidKey (NRC 0x35), when invalid key is send at "send key".

[SWS_Dcm_00325] [If the operation CompareKey() returns E_OK, the [DSP](#) submodule shall set the new access type with DslInternal_SetSecurityLevel()(see the conversion formula given in ECUC_Dcm_00754 DcmDspSecurityLevel).]

[SWS_Dcm_01397]

Upstream requirements: [RS_Diag_04005](#)

[If Xxx_CompareKey() returns value DCM_E_COMPARE_KEY_FAILED and [DcmDspSecurityNumAttDelay](#) is configured, the [Dcm](#) shall increment the attempt counter of the security level for which the sendKey subfunction request failed.]

[SWS_Dcm_00660] [If Xxx_CompareKey() returns value DCM_E_COMPARE_KEY_FAILED and if the number of failed attempts to enter the requested security level (AttemptCounter) is less than the value configured for the [DcmDspSecurityNumAttDelay](#) parameter of the requested security level, the [Dcm](#) module shall send a negative response with NRC 0x35 (InvalidKey) and shall not change the [Dcm](#) internal security level. Note: if [DcmDspSecurityNumAttDelay](#) parameter is not configured, the number of failed attempts to enter the requested security level (AttemptCounter) is not considered.]

[SWS_Dcm_01349] [If Xxx_CompareKey() returns value DCM_E_COMPARE_KEY_FAILED and if the number of failed attempts to enter the requested security level (AttemptCounter) is greater or equal than the value configured for the [DcmDspSecurityNumAttDelay](#) parameter of the requested security level, the [Dcm](#) module shall start the SecurityDelayTimer with the value configured in [DcmDspSecurityDelayTime](#) for the SecurityLevel which was requested in the failed request, send a negative response with NRC 0x36 (exceededNumberOfAttempts) and shall not change the [Dcm](#) internal security level.]

[SWS_Dcm_01150] [If Xxx_CompareKey() returns value E_NOT_OK, the [Dcm](#) module shall send a negative response with NRC code equal to the ErrorCode parameter value and shall not increment the attempt counter or change the [Dcm](#) internal security level.]

[SWS_Dcm_01350] [While the SecurityDelayTimer of SecurityLevel is not yet elapsed, the [Dcm](#) module shall send a negative response with NRC 0x37 (required-TimeDelayNotExpired) on a SecurityAccess (0x27) requestSeed subfunction request for that Security Level.]

[SWS_Dcm_00659] [If Xxx_GetSeed() returns value E_NOT_OK, the [Dcm](#) module shall send a negative response with NRC code equal to the ErrorCode parameter value.]

7.4.2.9 Service 0x28 - CommunicationControl

[SWS_Dcm_00511] [The [Dcm](#) module shall implement the CommunicationControl (service 0x28) of the Unified Diagnostic Services.]

[SWS_Dcm_00512] [On invocation of the sent confirmation function of the [UDS](#) Service CommunicationControl (0x28) from [DSD](#) with the subnet parameter of the request equal to 0x00, the [Dcm](#) shall do for each NetworkHandle (see [DcmDspAllComMChannelRef](#)) which is configured in [DcmDspComControlAllChannel](#):

1. trigger the mode switch [Dcm_CommunicationControl_<Network> ModeDeclarationGroupPrototype](#) to the mode corresponding the communicationType and controlType parameter from the CommunicationControl request.
2. call the Api [BswM_Dcm_CommunicationMode_CurrentState](#) with the parameters NetworkHandleType and [Dcm_CommunicationModeType](#) corresponding to the communicationType and controlType parameter from the CommunicationControl request (see [Dcm_CommunicationModeType](#) definition).

]

[SWS_Dcm_00785] [On invocation of the sent confirmation function of the [UDS](#) Service CommunicationControl (0x28) from [DSD](#) with the subnet parameter of the request equal to 0x0F(CommunicationControl on the network which request is received on), the [Dcm](#) shall do for the NetworkHandle (see [DcmDslProtocolComMChannelRef](#)) of the current received [DcmDslProtocolRxPduRef](#):

1. trigger the mode switch [Dcm_CommunicationControl_<Network> ModeDeclarationGroupPrototype](#) to the mode corresponding to the communicationType and controlType parameter from the CommunicationControl request.
2. call the Api [BswM_Dcm_CommunicationMode_CurrentState](#) with the parameters NetworkHandleType and [Dcm_CommunicationModeType](#) corresponding to the communicationType and controlType parameter from the CommunicationControl request (see [Dcm_CommunicationModeType](#) definition)

]

[SWS_Dcm_00786] [On invocation of the sent confirmation function of the UDS Service CommunicationControl (0x28) from DSD with the subnet parameter of the request between 0x01 and 0x0E, the Dcm shall check if the received subnet parameter (see DcmDspSubnetNumber) is supported. In case it is not supported a NegativeResponse code 0x31 shall be sent. In case it is supported the Dcm shall do for the corresponding NetworkHandle (see DcmDspSpecificComMChannelRef) of the received subnet parameter (see DcmDspSubnetNumber):

1. trigger the mode switch Dcm_CommunicationControl_<Network> ModeDeclarationGroupPrototype to the mode corresponding the communicationType and controlType parameter from the CommunicationControl request.
2. call the Api BswM_Dcm_CommunicationMode_CurrentState the parameters NetworkHandleType and with Dcm_CommunicationModeType corresponding the communicationType and controlType parameter from the CommunicationControl request (see Dcm_CommunicationModeType definition)

]

For some use-cases the Dcm may re-enable the CommunicationControl due to external changed mode conditions:

[SWS_Dcm_00753] [In case that the referenced ModeRule (see ECUC_Dcm_00943) is not fulfilled anymore for a NetworkHandle which is currently in a state other than DCM_ENABLE_RX_TX_NORM_NM, the Dcm shall:

1. switch the mode group Dcm_CommunicationControl_<Network> ModeDeclarationGroupPrototype to DCM_ENABLE_RX_TX_NORM_NM
2. call BswM_Dcm_CommunicationMode_CurrentState with the parameters NetworkHandleType set to the corresponding NetworkHandle of the network and RequestedCommunicationMode set to DCM_ENABLE_RX_TX_NORM_NM

]

[SWS_Dcm_00860] [For a NetworkHandle which is currently in a state other than DCM_ENABLE_RX_TX_NORM_NM if the Dcm is transitioning to default session or upon any diagnostic session change where the new session does not support UDS Service CommunicationControl anymore, the Dcm shall:

1. switch the mode group Dcm_CommunicationControl_<Network> ModeDeclarationGroupPrototype to DCM_ENABLE_RX_TX_NORM_NM
2. call BswM_Dcm_CommunicationMode_CurrentState with the parameters NetworkHandleType set to the corresponding NetworkHandle of the network and RequestedCommunicationMode set to DCM_ENABLE_RX_TX_NORM_NM

]

Note: the NetworkHandles to be considered are all ComM channels which are referenced from either [DcmDspSpecificComMChannelRef](#), [DcmDspAllComMChannelRef](#) or [DcmDspComControlSubNodeComMChannelRef](#).

[SWS_Dcm_01077] [If a CommunicationControl Request with the sub-function "enableRxAndDisableTxWithEnhancedAddressInformation" is received, the [Dcm](#) shall check the "nodeIdentification-Number" listed as [DcmDspComControlSubNodeId](#) and for the referenced network (see [DcmDspComControlSubNodeComMChannelRef](#)), it shall do the followings:

1. trigger the mode switch [Dcm_CommunicationControl_<Network> ModeDeclarationGroupPrototype](#) to the mode corresponding the communicationType and controlType parameter from the CommunicationControl request.
2. call the Api [BswM_Dcm_CommunicationMode_CurrentState](#) with the parameters NetworkHandleType and [Dcm_CommunicationModeType](#) corresponding to the communicationType and controlType parameter from the CommunicationControl request (see [Dcm_CommunicationModeType](#) definition).

The analogue controlType enableRxAndDisableTx shall be used with the the following existing [Dcm_CommunicationModeType](#) values:

- DCM_ENABLE_RX_DISABLE_TX_NORM
- DCM_ENABLE_RX_DISABLE_TX_NM
- DCM_ENABLE_RX_DISABLE_TX_NORM_NM.

]

[SWS_Dcm_01078] [The [Dcm](#) shall trigger a negative response with [NRC](#) 0x31 (RequestOutOfRange), if a CommunicationControl Request with the sub-function "enableRxAndDisableTxWithEnhancedAddressInformation" and a "nodeIdentification-Number" which is not listed as [DcmDspComControlSubNodeId](#) is received.]

[SWS_Dcm_01079] [If a CommunicationControl Request with the sub-function "enableRxAndTxWithEnhancedAddressInformation" is received, the [Dcm](#) shall check the "nodeIdentification-Number" listed as [DcmDspComControlSubNodeId](#) and for the referenced network (see [DcmDspComControlSubNodeComMChannelRef](#)) it shall do the followings:

1. trigger the mode switch [Dcm_CommunicationControl_<Network> ModeDeclarationGroupPrototype](#) to the mode corresponding the communicationType and controlType parameter from the CommunicationControl request.
2. call the Api [BswM_Dcm_CommunicationMode_CurrentState](#) with the parameters NetworkHandleType and [Dcm_CommunicationModeType](#) corresponding to the communicationType and controlType parameter from the CommunicationControl request (see [Dcm_CommunicationModeType](#) definition).

The analogue controlType enableRxAndTx shall be used with this the following existing Dcm_CommunicationType values :

- DCM_ENABLE_RX_TX_NORM
- DCM_ENABLE_RX_TX_NM
- DCM_ENABLE_RX_TX_NORM_NM.

]

[SWS_Dcm_01080] [The [Dcm](#) shall trigger a negative response with [NRC 0x31](#) (RequestOutOfRange), if a CommunicationControl Request with the sub-function "enableRxAndTxWithEnhancedAddressInformation" and a "nodeIdentification-Number" which is not listed as [DcmDspComControlSubNodeId](#) is received.]

[SWS_Dcm_01081] [If [DcmDspComControlSubNodeUsed](#) is set to FALSE the sub-system ([DcmDspComControlSubNode](#)) is not available in this configuration.]

[SWS_Dcm_01082] [If [DcmDspComControlSubNodeUsed](#) is set to TRUE the sub-system ([DcmDspComControlSubNode](#)) is available in this configuration.]

Note : Condition checks (i.e. [NRC 22](#) checks) on CommunicationType and NetworkType as well as check of CommunicationType support (i.e. [NRC 0x31](#) check for CommunicationType) are not directly supported by the Dcm. Supplier/manufacture notifications can be used.

7.4.2.10 Service 0x29 - Authentication

The UDS service Authentication (0x29) is used to provide dynamic means to control the access to diagnostic services. Execution of certain diagnostic services might have impact on safety, emissions or the vehicle. Controlling the access to diagnostic services via certificates provide means to control the access to diagnostic services after production. The access to these resources can be limited in time or bound to certain vehicles or ECUs only. AUTOSAR Dcm provides an out of the box solution for authenticated diagnostics with a given semantics of the UDS service Authentication. Even ISO14229-1 [9] leaves more freedom, the Dcm will use only the functionality specified in this chapter. Further interpretations, certificate types or certificate content are out of scope of the AUTOSAR Dcm module.

[SWS_Dcm_01559] Support of UDS service authentication

Upstream requirements: [RS_Diag_04230](#)

[The Dcm shall implement the UDS service Authentication (0x29) for sub-functions:

- deAuthenticate
- verifyCertificateUnidirectional
- verifyCertificateBidirectional

- proofOfOwnership
- authenticationConfiguration

]

Note: AUTOSAR Dcm only implements the authentication via PKI certificated exchange. Authentication with challenge-response (ACR) is out of scope of AUTOSAR. If it is required it needs a full custom implementation using existing Dcm callouts for custom service processing.

[SWS_Dcm_01551] NRC Handling for UDS service authentication

Upstream requirements: [RS_Diag_04230](#)

[The Dcm shall follow the NRC handling as defined for ISO 14229-1:2018 for UDS service authentication. This includes the NRC codes as well as the order of individual NRC checks.]

[SWS_Dcm_CONSTR_06091] Authentication configuration [The presence of a DcmDsdService with DcmDsdSidTabServiceId set to 0x29, requires a configured container DcmDspAuthentication on DcmDsp.]

[SWS_Dcm_CONSTR_06092] Authentication per connection [The presence of a DcmDsdService with DcmDsdSidTabServiceId set to 0x29, requires a configured DcmDspAuthenticationConnection per configured connection DcmDslConnection.]

[SWS_Dcm_CONSTR_06093] One authentication configuration per connection [Each DcmDspAuthenticationConnection shall refer a different DcmDslMainConnection by the reference in DcmDspAuthenticationConnectionMainConnectionRef.]

The Dcm is using the Csm and KeyM for certificate management. Parsing the certificate is a potential time-consuming operation and needs to be executed asynchronous. There are opinions that security shall not reveal any cause of failed authentication and thus have no dedicated NRCs for different certificate failures. To respect this use case the Dcm provides a general security NRC handling.

[SWS_Dcm_01560] Use of generic NRC for invalid certificate or content

Upstream requirements: [RS_Diag_04230](#)

[If the mode referenced by DcmDspAuthenticationGeneralNRCModeRuleRef evaluates to true, the Dcm shall send the NRC code given in DcmDspAuthenticationGeneralNRC instead of the specific NRC codes in all situation where a Certificate verification failed - NRC is returned.]

[SWS_Dcm_CONSTR_06094] Generic NRC configuration requirements

Upstream requirements: [RS_Diag_04230](#)

[If DcmDspAuthenticationGeneralNRCModeRuleRef is configured the parameter DcmDspAuthenticationGeneralNRC shall also be configured.]

7.4.2.10.1 De-authentication

The de-authenticate sub-function shall be always available if service 0x29 is used. This service set the authentication state to deauthenticated for the diagnostic connection the request was received on.

[SWS_Dcm_CONSTR_06095] [The presence of a DcmDsdService with DcmDsd-SidTabServiceId set to 0x29, requires a DcmDsdSubService on this DcmDsdService with DcmDsdSubServiceId set to deAuthenticate.]

[SWS_Dcm_01561] Deauthentication by diagnostic service request

Upstream requirements: [RS_Diag_04230](#)

[On reception of an Authentication (0x29) service with sub-function equal to de-authenticate, the Dcm shall set the authentication state to deauthenticated on the connection the Dcm has received that request.]

[SWS_Dcm_01565]

Upstream requirements: [RS_Diag_04230](#)

[On reception of an Authentication (0x29) service with sub-function equal to de-Authenticate, the [Dcm](#) shall reply with a positive response having the authentication-ReturnParameter set to DeAuthentication successful.]

The [Dcm](#) has an optional distance (mileage) supervision of the authenticated state. In that case the [Dcm](#) resets the authenticated state automatically after a certain distance is travelled. The unit of the distance is project specific, the [Dcm](#) only compares a distance provided by a data element to an internal configured value. The general idea is to reuse the same distance information that is also provided to the [Dem](#) to calculate certain PIDs values. It is recommended to use a definition that is compatible to SAE J1979-DA. AUTOSAR does not define a unit or resolution for the value retrieved by [DcmDspAuthenticationMileageInformation](#). This is intentionally left open as integration aspect, so it allows the [Dcm](#) to adapt to any existing distance information. The offset limit defined by [DcmDspAuthenticationMileageOffsetLimit](#) needs to take the current resolution into account.

[SWS_Dcm_01566] Authentication distance supervision [If [DcmDspAuthenticationMileageOffsetLimit](#) is configured and [DcmDspAuthenticationPersistStateModeRuleRef](#) is evaluated to false, the Dcm shall monitor the travelled distance with active authenticated state and return to de-authenticated state if the current [DcmDspAuthenticationMileageInformation](#) is greater than the [DcmDspAuthenticationMileageInformation](#) when the authenticated state was reached plus the value of [DcmDspAuthenticationMileageOffsetLimit](#).]

[SWS_Dcm_CONSTR_06114] Complete distance offset configuration [The parameters [DcmDspAuthenticationMileageOffsetLimit](#) and [DcmDspAuthenticationMileageInformation](#) are always needed together, thus if one of the parameters is configured both parameters are required.]

[SWS_Dcm_CONSTR_06115] Limiting data configuration for distance information [A [DcmDspData](#) that is referred by [DcmDspAuthenticationMileageInformation](#) shall have the following restricted settings:

- [DcmDspDataConditionCheckReadFnc](#) not configured
- [DcmDspDataConditionCheckReadFncUsed](#) not configured
- [DcmDspDataType](#) set to `UINT32`
- [DcmDspDataUsePort](#) set to `USE_DATA_SENDER_RECEIVER`, `USE_DATA_SYNCH_CLIENT_SERVER`, `USE_DATA_SYNCH_FNC` or `USE_ECU_SIGNAL`

]

7.4.2.10.2 Verify Certificates

[SWS_Dcm_01459] Supported communicationConfiguration

Upstream requirements: [RS_Diag_04230](#)

[On reception of an Authentication (0x29) service with sub-function equal to `verifyCertificateUnidirectional` or `verifyCertificateBidirectional`, the [Dcm](#) shall reply with an [NRC](#) 0x31 (`requestOutOfRange`) if the `communicationConfiguration` (COCO) parameter has a value different than 0x00.]

The support of a COCO with constant value of 0x00 implies that no session key is supported by the [Dcm](#). Upon receiving an authentication request with sub-function `verifyCertificateUniDirectional` and the `communicationConfiguration` (COCO) set to 0x00, the [Dcm](#) starts to verify the certificate. This service is implemented on BSW level by intention, to reduce the possible dialects of service 0x29 to what is natively supported by AUTOSAR. The [Csm](#) [17] and [KeyM](#) [18] modules allow to use different cryptographic methods. It is part of the [Dcm](#) strategy to explicitly require that certificates and all kind of access to information inside are handled by the [Csm](#) and its configuration. This allows to map different kind of certificates with different levels of security to the [Dcm](#), thus abstracting the certificate complexity from [Dcm](#).

[SWS_Dcm_01460] Verify `verifyCertificateUnidirectional` message length check

Upstream requirements: [RS_Diag_04230](#)

[On reception of an Authentication (0x29) service with sub-function equal to `verifyCertificateUnidirectional`, the [Dcm](#) shall verify that the message length fits to the size given in the parameters `lengthOfCertificateClient` and return a [NRC](#) 0x13 if the size does not match.]

[SWS_Dcm_01461] Verify `verifyCertificateBidirectional` message length check

Upstream requirements: [RS_Diag_04230](#)

[On reception of an Authentication (0x29) service with sub-function equal to `verifyCertificateBidirectional`, the [Dcm](#) shall verify that the message length fits to the size given

in the parameters `lengthOfCertificateClient` and `lengthOfChallengeClient` and return a `NRC 0x13` if the size does not match. The `Dcm` uses the `lengthOfCertificateClient` as offset to calculate the position of `lengthOfChallengeClient`.]

[SWS_Dcm_01462] Required configuration for bidirectional authentication [If the sub-function `verifyCertificateBidirectional` is configured in the `DsdServiceSubFunction` for `0x29`, the configuration parameter `DcmDspAuthenticationECUCertificateRef` is required.]

[SWS_Dcm_01579] Mandatory certificate data

Upstream requirements: [RS_Diag_04230](#)

[On reception of an Authentication (`0x29`) service with sub-function equal to `verifyCertificateUnidirectional` or `verifyCertificateBidirectional` and a `lengthOfCertificateClient` equal to zero, the `Dcm` shall respond with `NRC 0x13` (`incorrectMessageLengthOrInvalidFormat`)..]

Store certificate in Csm

[SWS_Dcm_01463] Certificate validation

Upstream requirements: [RS_Diag_04230](#)

[On reception of an Authentication (`0x29`) service with sub-function equal to `verifyCertificateUnidirectional` or `verifyCertificateBidirectional`, the `Dcm` shall use the `KeyM` API `KeyM_SetCertificate` to store the client certificate from the service request within the `KeyM` module. The following parameter values shall be used:

- `CertificateId` = `DcmDspAuthenticationConnectionCertificateRef` -> `KeyMCertificate.KeyMCertificateId`
- `certificateDataPtr.certData` = Pointer to `certificateClient` from Request
- `certificateDataPtr.certDataLength` = `lengthOfCertificateClient` from Request

]

Diagnostic certificate configuration is a task that is mainly executed in the `Csm` and `KeyM` modules. The `Dcm` provides an abstraction from these modules and only keeps symbolic references to containers that keep the information. Please take attention while configuring the `Csm` and `KeyM` 'in spirit of `Dcm` certificates'.

[SWS_Dcm_01464] Invalid certificate size failure

Upstream requirements: [RS_Diag_04230](#)

[If the API `KeyM_SetCertificate` returns `KEYM_E_SIZE_MISMATCH`, the `Dcm` shall return the `NRC 0x13` (`incorrectMessageLengthOrInvalidFormat`).]

[SWS_Dcm_01465] Behavior on busy crypto operation

Upstream requirements: [RS_Diag_04230](#)

[If any of the Csm or KeyM APIs called by the [Dcm](#) is returning CRYPTO_E_BUSY or KEYM_E_BUSY, the [Dcm](#) shall return the [NRC](#) 0x21 (busyRepeat).]

[SWS_Dcm_01466] Csm APIs returning failure code behavior

Upstream requirements: [RS_Diag_04230](#)

[Throughout this chapter the Csm or KeyM are used to process the authentication requests. These APIs have return values for failures. Unless there is no dedicated requirement for a given return value and if the return value is different to E_OK, the [Dcm](#) shall return a [NRC](#) 0x10 'GeneralReject'.]

The cryptographic operations have potential long execution times and are called asynchronously.

[SWS_Dcm_01467] Asynchronous certificate operation handling

Upstream requirements: [RS_Diag_04230](#)

[For all asynchronous Csm or KeyM operations, the [Dcm](#) shall wait until the callback has been called, indicating a successful job termination. If the result (e.g. KeyM_CertificateStatusType) is E_OK, the [Dcm](#) shall continue to process the 0x29 request, if the result is different to E_OK, the [Dcm](#) shall skip the further processing and return a negative response with [NRC](#) 'GeneralReject'.]

Parse and Verify certificate in Csm**[SWS_Dcm_01468] Verifying client certificate**

Upstream requirements: [RS_Diag_04230](#)

[After the [Dcm](#) has set the certificate according to [\[SWS_Dcm_01463\]](#), the [Dcm](#) shall verify the certificate by calling KeyM_VerifyCertificate with the following parameters:

- CertificateId = [DcmDspAuthenticationConnectionCertificateRef](#) -> KeyMMCertificate.KeyMCertificateId

]

[SWS_Dcm_01469] Behavior on failed certificate verification

Upstream requirements: [RS_Diag_04230](#)

[After the [Dcm](#) has verified a certificate and KeyM_CertificateStatusType is set to KEYM_E_CERTIFICATE_SIGNATURE_FAIL, the [Dcm](#) shall send a negative response with [NRC](#) set to 'Certificate verification failed - Invalid Signature'.]

[SWS_Dcm_01470] Check certificate chain of trust

Upstream requirements: [RS_Diag_04235](#)

[If the operation started in [\[SWS_Dcm_01468\]](#) returns a result of KEYM_E_CERTIFICATE_INVALID_CHAIN_OF_TRUST, the [Dcm](#) shall refuse the

client certificate and return a negative response with **NRC** 'Certificate verification failed - Invalid Chain of Trust'.]

[SWS_Dcm_01471] Check certificate type

Upstream requirements: [RS_Diag_04235](#)

[If the operation started in [\[SWS_Dcm_01468\]](#) returns a result of **KEYM_E_CERTIFICATE_INVALID_TYPE**, the **Dcm** shall refuse the client certificate and return a negative response with **NRC** 'Certificate verification failed - Invalid Type'.]

[SWS_Dcm_01472] Check certificate format

Upstream requirements: [RS_Diag_04235](#)

[If the operation started in [\[SWS_Dcm_01468\]](#) returns a result of **KEYM_E_CERTIFICATE_INVALID_FORMAT**, the **Dcm** shall refuse the client certificate and return a negative response with **NRC** 'Certificate verification failed - Invalid Format'.]

[SWS_Dcm_01473] Check certificate content

Upstream requirements: [RS_Diag_04235](#)

[If the operation started in [\[SWS_Dcm_01468\]](#) returns a result of **KEYM_E_CERTIFICATE_INVALID_CONTENT**, the **Dcm** shall refuse the client certificate and return a negative response with **NRC** 'Certificate verification failed - Invalid Content'.]

Note: While most of the KeyM return values can be mapped 1:1 on **UDS NRC** values, this does not apply for **KEYM_E_CERTIFICATE_INVALID_CONTENT**. An invalid content from KeyM indicates that a key element verification has failed. A failed key element verification means that the certificate itself is valid, but the KeyM refuses the certificate as the data does not fit to the server's required value. In **UDS** this is expressed by 'Certificate verification failed - Invalid Scope'. As result invalid content from KeyM will trigger the **NRC** invalid scope.

[SWS_Dcm_01475] Check certificate validity

Upstream requirements: [RS_Diag_04235](#)

[If the operation started in [\[SWS_Dcm_01468\]](#) returns a result of **KEYM_E_CERTIFICATE_REVOKED**, the **Dcm** shall refuse the client certificate and return a negative response with **NRC** 'Certificate verification failed - Invalid Certificate (revoked)'.]

[SWS_Dcm_01476] Check certificate valid until

Upstream requirements: [RS_Diag_04235](#)

[If the operation started in [\[SWS_Dcm_01468\]](#) returns a result of **KEYM_E_CERTIFICATE_VALIDITY_PERIOD_FAIL**, the **Dcm** shall refuse the client

certificate and return a negative response with NRC 'Certificate verification failed - Invalid Time Period'.]

Providing the server challenge

The Dcm uses the Csm to create a server challenge by returning a random value.

[SWS_Dcm_01588] Length of server challenge

Upstream requirements: RS_Diag_04230

[The Dcm shall create a server challenge with the length configured in DcmDspAuthenticationEcuChallengeLength.]

[SWS_Dcm_01493] Creating the server challenge

Upstream requirements: RS_Diag_04230

[After successfully validating the client certificate in [SWS_Dcm_01463], the Dcm shall create a server challenge by using the Csm in the following sequence:

- 1) Call of Csm_RandomGenerate with parameters
 - a. jobId : DcmDspAuthenticationRandomJobRef -> CsmJobId
 - 2) Wait until the asynchronous job has terminated according to [SWS_Dcm_01467].
-]

The API Csm_RandomGenerate requires an initialised random generator. The initialization of the random generator is a task of the system (integrator).

[SWS_Dcm_01621] [If calculation of server challenge fails i.e Csm_RandomGenerate returns CRYPTO_E_ENTROPY_EXHAUSTED, the Dcm shall send NRC 0x59 (Challenge calculation failed).]

[SWS_Dcm_01503] Sending positive response on verifyCertificateUniDirectional

Upstream requirements: RS_Diag_04230

[If the Dcm has successfully calculated server challenge the Dcm shall send a positive response for the verifyCertificateUniDirectional request with the following parameter values:

authenticationReturnParameter : 'Certificate verified, Ownership verification necessary'

lengthOfChallengeServer: length of the challenge generated in [SWS_Dcm_01493]

challengeServer: challenge generated in [SWS_Dcm_01493]

lengthOfEphemeralPublicKeyServer: 0x0000

]

Processing a diagnostic request verifyCertificateBidirectional will furthermore require the Dcm to send a server certificate and signing the client challenge. Therefore, the following steps are done additionally for verifyCertificateBidirectional.

[SWS_Dcm_01504] Signing client challenge

Upstream requirements: [RS_Diag_04230](#)

[On reception of an Authentication (0x29) service with sub-function equal to verifyCertificateBidirectional, the [Dcm](#) shall sign the received client challenge by calling Csm_SignatureGenerate with the following parameter values:

jobId: [DcmDspAuthenticationClientChallengeSignJobRef](#) -> CsmJobId

mode: CRYPTO_OPERATIONMODE_SINGLECALL

dataPtr: Pointer to challengeClient in request

dataLength: lengthOfChallengeClient from request

resultPtr: [Dcm](#) sided buffer to store the proofOfOwnershipServer

resultLengthPtr: Response data for lengthOfProofOfOwnershipServer

]

[SWS_Dcm_CONSTR_06096] Require asynchronous client challenge signing

Upstream requirements: [RS_Diag_04230](#)

[[DcmDspAuthenticationClientChallengeSignJobRef](#) shall be only accepted if the referenced CsmJob itself:

- has a CsmProcessingMode set to CRYPTO_PROCESSING_ASYNC
- references a CsmPrimitive with an aggregated CsmSignatureGenerate.

]

[SWS_Dcm_01506] Providing the server certificate

Upstream requirements: [RS_Diag_04230](#)

[On reception of an Authentication (0x29) service with sub-function equal to verifyCertificateBidirectional, the [Dcm](#) shall provide the server certificate in the response. The [Dcm](#) shall call KeyM_GetCertificate with the following parameters:

certId: [DcmDspAuthenticationECUCertificateRef/KeyMCertificateId](#)

certificateDataPtr: [Dcm](#) implementation specific

]

[SWS_Dcm_01507] Server certificate size check

Upstream requirements: [RS_Diag_04230](#)

[If the API KeyM_GetCertificate returns KEYM_E_KEY_CERT_SIZE_MISMATCH (0x04), the [Dcm](#) shall return [NRC](#) 0x10 (generalReject).]

[SWS_Dcm_01508] Sending positive response on verifyCertificateBidirectional

Upstream requirements: [RS_Diag_04230](#)

[If the [Dcm](#) has successfully calculated client challenge, the server challenge and the server certificate, the [Dcm](#) shall send a positive response for the verifyCertificateBidirectional request with the following parameter values:

authenticationReturnParameter: 'Certificate verified, Ownership verification necessary'

```
lengthOfChallengeServer: length of the challenge generated in [SWS_Dcm_01493]
challengeServer: challenge generated in [SWS_Dcm_01493]
lengthOfCertificateServer: length of the certificated provided in [SWS_Dcm_01506]
certificateServer: certificated provided in [SWS_Dcm_01506]
lengthOfProofOfOwnershipServer: length of proof-of-ownership server created in
[SWS_Dcm_01504]
proofOfOwnershipServer: proof-of-ownership server created in [SWS_Dcm_01504]
lengthOfEphemeralPublicKeyServer: 0x0000
]
```

7.4.2.10.3 Proof of ownership client

[SWS_Dcm_01509] Sequence check

Upstream requirements: [RS_Diag_04230](#)

[On reception of an Authentication (0x29) service with sub-function equal to proofOfOwnership and if on this connection the most recent processed verifyCertificateUnidirectional or verifyCertificateBidirectional service failed or no such sub-function was processed yet, the [Dcm](#) shall send the negative response 'requestSequenceError'. The connection shall remain in de-authenticated state.]

[SWS_Dcm_01510] Proof of ownership message length check

Upstream requirements: [RS_Diag_04230](#)

[On reception of an Authentication (0x29) service with sub-function equal to proofOfOwnership, the [Dcm](#) shall verify that the message length fits to the size given in the parameters lengthOfProofOfOwnershipClient and lengthOfEphemeralPublicKeyClient and return a [NRC](#) 0x13 if the size does not match.]

[SWS_Dcm_01511] Client proof of ownership verification

Upstream requirements: [RS_Diag_04230](#)

[On reception of an Authentication (0x29) service with sub-function equal to proofOfOwnership, the [Dcm](#) shall verify the client's proof of ownership data provided in the request by calling Csm_SignatureVerify with the following in-parameters:

```
jobId: DcmDspAuthenticationVerifyProofOfOwnerShipClientJobRef ->
CsmJob.CsmJobId
mode: set to CRYPTO_OPERATIONMODE_SINGLECALL
dataPtr: challenge generated [SWS_Dcm_01493]
dataLength: length of the challenge generated in [SWS_Dcm_01493]
signaturePtr: proofOfOwnerShipClient from request
signatureLength: lengthOfProofOfOwnerShipClient from request
]
```

[SWS_Dcm_CONSTR_06098] Require asynchronous signature verification [[DcmDspAuthenticationVerifyProofOfOwnerShipClientJobRef](#) is only valid if the referenced CsmJob itself:

- has a CsmProcessingMode set to CRYPTO_PROCESSING_ASYNC
- references a CsmPrimitive with an aggregated CsmSignatureVerify.

]

[SWS_Dcm_01512] Failed ownership verification*Upstream requirements:* [RS_Diag_04230](#)

[If the result of Csm_SignatureVerify from [\[SWS_Dcm_01511\]](#) is Crypto_VerifyResultType equal to CRYPTO_E_VER_NOT_OK, the Dcm shall send a negative response with [NRC](#) 'Ownership verification failed'.]

[SWS_Dcm_01513] SessionKey support*Upstream requirements:* [RS_Diag_04230](#)

[Upon sending a positive response for an authentication request with sub-function equal to proofOfOwnership, the Dcm shall:

- set all bytes of lengthOfSessionKeyInfo to 0x00
- omit the sessionKeyInfo

]

Set current role**[SWS_Dcm_01514] Update the current role***Upstream requirements:* [RS_Diag_04233](#)

[If the proof of ownership verification in [\[SWS_Dcm_01511\]](#) was successful and resulted in CRYPTO_E_VER_OK, the Dcm shall use the value read from the certificate extension [DcmDspAuthenticationRoleElementRef](#) as new role for the current authenticated state.]

[SWS_Dcm_01515] Role size check*Upstream requirements:* [RS_Diag_04230](#)

[If the size of the read role information in [\[SWS_Dcm_01514\]](#) is different than the size in [DcmDspAuthenticationRoleSize](#), the Dcm shall send a negative response with [NRC](#) 'Certificate verification failed - Invalid Content'.]

Set white list**[SWS_Dcm_01516] Update the current whitelist***Upstream requirements:* [RS_Diag_04232](#)

[If the proof of ownership verification in [\[SWS_Dcm_01511\]](#) was successful and resulted in CRYPTO_E_VER_OK, the Dcm shall use the white list read from the certificate according to [\[SWS_Dcm_01517\]](#) as new white list for the current authenticated state.]

[SWS_Dcm_01517] White list access [To read the white list from a certificate, the *Dcm* shall read all child elements from the referenced *DcmDspAuthenticationWhiteListServicesElementRef*, *DcmDspAuthenticationWhiteListDIDElementRef*, *DcmDspAuthenticationWhiteListRIDElementRef* and *DcmDspAuthenticationWhiteListMemorySelectionElementRef* certificate data.]

[SWS_Dcm_01518] White list size check

Upstream requirements: [RS_Diag_04230](#)

[If the size of the read white list information in [\[SWS_Dcm_01516\]](#) is larger than the size in *DcmDspAuthenticationWhiteListServicesMaxSize* or *DcmDspAuthenticationWhiteListDIDMaxSize* or *DcmDspAuthenticationWhiteListRIDMaxSize* or *DcmDspAuthenticationWhiteListMemorySelectionMaxSize*, the *Dcm* shall send a negative response with NRC 'Certificate verification failed - Invalid Content' (0x25).]

[SWS_Dcm_CONSTR_06087] Required size for white lists

Upstream requirements: [RS_Diag_04232](#)

[If any of the optional *DcmDspAuthenticationWhiteListMemorySelectionElementRef* are configured, the corresponding *DcmDspAuthenticationWhiteListMemorySelectionMaxSize* shall be configured for that white list.]

7.4.2.10.4 Definition and verification of roles

The roles transmitted inside the certificates are used to assign a tester on one connection one or more roles. A single role itself is presented by a certain bit within the bitfield representation of the role. Within the *Dcm* there is static role assignment to each diagnostic service. A service can be executed if at least one role (bit) assigned to a service matches a role (bit) in the certificate.

[SWS_Dcm_CONSTR_06088] Supported role sizes

Upstream requirements: [RS_Diag_04233](#)

[The parameter *DcmDspAuthenticationRoleSize* defines the size in bytes used in both, certificates and ECU internal static role configuration. All role parameters (e.g. *DcmDspServiceRole*) shall have values that would fit in the amount of bytes given by *DcmDspAuthenticationRoleSize*.]

[SWS_Dcm_01521] Integer interpretation of roles in certificates

Upstream requirements: [RS_Diag_04234](#)

[The *Dcm* shall interpret all configured role integer values in the little endian format.]

[SWS_Dcm_01522] Role verification

Upstream requirements: [RS_Diag_04233](#), [RS_Diag_04234](#)

[Upon each role verification, the [Dcm](#) shall make a bitwise 'and' operation on the value provided by the role in the certificate for that connection and the role value assigned to the service. The value that is assigned to the service is calculated by setting all the bits referred by each [DcmDspAuthenticationRow](#). If the result is greater than 0, the [Dcm](#) shall treat the service as allowed to be executed.]

[SWS_Dcm_01523] Failed role verification

Upstream requirements: [RS_Diag_04233](#), [RS_Diag_04234](#)

[Upon each role verification, the [Dcm](#) shall make a bitwise 'and' operation on the role provided in the certificate for that connection and the role assigned to the service. If the result is equal to 0, the [Dcm](#) shall stop the processing of that service and send a negative response 'authenticationRequired'.]

An example of roles in certificates and services with assigned certificates is given in Figure 7.11. The provided certificate uses 1 byte for roles, with 5 role definitions in. The certificate will grant the tester rights for the roles 'AfterSales' and 'ExtendedUser'. With that certificate, the tester can execute the services that have the corresponding permissions to be executed in that roles. In that example, the service 0x28 and 0x11 01 are both allowed to be executed. The routine with identifier 0x5678 is only allowed to be executed in role 'production', the [Dcm](#) will deny the execution with [NRC](#) 'authenticationRequired'.

	Development Production AfterSales ExtendedUser epT unused unused unused							
Bit	7	6	5	4	3	2	1	0
Certificate	0	0	1	1	0	0	0	0
Service List								
Service 0x28	0	0	1	1	0	0	0	0
Service 0x11 01	0	0	1	0	0	0	0	0
RID 0x5678	0	1	0	0	0	0	0	0

Figure 7.11: Example for roles

7.4.2.10.5 Definition and verification of white lists

Besides the use of roles, the certificates can get extended with optional white lists for service execution. This allows the issuer of the certificate to extend the range of allowed services without updating the access lists in the ECU.

The white list is build out of the following elements:

- List of allowed services, 1-4 byte each
- List of allowed data identifiers ([DID](#)) and access information, 3 byte each
- List of allowed routine identifiers ([RID](#)) and access information, 3 byte each
- List of allowed user defined fault memories, 1 byte each

Parsing and access to the different elements of the white lists is done by invoking `KeyM_CertElementGetFirst` and `KeyM_CertElementGetNext` according to [\[SWS_Dcm_01517\]](#).

[SWS_Dcm_01524] White list for services definition

Upstream requirements: [RS_Diag_04233](#), [RS_Diag_04234](#)

[The white list for services is a set of values, each consisting of up to 4 bytes. Each value itself contains the first bytes of a diagnostic service that is allowed be executed.]

Example:

In the following example, a white list section within a certificate is shown. It contains 5 additional services that can be executed:

```
1  ...
2  SEQUENCE (1 elem)
3    SEQUENCE (2 elem)
4      OBJECT IDENTIFIER 3.9.3.345.3.1.3453.24.3.9.355.73
5    SET (6 elem)
6      OCTET STRING (1 byte) 85
7      OCTET STRING (2 byte) 1101
8      OCTET STRING (3 byte) 22123A
9      OCTET STRING (3 byte) 2E123A
10     OCTET STRING (4 byte) 310113F4
11  SEQUENCE (3 elem) ....
```

This will allow the Dcm to execute:

- Service 0x85 (with any sub-subfunction and `DTCSettingControlOptionRecord` byte)
- Service 0x11 01
- Read and WriteDataByIdentifier for [DID](#) 0x123A
- Routine Start for [RID](#) 0x13F4

Checks for white lists for services are all done at [DSD](#) level. The [Dcm](#) checks if the first bytes of a received diagnostic request match the values allowed by the white list. If a white list entry exists where all bytes are matching the first bytes of the request, the service is granted access.

For all other white lists, the [Dcm](#) performs the checks at [DSP](#) level in the corresponding service processors.

Please note that it is possible to allow a **DID** execution in two places in the white list: 1) as 3 byte service and 2) as **DID** in the **DID** list. The difference is that the service checks only the first 3 bytes of the **PDU** and will not detect the **DID** being used in multiple **DID** or dynamically defined **DID** requests, while the **DID** list element is verified in all services requesting the **DID**.

[SWS_Dcm_01525] White list definition for **DIDs**

Upstream requirements: [RS_Diag_04233](#), [RS_Diag_04234](#)

[The white list for **DIDs** defines the set of data identifiers that are allowed to be read, written and controlled. Each entry shall contain in the first two bytes the **DID** number and in the third byte the following access definitions:

- Bit 0: Read access
- Bit1: Write access
- Bit2: Control access (by InputOutputControlByIdentifier)

A bit value of 0 means that the operation is forbidden, a bit value of 1 means that the operation is allowed. All not used bits (3-7) shall be set to zero.

DID numbers are always in big endian format (MSB first).]

Example:

DID access record	Grants access to
0x1A90 0x01	All read data by identifier operations for DID 0x1A90
0x314F 0x05	All read data by identifier and IOcontrol operations for DID 0x314F

DID read operations are performed from various **UDS** services. **Dcm** checks on each **DID** read the access to that **DID**.

[SWS_Dcm_01526] White list definition for **RIDs**

Upstream requirements: [RS_Diag_04233](#), [RS_Diag_04234](#)

[The white list for **RIDs** defines the set of routine identifiers that have access to the sub-functions startRoutine, stopRoutine and requestRoutineResult. Each entry shall contain in the first two bytes the **RID** number and in the third byte the following access definitions:

- Bit 0: Access to sub-function 'startRoutine'
- Bit1: Access to sub-function 'stopRoutine'
- Bit2: Access to sub-function 'requestRoutineResult'

A bit value of 0 means that the sub-function is forbidden, a bit value of 1 means that the sub-function is allowed. All not used bits (3-7) shall be set to zero.

RID numbers are always in big endian format (MSB first).]

Example:

RID access record	Grants access to
0x0240 0x01	StartRoutine is allowed for RID 0x0240
0x028A 0x07	All routine sub-functions are allowed for RID 0x28A

[SWS_Dcm_01527] White list definition for MemorySelection

Upstream requirements: [RS_Diag_04233](#), [RS_Diag_04234](#)

[The white list memory selection is used for the UDS service 0x19 with sub-functions 0x17, 0x18 and 0x19. The value in the white list defines the accepted parameter values for MemorySelection in the UDS request.]

Transition into authenticated session

[SWS_Dcm_01528] Transition into authenticated state

Upstream requirements: [RS_Diag_04230](#)

[If the proof of ownership verification in [\[SWS_Dcm_01511\]](#) was successful and resulted in CRYPTO_E_VER_OK and after the role and white list setting was done, the Dcm shall set the DcmAuthenticationState_<ConnectionName> on that connection into DCM_AUTHENTICATED.]

[SWS_Dcm_01529] Successful ownership verification

Upstream requirements: [RS_Diag_04230](#)

[If the result of Csm_SignatureVerify from [\[SWS_Dcm_01511\]](#) is Crypto_VerifyResultType equal to CRYPTO_E_VER_OK, the Dcm shall send a positive response with authenticationReturnParameter set to 'Ownership verified, Authentication complete' and providing a size of 0 for lengthOfSessionKeyInfo and no sessionKeyInfo.]

[SWS_Dcm_01530] Persisting authentication state

Upstream requirements: [RS_Diag_04230](#)

[If the result of Csm_SignatureVerify from [\[SWS_Dcm_01511\]](#) is Crypto_VerifyResultType equal to CRYPTO_E_VER_OK and the mode rule referenced by [DcmDspAuthenticationPersistStateModeRuleRef](#) is evaluated to true, the Dcm shall persist the authentication state, role and white list on that connection.]

[SWS_Dcm_01531] Activation of role and white list [The Dcm shall consider the role and white list for access control only, if the Dcm is in authenticated state.]

[SWS_Dcm_01532] Re-entering authenticated state

Upstream requirements: [RS_Diag_04230](#)

[If the [Dcm](#) is already in authenticated state while a transition to authentication stated is requested according to [\[SWS_Dcm_01529\]](#), the [Dcm](#) shall overwrite the roles and white list and use only the role and white list from the last received certificate.]

7.4.2.10.6 AuthenticationConfiguration

[SWS_Dcm_01533] Providing the authentication configuration

Upstream requirements: [RS_Diag_04230](#)

[If [DcmDspAuthentication](#) is configured and an Authentication (0x29) service with sub-function equal to authenticationConfiguration is received, the [Dcm](#) shall send a positive response with authenticationReturnParameter set to 'AuthenticationConfiguration APCE'.]

7.4.2.10.7 TransmitCertificate

The subfunction transmitCertificate (0x4) offers a mechanism to transmit general-purpose certificates to the ECU. The [Dcm](#) does not directly interpret such certificates but allows the application to use them. In general, the semantics is out of the scope of the [Dcm](#). Typical examples are:

- Secure coding certificates - authenticates data written by WBDI
- SecOC certificates
- TLS ECU certificates
- Enhanced diagnostic rights certificates

More than one of these certificates can be supported by the [Dcm](#). The key to identifying the certificate type is the certificateEvaluationId parameter from the UDS request message. A [DcmDspAuthenticationTransmitCertificate](#) provides the configuration information on how the certificates are processed and which application callback is used to inform about successful certificate transmission.

[SWS_Dcm_01681] Support of subfunction transmitCertificate [The [Dcm](#) shall support the subfunction transmitCertificate of the UDS service Authentication according to ISO 14229-1:2020 [\[9\]](#).]

[SWS_Dcm_01682] Verify certificateEvaluationId

Upstream requirements: [RS_Diag_04271](#)

[If the UDS service Authentication (0x29) service with sub-function equal to transmitCertificate is received and the certificateEvaluationId has no configured [DcmDspAuthenticationTransmitCertificateEvaluationId](#) in any [DcmDspAuthenticationTransmitCertificate](#) containers, the [Dcm](#) shall return a NRC 0x31 (requestOutOfRange).]

The [Dcm](#) uses the KeyM for all certificate management operations. This document does only mention the KeyM and a configuration reference to a KeyMCertificate. For details on how the KeyM module is used please read the KeyM SWS document. Be aware that cryptographic operations have potentially long execution times and are called asynchronously.

[SWS_Dcm_01683] Certificate storage parsing and verification

Upstream requirements: [RS_Diag_04271](#)

[The [Dcm](#) shall store, parse, and verify a certificate in the KeyM module, using the KeyM destination [DcmDspAuthenticationTransmitCertificateRef](#) for the received certificateEvaluationId.]

After the certificate was processed successfully according to [\[SWS_Dcm_01683\]](#) the certificate is correctly stored in the server and the application is informed by KeyM about the existence of a new certificate.

7.4.2.11 Service 0x2A - ReadDataByPeriodicIdentifier

[SWS_Dcm_01613]

Upstream requirements: [RS_Diag_04215](#)

[The [Dcm](#) shall support the UDS service ReadDataByPeriodicIdentifier (0x2A) with all transmissionMode types according to ISO 14229-1:2020 [\[9\]](#).]

[SWS_Dcm_01552] Read Periodic [DID](#) authentication check for statically defined [DIDs](#)

Upstream requirements: [RS_Diag_04233](#)

[On reception of the UDS Service ReadDataByPeriodicIdentifier (0x2A), the [Dcm](#) shall check if the access to all static defined requested [DIDs](#) is authenticated and read the data identifiers periodically only if:

- a [DID](#) read role is configured via [DcmDspDidReadRoleRef](#) for that [DID](#) and the verification according to [\[SWS_Dcm_01522\]](#) was successful or
- the active white list on that connection has for each requested DID one entry with read access that matches that DID.

]

[SWS_Dcm_01553] Read Periodic DID authentication check for dynamically defined DIDs

Upstream requirements: [RS_Diag_04233](#)

[On reception of the UDS Service ReadDataByPeriodicIdentifier (0x2A), the Dcm shall check if the access to all dynamically defined requested DIDs if the content is based of DIDs or parts of DIDs is authenticated and read the data identifiers periodically only if:

- a DID read role is configured via [DcmDspDidReadRoleRef](#) for that DID and the verification according to [\[SWS_Dcm_01522\]](#) was successful or
- the active white list on that connection has for each requested DID one entry with read access that matches that DID.

]

According to [\[SWS_Dcm_01537\]](#) the authentication checks are only executed if [DcmDspAuthentication](#) is configured. In case of a failed authentication the NRC handling is according to [\[SWS_Dcm_01544\]](#) and [\[SWS_Dcm_01551\]](#) applies.

[SWS_Dcm_00721] Session check for ReadDataByPeriodicIdentifier DID

Upstream requirements: [RS_Diag_04215](#)

[On reception of the UDS Service ReadDataByPeriodicIdentifier (0x2A), for every requested periodicDIDs, the Dcm module shall check if the periodicDID can be read in the current session (see configuration parameter [DcmDspDidReadSessionRef](#)). If none of the periodicDID can be read in the current session, the Dcm module shall send a NRC 0x31 (RequestOutOfRange).]

[SWS_Dcm_00722] Security level check for ReadDataByPeriodicIdentifier DID

Upstream requirements: [RS_Diag_04215](#)

[On reception of the UDS Service ReadDataByPeriodicIdentifier (0x2A), for every requested periodicDIDs, the Dcm module shall check if the periodicDID can be read in the current security level (see configuration parameter [DcmDspDidReadSecurityLevelRef](#)). If not, the Dcm module shall send NRC 0x33 (Security access denied).]

Note: To evaluate the session and security assignments the Dcm evaluates the configuration that is used for ReadDataByIdentifier, e.g. [DcmDspDidReadSessionRef](#) or [DcmDspDidReadSecurityLevelRef](#).

[SWS_Dcm_00820] Mode condition check for ReadDataByPeriodicIdentifier DID

Upstream requirements: [RS_Diag_04215](#)

[On reception of the UDS Service ReadDataByPeriodicIdentifier (0x2A), for every requested periodicDIDs, the Dcm module shall check if the periodicDID can be read in the current mode condition (see configuration parameter [DcmDspDidReadModeRuleRef](#)). If not, the Dcm module shall send the calculated negative response code of the reference [DcmModeRule](#)]

[SWS_Dcm_01097] Checks for DDDIDs in ReadDataByPeriodicIdentifier

Upstream requirements: [RS_Diag_04215](#)

[On reception of the [UDS](#) Service ReadDataByPeriodicIdentifier (0x2A), if verification has been successfully done ([[SWS_Dcm_00721](#)], [[SWS_Dcm_00722](#)] and [[SWS_Dcm_00820](#)]), and if the request contains one or more dynamically defined DID(s), the [Dcm](#) module shall do the session, security and mode dependencies checks for all source data in case the configuration parameter [DcmDspDDDIDcheckPerSourceDID](#) is set to TRUE.]

[SWS_Dcm_01098] Condition check for ReadDataByPeriodicIdentifier DID

Upstream requirements: [RS_Diag_04215](#)

[On reception of the [UDS](#) Service ReadDataByPeriodicIdentifier (0x2A), for every requested periodicDIDs, the [Dcm](#) module shall invoke the ConditionCheckRead operation (or the respective C-Function) if configured. In case of a negative result, the returned ErrorCode shall be used as final negative response code.]

[SWS_Dcm_01099] Support of dynamic length DIDs

Upstream requirements: [RS_Diag_04215](#)

[On reception of the [UDS](#) Service ReadDataByPeriodicIdentifier (0x2A), for every requested periodicDIDs, with a configured dynamic length the [Dcm](#) module shall invoke the ReadDataLength operation (or the respective C-Function) to retrieve the length of the periodicDID. This length is valid for each ReadData operation till the periodicDID is removed from the scheduler or updated via a new request. This length shall further be used to check against the UUDT size.]

[SWS_Dcm_00843] Ensuring maxim number of periodic DIDs

Upstream requirements: [RS_Diag_04215](#)

[On reception of the [UDS](#) Service ReadDataByPeriodicIdentifier (0x2A), the [Dcm](#) module shall check if the periodicDataIdentifiers requested in a single request do not exceed the configured [DcmDspMaxPeriodicDidToRead](#) (maximum length check). Otherwise (in case the number of elements is exceeded) the [Dcm](#) module shall send a [NRC 0x13](#) (Incorrect message length or invalid format).]

[SWS_Dcm_01096] Behavior on unused DIDs

Upstream requirements: [RS_Diag_04215](#)

[If [DcmDspDidUsed](#) is set to FALSE, the [Dcm](#) shall consider the [DID](#) as not supported.]

7.4.2.11.1 Scheduler PeriodicTransmission

Note: The periodic response message layout is according to ISO 14229-1:2020 [9]. It contains the periodic data identifier and the data. The service [ID](#) and A_PCI byte is not part of the periodic response message.

[SWS_Dcm_01101]

Upstream requirements: [RS_Diag_04215](#)

[The [Dcm](#) shall send all periodic scheduled response by using the configured [DcmDspPeriodicConnections](#).]

ISO 14229-1:2020 [9] defines two distinct scheduler types for ReadDataByPeriodicIdentifier. Depending on the scheduler type, the [Dcm](#) has different transmission strategies. The used scheduler type of the server is selected by setting [DcmDspPeriodicTransmissionSchedulerType](#) to one of the two types.

[SWS_Dcm_01568] Support of periodic transmission type 1

Upstream requirements: [RS_Diag_04215](#)

[If [DcmDspPeriodicTransmissionSchedulerType](#) is set to SCHEDULER_TYPE1 and the [Dcm](#) is set up to send PDIDs, the [Dcm](#) shall trigger the transmission of all scheduled PDIDs each time the configured periodic transmission rate elapses.]

[SWS_Dcm_01569] Transmission of one PIDID per available connection

Upstream requirements: [RS_Diag_04215](#)

[If PIDID transmission is triggered according to [\[SWS_Dcm_01568\]](#), [Dcm](#) shall transmit one PIDID per available periodic connection starting with the first PIDID in the list of scheduled PDIDs.]

[SWS_Dcm_01570] Continue in next main function if more PIDIDs than available connections

Upstream requirements: [RS_Diag_04215](#)

[If PIDID transmission is triggered according to [\[SWS_Dcm_01568\]](#) and more PIDIDs are to transmit than the number of available periodic connections, the [Dcm](#) shall continue the transmission in the next [Dcm](#) main function call. Each time the available periodic connections shall be used.]

[SWS_Dcm_01571] Overlapping triggers and transmissions

Upstream requirements: [RS_Diag_04215](#)

[If PIDID transmission is triggered according to [\[SWS_Dcm_01568\]](#) or [\[SWS_Dcm_01572\]](#) and the next periodic transmission rate trigger occurs before the [Dcm](#) has finished sending the scheduled PDIDs according to [\[SWS_Dcm_01570\]](#), the [Dcm](#) shall continue to transmit PDIDs according to [\[SWS_Dcm_01570\]](#) and schedule the PDIDs of the new trigger after the last previous PIDID has been transmitted.]

[SWS_Dcm_01572] Support of periodic transmission type 2

Upstream requirements: [RS_Diag_04215](#)

[If [DcmDspPeriodicTransmissionSchedulerType](#) is set to SCHEDULER_TYPE2 the [Dcm](#) is set up to send PDIDs and the periodic transmission

rate for a scheduler elapses, the [Dcm](#) shall trigger the transmission of the next scheduled PDIDs on all available periodic connections.]

[SWS_Dcm_01573] PDID transmission sequence for scheduler type 2

Upstream requirements: [RS_Diag_04215](#)

[For scheduler type 2 transmission of PDIDs within a single elapse of the scheduler, the [Dcm](#) shall start to transmit the first configured PDID and continue to transmit consecutively all other configured PDIDs. If the last PDID is transmitted, the [Dcm](#) restarts the sequence with the first configured PDID. If a PDID is sent more than once within a single elapse of the scheduler then the PDIDs contents shall be reloaded with the applicable [DID](#) / Memory data for every re-transmission.]

[SWS_Dcm_01574] Continue in next periodic rate scheduler if more PDIDs than available connections

Upstream requirements: [RS_Diag_04215](#)

[If a scheduler type 2 PDID transmission is active according to [\[SWS_Dcm_01572\]](#) and the number of scheduled PDIDs is larger than the number of available periodic connections, the [Dcm](#) shall continue to send the next scheduled PDIDs the next time the periodic transmission rate scheduler elapses.]

The following example gives an overview about scheduler type 1 and type 2. They use PDIDs starting at 0xF201 and having all PDID data set to zero. The PDIDs are send on CAN 2.0, using the CAN-IDs 0x6A0, 0x6A1.

Example 1: Scheduler Type 1

```
1 SchedulerRate:      1000ms
2
3 PeriodicRate:       4000ms
4
5 NumPeriodicAddr:    1
6
7 NumPDID              2
8
9 1017.02 6A0 01 00 00 00 00 00 00 00
10
11 2016.97 6A0 02 00 00 00 00 00 00 00
12
13
14
15 5016.87 6A0 01 00 00 00 00 00 00 00
16
17 6016.78 6A0 02 00 00 00 00 00 00 00
18
19
20
21 9016.67 6A0 01 00 00 00 00 00 00 00
22
23 10016.63 6A0 02 00 00 00 00 00 00 00
24
25
```

```

26
27 13016.52 6A0 01 00 00 00 00 00 00 00
28
29 14016.46 6A0 02 00 00 00 00 00 00 00

```

Example 2: Scheduler Type 2

```

1 SchedulerRate:      1000ms
2
3 PeriodicRate:      4000ms
4
5 NumPeriodicAddr:   1
6
7 NumPDID            2
8
9 945.01 6A0 01 00 00 00 00 00 00 00
10
11 4944.78 6A0 02 00 00 00 00 00 00 00
12
13 8944.66 6A0 01 00 00 00 00 00 00 00
14
15 12944.49 6A0 02 00 00 00 00 00 00 00
16
17 16944.33 6A0 01 00 00 00 00 00 00 00
18
19 20944.28 6A0 02 00 00 00 00 00 00 00
20
21 24944.01 6A0 01 00 00 00 00 00 00 00

```

Example 3: Aliasing with Scheduler Type 1

```

1 SchedulerRate:      1000ms
2
3 PeriodicRate:      4000ms
4
5 NumPeriodicAddr:   1
6
7 NumPDID            5
8
9
10 529.89 6A0 01 00 00 00 00 00 00 00
11
12 1529.70 6A0 02 00 00 00 00 00 00 00
13
14 2529.59 6A0 03 00 00 00 00 00 00 00
15
16 3529.35 6A0 04 00 00 00 00 00 00 00
17
18
19
20 4529.16 6A0 05 00 00 00 00 00 00 00 <-- Alias (should start over at
    01)
21
22 5529.02 6A0 01 00 00 00 00 00 00 00
23
24 6529.15 6A0 02 00 00 00 00 00 00 00

```

```

25
26 7529.03 6A0 03 00 00 00 00 00 00 00
27
28
29
30 8528.92 6A0 04 00 00 00 00 00 00 00 <-- Alias (should start over at
    01)
31
32 9528.82 6A0 05 00 00 00 00 00 00 00
33
34 10528.46 6A0 01 00 00 00 00 00 00 00
35
36 11528.60 6A0 02 00 00 00 00 00 00 00
37
38
39
40 12528.47 6A0 03 00 00 00 00 00 00 00 <-- Alias (should start over at
    01)
41
42 13528.36 6A0 04 00 00 00 00 00 00 00
43
44 14528.27 6A0 05 00 00 00 00 00 00 00

```

Example 4: Scheduler Type 2

```

1 SchedulerRate:      1000ms
2
3 PeriodicRate:      4000ms
4
5 NumPeriodicAddr:   1
6
7 NumPDID            5
8
9
10 592.03 6A0 01 00 00 00 00 00 00 00
11
12 4591.56 6A0 02 00 00 00 00 00 00 00
13
14 8591.13 6A0 03 00 00 00 00 00 00 00
15
16 12590.69 6A0 04 00 00 00 00 00 00 00
17
18 16590.26 6A0 05 00 00 00 00 00 00 00
19
20 20589.78 6A0 01 00 00 00 00 00 00 00
21
22 24589.23 6A0 02 00 00 00 00 00 00 00
23
24 28588.90 6A0 03 00 00 00 00 00 00 00
25
26 32588.45 6A0 04 00 00 00 00 00 00 00
27
28 36588.05 6A0 05 00 00 00 00 00 00 00

```

Example 5: Scheduler 1 with Multiple Periodic Addresses

```

1 SchedulerRate:      1000ms

```

```

2
3 PeriodicRate:          4000ms
4
5 NumPeriodicAddr:    2
6
7 NumPDID              5
8
9
10
11    778.69 6A0 01 00 00 00 00 00 00 00
12
13    778.73 6A1 02 00 00 00 00 00 00 00
14
15
16
17   1778.56 6A0 03 00 00 00 00 00 00 00
18
19   1778.61 6A1 04 00 00 00 00 00 00 00
20
21
22
23   2778.44 6A0 05 00 00 00 00 00 00 00
24
25
26
27   4778.24 6A0 01 00 00 00 00 00 00 00
28
29   4778.27 6A1 02 00 00 00 00 00 00 00
30
31
32
33   5778.08 6A0 03 00 00 00 00 00 00 00
34
35   5778.12 6A1 04 00 00 00 00 00 00 00
36
37
38
39   6778.12 6A0 05 00 00 00 00 00 00 00
40
41
42   8777.90 6A0 01 00 00 00 00 00 00 00
43
44   8777.93 6A1 02 00 00 00 00 00 00 00
45
46
47
48   9777.70 6A0 03 00 00 00 00 00 00 00
49
50   9777.73 6A1 04 00 00 00 00 00 00 00
51
52
53
54  10777.57 6A0 05 00 00 00 00 00 00 00

```

Example 6: Scheduler 2 with Multiple Periodic Addresses

```

1 SchedulerRate:      1000ms
2
3 PeriodicRate:      4000ms
4
5 NumPeriodicAddr:    2
6
7 NumPDID             5
8
9
10
11
12   764.64 6A0  01 00 00 00 00 00 00 00
13
14   764.69 6A1  02 00 00 00 00 00 00 00
15
16
17
18   4762.79 6A0  03 00 00 00 00 00 00 00
19
20   4762.82 6A1  04 00 00 00 00 00 00 00
21
22
23
24   8762.35 6A0  05 00 00 00 00 00 00 00
25
26   8762.36 6A1  01 00 00 00 00 00 00 00
27
28
29
30  12762.01 6A0  02 00 00 00 00 00 00 00
31
32  12762.04 6A1  03 00 00 00 00 00 00 00

```

Example 7: Scheduler 2 with 2 Periodic Addresses and 1 PDID

```

1 SchedulerRate:      1000ms
2
3 PeriodicRate:      1000ms
4
5 NumPeriodicAddr:    2
6
7 NumPDID             1
8
9
10
11   7.19 6A0  01 00 00 00 00 00 00 00
12
13   77.21 6A1  01 00 00 00 00 00 00 00
14
15
16
17  1077.61 6A0  01 00 00 00 00 00 00 00
18
19  1077.68 6A1  01 00 00 00 00 00 00 00
20
21

```



```

22
23 2078.29 6A0 01 00 00 00 00 00 00 00
24
25 2078.40 6A1 01 00 00 00 00 00 00 00
26
27
28
29 3078.26 6A0 01 00 00 00 00 00 00 00
30
31 3078.33 6A1 01 00 00 00 00 00 00 00
32
33
34
35 4079.13 6A0 01 00 00 00 00 00 00 00
36
37 4079.19 6A1 01 00 00 00 00 00 00 00
38
39
40
41 5079.17 6A0 01 00 00 00 00 00 00 00
42
43 5079.24 6A1 01 00 00 00 00 00 00 00
44
45
46
47 6079.75 6A0 01 00 00 00 00 00 00 00
48
49 6079.82 6A1 01 00 00 00 00 00 00 00
50
51
52
53 7080.77 6A0 01 00 00 00 00 00 00 00
54
55 7080.91 6A1 01 00 00 00 00 00 00 00
56
57
58
59 8080.90 6A0 01 00 00 00 00 00 00 00
60
61 8080.97 6A1 01 00 00 00 00 00 00 00
62
63
64
65 9081.09 6A0 01 00 00 00 00 00 00 00
66
67 9081.15 6A1 01 00 00 00 00 00 00 00

```

[SWS_Dcm_01103] Require previous confirmation before next transmission

Upstream requirements: [RS_Diag_04215](#)

[The `Dcm` shall trigger a transmission `DcmDslPeriodicTxPduRef` only after the `DcmDslPeriodicTxConfirmationPduId` transmit confirmation for the previously transmitted periodic response is received.]

[SWS_Dcm_01104] Transmission error behavior

Upstream requirements: [RS_Diag_04215](#)

[In case of a PDID transmission error, the [Dcm](#) shall use always the same order of periodicDIDs per client. Transmission errors shall not influence this order, the [Dcm](#) shall continue to retry the transmission and start the next PDID only after the PDID was transmitted successfully.]

[SWS_Dcm_01105] No negative response for periodic [DID](#) after positive response

Upstream requirements: [RS_Diag_04215](#)

[After periodicDIDs are started and a positive response was sent, the [Dcm](#) shall start to send the periodic [DID](#).]

[SWS_Dcm_01106] Sending periodic [DIDs](#) without further condition checks

Upstream requirements: [RS_Diag_04215](#)

[Each time the counter of a periodicDataIdentifiers elapses, the [Dcm](#) shall retrieve the [DID](#) data without validating further conditions (i.e. session, security, mode dependencies, ConditionCheckRead and ReadDataLength).]

Note: The rate for a specific transmissionMode is defined as the time between any two consecutive response messages with the same periodicDataIdentifier, when only a single periodicDID is scheduled. If multiple periodicDIDs are scheduled concurrently, the effective period between the same periodicDataIdentifier will vary based upon the following design parameters:

- The main function recurrence [DcmTaskTime](#)
- The number of available periodic connections
- The number of periodicDIDs that can be scheduled concurrently (see configuration parameter [DcmDspMaxPeriodicDidScheduler](#)).

[SWS_Dcm_01575] Connection limitation for fast transmission

Upstream requirements: [RS_Diag_04215](#)

[If [DcmDspPeriodicTransmissionMaxPeriodicFastTransmissions](#) is configured, the [Dcm](#) shall limit the number of used periodic connections for transmissionMode 'sendAtFastRate' to [DcmDspPeriodicTransmissionMaxPeriodicFastTransmissions](#) connections.]

[SWS_Dcm_01576] Connection limitation for medium transmission

Upstream requirements: [RS_Diag_04215](#)

[If [DcmDspPeriodicTransmissionMaxPeriodicMediumTransmissions](#) is configured, the [Dcm](#) shall limit the number of used periodic connections for transmissionMode 'sendAtMediumRate' to [DcmDspPeriodicTransmissionMaxPeriodicMediumTransmissions](#) connections.]

[SWS_Dcm_01577] Connection limitation for slow transmission

Upstream requirements: [RS_Diag_04215](#)

[If [DcmDspPeriodicTransmissionMaxPeriodicSlowTransmissions](#) is configured, the [Dcm](#) shall limit the number of used periodic connections for transmission-Mode 'sendAtSlowRate' to [DcmDspPeriodicTransmissionMaxPeriodicSlowTransmissions](#) connections.]

[SWS_Dcm_01578] Connection order for limited transmissions

Upstream requirements: [RS_Diag_04215](#)

[If any of the transmission connection limitations according to [\[SWS_Dcm_01575\]](#), [\[SWS_Dcm_01576\]](#) or [\[SWS_Dcm_01577\]](#) is configured, the [Dcm](#) shall select the [DcmDslPeriodicConnection](#) in the order of definition. That means that the first transmitted periodic [DID](#) shall take the first configured connections and so forth.]

[SWS_Dcm_01111] Periodic DDDID transmission and session change

Upstream requirements: [RS_Diag_04215](#)

[If [DcmDspDDDIDcheckPerSourceDID](#) is set to TRUE upon a session change, the [Dcm](#) shall stop any scheduled periodic DDDID, that contains source data not supported in the current session.]

[SWS_Dcm_01112] Periodic DDDID transmission and security level change

Upstream requirements: [RS_Diag_04215](#)

[If [DcmDspDDDIDcheckPerSourceDID](#) is set to TRUE, upon any security level change, the [Dcm](#) shall stop any scheduled periodic DDDID that contains source data not supported in the current security level.]

[SWS_Dcm_01115] Cancel pending read operations

Upstream requirements: [RS_Diag_04215](#)

[Upon stopping the periodical read of a data identifier while a pending asynchronous ReadData operation is still in progress, the [Dcm](#) shall cancel the pending read operation by calling ReadData with OpStatus=DCM_CANCEL.]

[SWS_Dcm_01117] Stop cancels pending transmissions

Upstream requirements: [RS_Diag_04215](#)

[Upon stopping the periodical read of a data identifier, the [Dcm](#) shall cancel pending transmissions of that [DID](#).]

[SWS_Dcm_01118] Finalizing initiated transmissions upon stop

Upstream requirements: [RS_Diag_04215](#)

[Upon stopping the periodical read of a data identifier the [Dcm](#) shall finalize already initiated transmissions of that [DID](#).]

7.4.2.12 Service 0x2C - DynamicallyDefineDataIdentifier

[SWS_Dcm_00259] [The [DSP](#) submodule shall implement the DynamicallyDefineDataIdentifier (service 0x2C, diagnostic data access) of the Unified Diagnostic Services.]

The DynamicallyDefineDataIdentifier service is implemented internally in [Dcm](#) module.

[SWS_Dcm_00866] [If [DcmDDDIDStorage](#) configuration parameter is set to FALSE, the [Dcm](#) shall initialize all DDDIDs as not present at power-up [Dcm_Init](#)).]

[SWS_Dcm_00867] [If [DcmDDDIDStorage](#) configuration parameter is set to TRUE, the [Dcm](#) shall restore the DDDID definition from NvM at power-up [Dcm_Init](#)).]

[SWS_Dcm_00868] [If [DcmDDDIDStorage](#) configuration parameter is set to TRUE, the [Dcm](#) shall trigger the storage of the DDDID definition to NvRam (via [NvM_SetRamBlockStatus](#)).]

[SWS_Dcm_00646] [On reception of service DynamicallyDefineDataIdentifier with subservice [defineByIdentifier](#) or [defineByMemoryAddress](#), the [Dcm](#) module shall configure this new [DID](#) with associated information receive from the diagnostic request: Memory address and memory length or [DID](#) source, position and size.]

[SWS_Dcm_00861] [On reception of the [UDS](#) Service DynamicallyDefineDataIdentifier (0x2C), the [Dcm](#) module shall check if the DDDID will not exceed the configured parameter value [DcmDspDDDIDMaxElements](#). Otherwise (in case the number of elements will be exceeded) the [Dcm](#) module shall send a [NRC](#) 0x31 (RequestOutOfRange).]

[SWS_Dcm_00854] [On reception of the [UDS](#) Service DynamicallyDefineDataIdentifier (0x2C) with subservice [defineByMemoryAddress](#), the [Dcm](#) shall check if the requested [AddressAndLengthFormatIdentifier](#) is supported (refer to configuration parameter [DcmDspSupportedAddressAndLengthFormatIdentifier](#)), Otherwise the [NRC](#) 0x31 (requestOutOfRange) shall be responded. In case the container [AddressAndLengthFormatIdentifier](#) is not present, the [Dcm](#) shall accept all possible AddressAndLengthFormatIdentifiers.]

[SWS_Dcm_00647] [On reception of service DynamicallyDefineDataIdentifier with subservice [clearDynamicallyDefinedDataIdentifier](#), the [Dcm](#) module shall remove the configuration of this DID.]

[SWS_Dcm_00723] [On reception of the [UDS](#) Service DynamicallyDefineDataIdentifier (0x2C), the [Dcm](#) module shall check if the DDDID can be defined in the current session (see configuration parameter [DcmDspDidReadSessionRef](#)). If not, the [Dcm](#) module shall send a [NRC](#) 0x31 (RequestOutOfRange).]

[SWS_Dcm_00724] [On reception of the [UDS](#) Service DynamicallyDefineDataIdentifier (0x2C), the [Dcm](#) module shall check if the DDDID can be defined in the current

security level (see configuration parameter [DcmDspDidReadSecurityLevelRef](#)). If not, the [Dcm](#) module shall send [NRC 0x33](#) (Security access denied).]

[SWS_Dcm_00725] [On reception of the [UDS](#) Service DynamicallyDefineDataIdentifier (0x2C), the [Dcm](#) module shall check if the requested Source-DIDs are supported in the current session (see configuration parameter of referenced [DID DcmDspDidReadSessionRef](#)). If not, the [Dcm](#) module shall send a [NRC 0x31](#) (RequestOutOfRange).]

[SWS_Dcm_00726] [On reception of the [UDS](#) Service DynamicallyDefineDataIdentifier (0x2C), the [Dcm](#) module shall check if the requested Source-DID or the memoryRange are supported in the current security level (see configuration parameter of referenced [DID DcmDspDidReadSecurityLevelRef](#) or memoryRange [DcmDspReadMemoryRangeSecurityLevelRef](#)). If not, the [Dcm](#) module shall send a [NRC 0x33](#) (Security access denied).]

[SWS_Dcm_00821] [On reception of the [UDS](#) Service DynamicallyDefineDataIdentifier (0x2C), the [Dcm](#) module shall check if the requested Source-DID or the memoryRange are supported in the current mode condition (see configuration parameter of referenced [DID DcmDspDidReadModeRuleRef](#) or memoryRange [DcmDspReadMemoryRangeModeRuleRef](#)). If not, the [Dcm](#) module shall send the calculated negative response code of the referenced [DcmModeRule](#).]

In case of memory address(es), on reception of [ReadDataByIdentifier](#) or [ReadDataByPeriodicIdentifier](#) request for a dynamically defined DID, the [Dcm](#) will use the callout [Dcm_ReadMemory](#) for all contained memory addresses to access the data.

[SWS_Dcm_01051] [On reception of the [UDS](#) Service DynamicallyDefineDataIdentifier (0x2C), if the request message contains different MemoryIdValue compare to the configured values in [DcmDspMemoryIdInfo](#) container, the [Dcm](#) shall send a [NRC 0x31](#) (RequestOutOfRange).]

In case of [DID](#) source(s), on reception of [ReadDataByIdentifier](#) or [ReadDataByPeriodicIdentifier](#) request for a dynamically defined DID, the [Dcm](#) will use the configuration of the contained DIDs to read the data.

7.4.2.13 Service 0x2E - WriteDataByIdentifier

[SWS_Dcm_00255] [The [Dcm](#) module shall implement the [UDS](#) Service WriteDataByIdentifier (0x2E) of the Unified Diagnostic Services.]

When using Service 0x2E, the request of the tester contains a 2-byte [DID](#) and a dataRecord with the data to be written. The configuration of the [Dcm](#) contains a list of supported DIDs and defines for each configured DID:

- The 2-byte [DID](#) (see configuration parameter [DcmDspDidIdentifier](#))
- For every data of the DID:

- The function `WriteData` to be used for this data (see configuration parameters `DcmDspDataWriteFnc` and `DcmDspDataUsePort`)

[SWS_Dcm_01496] Write UDS DID authentication check

Upstream requirements: [RS_Diag_04233](#)

[On reception of the UDS Service `WriteDataByIdentifier` (0x2E), the Dcm shall check if the write access to the requested DID is authenticated and write the data identifier only if:

- for that write `DID` a role is configured via `DcmDspDidWriteRoleRef` and the verification according to [\[SWS_Dcm_01522\]](#) was successful or
- the active white list on that connection has for the requested DID one entry with write access that matches that DID.

]

According to [\[SWS_Dcm_01537\]](#) the authentication checks are only executed if `DcmDspAuthentication` is configured. In case of a failed authentication the NRC handling is according to [\[SWS_Dcm_01544\]](#) and [\[SWS_Dcm_01551\]](#) applies.

[SWS_Dcm_00467] [On reception of the UDS Service `WriteDataByIdentifier` (0x2E), the Dcm module shall check if the `DID` is supported (see configuration parameter `DcmDspDid` and `DcmDspDidRange`) If not, the Dcm module shall send NRC 0x31 (Request out of range) .]

[SWS_Dcm_00562] [If a `DID` is set as unused (`DcmDspDidUsed` set to FALSE), the Dcm shall consider the `DID` as not supported (according to [\[SWS_Dcm_00467\]](#))]

[SWS_Dcm_00468] [On reception of the UDS Service `WriteDataByIdentifier` (0x2E), the Dcm module shall check if the `DID` has a Write access configured (see configuration parameter `DcmDspDidWrite` in `DcmDspDidInfo`). If not, the Dcm module shall send NRC 0x31 (Request out of range).]

[SWS_Dcm_00469] [On reception of the UDS Service `WriteDataByIdentifier` (0x2E), the Dcm module shall check if the `DID` can be written in the current session (see configuration parameter `DcmDspDidWriteSessionRef`). If not, the Dcm module shall send a NRC 0x31 (Request Out of Range).]

[SWS_Dcm_00470] [On reception of the UDS Service `WriteDataByIdentifier` (0x2E), the Dcm module shall check if the `DID` can be written in the current security level (see configuration parameter `DcmDspDidWriteSecurityLevelRef`). If not, the Dcm module shall send NRC 0x33 (Security access denied).]

[SWS_Dcm_00822] [On reception of the UDS Service `WriteDataByIdentifier` (0x2E), the Dcm module shall check if the `DID` can be written in the current mode condition (see configuration parameter `DcmDspDidWriteModeRuleRef`). If not, the Dcm module shall send the calculated negative response code of the referenced `DcmModeRule`.]

[SWS_Dcm_00473] [On reception of the UDS Service WriteDataByIdentifier (0x2E), if all signals (*DcmDspDidSignal*) of the DID have fixed length (*DcmDspDataType* is different than *UINT8_DYN*), the Dcm module shall check if the received data length corresponds to the DID data length (addition of all *DcmDspDataByteSize*).]

[SWS_Dcm_00395] [After all verifications (see [\[SWS_Dcm_00467\]](#), [\[SWS_Dcm_00468\]](#), [\[SWS_Dcm_00469\]](#), [\[SWS_Dcm_00470\]](#), [\[SWS_Dcm_00473\]](#)) the Dcm module shall write all the signals (*DcmDspDidSignal*) of the DID by either calling the configured function *DcmDspDataWriteFnc* (if parameter *DcmDspDataUsePort* is set to *USE_DATA_SYNCH_FNC* or *USE_DATA_ASYNCH_FNC* or *USE_DATA_ASYNCH_FNC_ERROR* or *USE_DATA_SYNCH_FNC_PROXY* or *USE_DATA_ASYNCH_FNC_PROXY*) or the associated WriteData operations (if parameter *DcmDspDataUsePort* is set to *USE_DATA_SYNCH_CLIENT_SERVER* or *USE_DATA_ASYNCH_CLIENT_SERVER* or *USE_DATA_ASYNCH_CLIENT_SERVER_ERROR*) or the associated SenderReceiver interfaces (if parameter *DcmDspDataUsePort* is set to *USE_DATA_SENDER_RECEIVER* or to *USE_DATA_SENDER_RECEIVER_AS_SERVICE*) with the following parameter values:

Data: the dataRecord from the request

DataLength: the number of bytes in the dataRecord (get from the configuration if the data has fixed length (*DcmDspDataType* is different than *UINT8_DYN*) or from the diagnostic request length if the data has dynamic length (*DcmDspDataType* is set to *UINT8_DYN*)).]

[SWS_Dcm_01433] [After all verifications (see [\[SWS_Dcm_00467\]](#), [\[SWS_Dcm_00468\]](#), [\[SWS_Dcm_00469\]](#), [\[SWS_Dcm_00470\]](#), [\[SWS_Dcm_00473\]](#)) for DID's with *DcmDspDidUsePort* is set to *USE_ATOMIC_SENDER_RECEIVER_INTERFACE*, *USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE* or *USE_ATOMIC_NV_DATA_INTERFACE*, the Dcm module shall write the data by writing the associated sender-receiver or NvDataInterface DataServices_{DID}.]

[SWS_Dcm_CONSTR_06039] Signals with variable datalength [Only the last signal (*DcmDspDidSignal*) of a DID can have variable datalength (*DcmDspDataType* is set to *UINT8_DYN*).]

In other case the Dcm won't be able to split the data from the request.

[SWS_Dcm_00639] [To serialize the request message of UDS Service WriteDataByIdentifier request into the required AUTOSAR data types (signed- and unsigned integer), the target endianness configured in *DcmDspDataEndianness* shall be considered for *DcmDspData* elements having *DcmDspDataUsePort* set to *USE_DATA_SENDER_RECEIVER*, *USE_DATA_SENDER_RECEIVER_AS_SERVICE*. In case *DcmDspDataEndianness* is not present, the *DcmDspDataDefaultEndianness* shall be used instead.]

[SWS_Dcm_CONSTR_06018] [*DcmDspData* elements used in service 0x2E shall not have *DcmDspDataUsePorts* set to *USE_ECU_SIGNAL*.]

[SWS_Dcm_CONSTR_06073] Dependency for `DcmDspDataWriteFnc` [`DcmDspDataWriteFnc` shall be only present if:

- `DcmDspDataUsePort` is set to `USE_DATA_SYNCH_FNC` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_ERROR` or
- `DcmDspDataUsePort` is set to `USE_DATA_SYNCH_FNC_PROXY` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_PROXY`

]

Note: The invocation of functions `BndM_WriteStart` and `BndM_WriteFinalize` are not part of `Dcm` Specification. The functions are called via project specific implementation (e.g. CDD).

[SWS_Dcm_01582] Atomic BndM write operation

Upstream requirements: [RS_Diag_04243](#)

[After all verifications (see [\[SWS_Dcm_00467\]](#), [\[SWS_Dcm_00468\]](#), [\[SWS_Dcm_00469\]](#), [\[SWS_Dcm_00470\]](#), [\[SWS_Dcm_00473\]](#)) for `DIDs` with `DcmDspDidUsePort` is set to `USE_ATOMIC_BNDM`, the `Dcm` module shall write the data by writing the data to the associated `BlockId` from the `BndM` (`DcmDspDidBndMBlockIdRef`) using the block specific writing function `BndM_WriteBlock_<BlockId.Shortname>`.]

[SWS_Dcm_01583] Not allowed atomic BndM write operation

Upstream requirements: [RS_Diag_04243](#)

[If the `Dcm` calls `BndM_WriteBlock_‘BlockId.Shortname’` according to [\[SWS_Dcm_01582\]](#) and `BndM_WriteBlock_‘BlockId.Shortname’` returns `E_NOT_OK`, the `Dcm` shall return a negative response `0x24` (`requestSequenceError`).]

Note: The `BndM` needs to be set into writing mode as a precondition. This is either done by the coding sub-module in `Dcm` or by a CDD.

[SWS_Dcm_01584] Positive response on atomic BndM write operation

Upstream requirements: [RS_Diag_04243](#)

[If the `Dcm` has triggered an atomic `BndM` write operation according to [\[SWS_Dcm_01582\]](#), the `Dcm` shall return a positive response if the write operation of the `BndM` has called `Dcm_BndMWriteBlockFinish` with the parameter `result` set to `E_OK`.]

[SWS_Dcm_01585] Negative response on atomic BndM write operation

Upstream requirements: [RS_Diag_04243](#)

[If the [Dcm](#) has triggered an atomic BndM write operation according to [\[SWS_Dcm_01582\]](#), the [Dcm](#) shall return a negative response 0x72 (generalProgrammingFailure) if the write operation of the BndM has called [Dcm_BndMWriteBlockFinish](#) with the parameter result set to E_NOT_OK.]

Note: The [Dcm](#) needs to ensure that [\[SWS_Dcm_00024\]](#) requirement is respected while waiting on the job finish callback as specified in [\[SWS_Dcm_01584\]](#) and [\[SWS_Dcm_01585\]](#).

[SWS_Dcm_01586]

Upstream requirements: [RS_Diag_04243](#)

[If the [Dcm](#) is reading or writing a [DID](#) with [DcmDspDidUsePort](#) set to USE_ATOMIC_BNDM, the [Dcm](#) shall transform the dataRecord into the corresponding BndM type. The [DID](#) data element structure needs to be compatible to the referenced ImplementationDataType of the BndMBlockDescriptor. For each [DID](#) data element a corresponding sub-element shall exist in the ImplementationDataType with the same shortname and the basetypes shall be compatible.]

[SWS_Dcm_CONSTR_06099] Consistency of BndM Block configuration [If a [DcmDspDid](#) has [DcmDspDidUsePort](#) set to USE_ATOMIC_BNDM, the parameter [DcmDspDidBndMBlockIdRef](#) shall be present.]

7.4.2.14 Service 0x2F - InputOutputControlByIdentifier**[SWS_Dcm_00256]**

Upstream requirements: [RS_Diag_04218](#)

[The [Dcm](#) module shall implement the UDS Service InputOutputControlByIdentifier (0x2F).]

When using Service 0x2F, the request of the tester contains a 2-byte DID.

The configuration of the [Dcm](#) contains a list of supported DID's. For each DID, the [Dcm](#) configuration specifies:

- The 2-bytes [DID](#) (see configuration parameter [DcmDspDidIdentifier](#))
- For every data of the [DID](#) :
 - The function [Xxx_ReturnControlToECU\(\)](#) for this data (see configuration parameters [DcmDspDataReturnControlToEcuFnc](#) and [DcmDspDataUsePort](#))
 - The function [Xxx_ResetToDefault\(\)](#) for this data (see configuration parameters [DcmDspDataResetToDefaultFnc](#) and [DcmDspDataUsePort](#))

- The function `Xxx_FreezeCurrentState()` for this `DID` (see configuration parameters `DcmDspDataFreezeCurrentStateFnc` and `DcmDspDataUsePort`)
- The function `Xxx_ShortTermAdjustment()` for this `DID` (see configuration parameters `DcmDspDataShortTermAdjustmentFnc` and `DcmDspDataUsePort`)
- The sizes of the control record used in the function `Xxx_ShortTermAdjustment()` (see configuration parameter and `DcmDspDataByteSize`)

[SWS_Dcm_00579]

Upstream requirements: [RS_Diag_04218](#)

[The `Dcm` shall support `InputOutputControlParameter` definitions according to [\[SWS_Dcm_01661\]](#).]

[SWS_Dcm_01661] `InputOutputControlParameter` definitions [

Hex	Description
00	<code>returnControlToECU</code>
01	<code>resetToDefault</code>
02	<code>freezeCurrentState</code>
03	<code>shortTermAdjustment</code>

]

[SWS_Dcm_01554] `IOControl` `DID` authentication check

Upstream requirements: [RS_Diag_04218](#)

[On reception of the UDS Service `InputOutputControlByIdentifier` (0x2F), the `Dcm` shall check if the control access to the requested `DID` is authenticated and control the IO only if:

- for that IO control a role is configured via `DcmDspDidControlRoleRef` and the verification according to [\[SWS_Dcm_01522\]](#) was successful or
- the active white list on that connection has for the requested `DID` one entry with control access that matches that `DID`.

]

According to [\[SWS_Dcm_01522\]](#) the authentication checks are only executed if `DcmDspAuthentication` is configured. In case of a failed authentication the NRC handling is according to [\[SWS_Dcm_01544\]](#) and [\[SWS_Dcm_01551\]](#) applies.

[SWS_Dcm_00563]

Upstream requirements: [RS_Diag_04218](#)

[On reception of the [UDS](#) Service InputOutputControlByIdentifier (0x2F), the [Dcm](#) module shall check if the [DID](#) is supported (see configuration parameter [DcmDspDid](#)). If not, the [Dcm](#) module shall send [NRC](#) 0x31 (Request out of range).]

[SWS_Dcm_00564]

Upstream requirements: [RS_Diag_04218](#)

[If a [DID](#) is set as unused ([DcmDspDidUsed](#) set to FALSE), the [Dcm](#) shall consider the [DID](#) as not supported (according to [\[SWS_Dcm_00563\]](#))]

[SWS_Dcm_00565]

Upstream requirements: [RS_Diag_04218](#)

[On reception of the [UDS](#) Service InputOutputControlByIdentifier (0x2F), the [Dcm](#) module shall check if the [DID](#) has a Control access configured (see configuration parameter [DcmDspDidControl](#) in [DcmDspDidInfo](#)). If not, the [Dcm](#) module shall send [NRC](#) 0x31 (Request out of range).]

[SWS_Dcm_00566] [On reception of the [UDS](#) Service InputOutputControlByIdentifier (0x2F), the [Dcm](#) module shall check if the [DID](#) can be control in the current session (see configuration parameter [DcmDspDidControlSessionRef](#)). If not, the [Dcm](#) module shall send a [NRC](#) 0x31 (Request Out of Range).]

[SWS_Dcm_00567] [On reception of the [UDS](#) Service InputOutputControlByIdentifier (0x2F), the [Dcm](#) module shall check if the [DID](#) can be control in the current security level (see configuration parameter [DcmDspDidControlSecurityLevelRef](#)). If not, the [Dcm](#) module shall send [NRC](#) 0x33 (Security access denied).]

[SWS_Dcm_00823] [On reception of the [UDS](#) Service InputOutputControlByIdentifier (0x2F), the [Dcm](#) module shall check if the [DID](#) can be control in the current mode condition (see configuration parameter [DcmDspDidControlModeRuleRef](#)). If not, the [Dcm](#) module shall send the calculated negative response code of the referenced [DcmModeRule](#).]

[SWS_Dcm_01600] [CEMR](#) on SR interface for [DIDs](#) with no [CEMR](#) in the [UDS](#) request

Upstream requirements: [RS_Diag_04218](#)

[If the IOC [DID](#) has no [CEMR](#) in the [UDS](#) request message, the [Dcm](#) shall use a [Dcm_Cemr_Type](#) that has a bit for each [DcmDspData](#) of the [DID](#) on the interface [IOControlRequest](#) that has each bit set to 1.]

[SWS_Dcm_00580]

Upstream requirements: [RS_Diag_04218](#)

[On reception of a request for [UDS](#) Service InputOutputControlByIdentifier (0x2F), if all verifications have been successfully done (see [\[SWS_Dcm_00563\]](#),

[SWS_Dcm_00565], [SWS_Dcm_00566], [SWS_Dcm_00567]) and if the data is configured as a "ECU signal" of the IoHwAb (parameter `DcmDspDataUsePort`), the `Dcm` shall call the Api `IoHwAb_Dcm_<symbolic name of ECU signal (parameter DcmDspDataEcuSignal)>()` with `InputOutputControlParameter` for the 'action' parameter and in case of `InputOutputControlParameter` is set to 'shortTermAdjustment' the signal value for the "signal" parameter. In this case the requirements [SWS_Dcm_00396], [SWS_Dcm_00397], [SWS_Dcm_00398] and [SWS_Dcm_00399] doesn't apply.]

[SWS_Dcm_00581]

Upstream requirements: [RS_Diag_04218](#)

[In case of more than one supported I/O signal per `DataIdentifier` and the configuration parameter `DcmDspDidControlMask` is set to `DCM_CONTROLMASK_INTERNAL`, the `Dcm` shall internally consider the parameter `controlEnableMaskRecord` and control only the included signals in the request message.]

[SWS_Dcm_CONSTR_06051] [The configuration parameter `DcmDspDidControlMaskSize` shall be only present if `DcmDspDidControlMask` is equal to `DCM_CONTROLMASK_EXTERNAL` or `DCM_CONTROLMASK_INTERNAL`.]

[SWS_Dcm_01273]

Upstream requirements: [RS_Diag_04218](#)

[If the configuration parameter `DcmDspDidControlMask` is set to `DCM_CONTROLMASK_EXTERNAL` or `DCM_CONTROLMASK_INTERNAL`, or the `DcmDspDid` element used in service 0x2F is configured to have an atomic S/R interface, the `Dcm` shall reject requests without included control enable mask record with the `NRC` 0x13 (`incorrectMessageLengthOrInvalidFormat`).]

[SWS_Dcm_01274]

Upstream requirements: [RS_Diag_04218](#)

[If the configuration parameter `DcmDspDidControlMask` is set to `DCM_CONTROLMASK_NO`, the `Dcm` shall reject request with included control enable mask record with the `NRC` 0x13 (`incorrectMessageLengthOrInvalidFormat`).]

[SWS_Dcm_CONSTR_06084] Sender-receiver communication for IOControls is limited to atomic S/R interfaces

Upstream requirements: [RS_Diag_04218](#)

[If a `DID` has a configured `DcmDspDidUsePort` = `USE_DATA_ELEMENT_SPECIFIC_INTERFACES`, the possible values of `DcmDspDataUsePort` are limited to non S/R interfaces.]

[SWS_Dcm_CONSTR_06085] Atomic S/R for IOControls are limited to non-NV interfaces

Upstream requirements: [RS_Diag_04218](#)

[If a [DID](#) has a configured [DcmDspDidControl](#), the possible values of [DcmDspDidUsePort](#) are limited to atomic S/R interface and [USE_DATA_ELEMENT_SPECIFIC_INTERFACES](#).]

[SWS_Dcm_CONSTR_06086] Signals for [DIDs](#) with Atomic S/R are not shared with other [DIDs](#)

Upstream requirements: [RS_Diag_04218](#)

[If a [DcmDspDid](#) is configured to have an atomic S/R interface, all [DcmDspDataElements](#) referenced by this [DID](#) shall be referenced only from this [DID](#).

]

[SWS_Dcm_CONSTR_06050] [If a [DcmDspDid](#) is used in service 0x2F and is configured to have an atomic S/R interface, the [DcmDspDidControlMask](#) shall be set to [DCM_CONTROLMASK_EXTERNAL](#) and the parameter [DcmDspDidControlMaskSize](#) shall be present with a value greater than zero.]

[SWS_Dcm_00680] Mapping of internal ControlEnableMaskRecord to [DID](#) data elements

Upstream requirements: [RS_Diag_04218](#)

[If [DcmDspDidControlMask](#) is set to [DCM_CONTROLMASK_INTERNAL](#), the [ControlEnableMaskRecord](#) shall be mapped to the [DID](#) data elements by applying the following mapping :

- The most significant bit of the first byte of the [ControlEnableMask](#) shall correspond to the first [DID](#) data element
- The second most significant bit of the first byte of the [ControlEnableMask](#) shall correspond to the second [DID](#) data element and continuing on in this fashion utilizing as many [ControlEnableMask](#) bytes as necessary to map all [DID](#) data elements.

]

The [ControlEnableMaskRecord](#) is only present, if the [DataIdentifier](#) supports more than one signal.

The [Dcm](#) supports atomic S/R interfaces activated by the configuration [DcmDspDidUsePort](#) set to [USE_ATOMIC_SENDER_RECEIVER_INTERFACE](#) or [USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE](#). In the text and requirements of this chapter the term 'atomic S/R interface' for IO control means that the IO controlled [DID](#) is configured to one of the two choices.

The service use case TPS_SWCT_01654 and the constr_1679 limits the S/R interfaces used for IOControl to explicit S/R communication. In implicit communication is not supported by the [Dcm](#).

[SWS_Dcm_01434] IOControl General execution sequence

Upstream requirements: [RS_Diag_04218](#)

[On reception of a request for [UDS Service InputOutputControlByIdentifier \(0x2F\)](#) the [Dcm](#) shall first execute the service verifications according to [\[SWS_Dcm_00563\]](#), [\[SWS_Dcm_00565\]](#), [\[SWS_Dcm_00566\]](#), [\[SWS_Dcm_00567\]](#) and on successful passing the verifications start the configured service processing.]

[SWS_Dcm_00396]

Upstream requirements: [RS_Diag_04218](#)

[On reception of a request for [UDS Service InputOutputControlByIdentifier \(0x2F\)](#) with [inputOutputControlParameter](#) equal to [returnControlToEcu](#), the [Dcm](#) module shall invoke all impacted configured function of the [controlEnableMaskRecord](#) (if parameter [DcmDspDataUsePort](#) set to [USE_DATA_SYNCH_FNC](#) or [USE_DATA_ASYNCH_FNC](#) or [USE_DATA_ASYNCH_FNC_ERROR](#) or [USE_DATA_SYNCH_FNC_PROXY](#) or [USE_DATA_ASYNCH_FNC_PROXY](#); see configuration parameter [DcmDspDataReturnControlToEcuFnc](#)). Alternatively call all the associated [ReturnControlToECU](#) operations (if parameter [DcmDspDataUsePort](#) set to [USE_DATA_SYNCH_CLIENT_SERVER](#) or [USE_DATA_ASYNCH_CLIENT_SERVER](#) or [USE_DATA_ASYNCH_CLIENT_SERVER_ERROR](#)) for every data of the [DID](#) received in the request.]

[SWS_Dcm_00397]

Upstream requirements: [RS_Diag_04218](#)

[On reception of a request for [UDS Service InputOutputControlByIdentifier \(0x2F\)](#) with [inputOutputControlParameter](#) equal to [resetToDefault](#), the [Dcm](#) module shall invoke all impacted configured function of the [controlEnableMaskRecord](#) (if parameter [DcmDspDataUsePort](#) set to [USE_DATA_SYNCH_FNC](#) or [USE_DATA_ASYNCH_FNC](#) or [USE_DATA_ASYNCH_FNC_ERROR](#) or [USE_DATA_SYNCH_FNC_PROXY](#) or [USE_DATA_ASYNCH_FNC_PROXY](#); see configuration parameter [DcmDspDataResetToDefaultFnc](#)). Alternatively call all the associated [ResetToDefault](#) operations (if parameter [DcmDspDataUsePort](#) set to [USE_DATA_SYNCH_CLIENT_SERVER](#) or [USE_DATA_ASYNCH_CLIENT_SERVER](#) or [USE_DATA_ASYNCH_CLIENT_SERVER_ERROR](#)) for every data of the [DID](#) received in the request.]

[SWS_Dcm_00398]

Upstream requirements: [RS_Diag_04218](#)

[On reception of a request for [UDS Service InputOutputControlByIdentifier \(0x2F\)](#) with [inputOutputControlParameter](#) equal to [freezeCurrentState](#), the [Dcm](#) module shall invoke all impacted configured function of the [controlEnableMaskRecord](#)

(if parameter `DcmDspDataUsePort` set to `USE_DATA_SYNCH_FNC` or `USE_DATA_ASYNCH_FNC` or `USE_DATA_ASYNCH_FNC_ERROR` or `USE_DATA_SYNCH_FNC_PROXY` or `USE_DATA_ASYNCH_FNC_PROXY`; see configuration parameter `DcmDspDataFreezeCurrentStateFnc`). Alternatively call all the associated Freeze-CurrentState operations (if parameter `DcmDspDataUsePort` set to `USE_DATA_SYNCH_CLIENT_SERVER` or `USE_DATA_ASYNCH_CLIENT_SERVER` or `USE_DATA_ASYNCH_CLIENT_SERVER_ERROR`) for every data of the `DID` received in the request.]

[SWS_Dcm_00399]

Upstream requirements: [RS_Diag_04218](#)

[On reception of a request for UDS Service InputOutputControlByIdentifier (0x2F) with `inputOutputControlParameter` equal to `shortTermAdjustment`, the `Dcm` module shall invoke all impacted configured function of the `controlEnableMaskRecord` (if parameter `DcmDspDataUsePort` set to `USE_DATA_SYNCH_FNC` or `USE_DATA_ASYNCH_FNC` or `USE_DATA_ASYNCH_FNC_ERROR` or `USE_DATA_SYNCH_FNC_PROXY` or `USE_DATA_ASYNCH_FNC_PROXY`; see configuration parameter `DcmDspDataShortTermAdjustmentFnc`). Alternatively call all the associated ShortTermAdjustment operations (if parameter `DcmDspDataUsePort` set to `USE_DATA_SYNCH_CLIENT_SERVER` or `USE_DATA_ASYNCH_CLIENT_SERVER` or `USE_DATA_ASYNCH_CLIENT_SERVER_ERROR`) for every data of the `DID` received in the request.]

[SWS_Dcm_00858] Cancel active IO controls on session change

Upstream requirements: [RS_Diag_04119](#)

[On any session transition, the `Dcm` shall stop all active IO controls according to [\[SWS_Dcm_01435\]](#) which are not supported by the new session.]

[SWS_Dcm_00628] Cancel active IO controls in default sessions

Upstream requirements: [RS_Diag_04119](#)

[On a session transition to default session, the `Dcm` shall stop all active IO controls according to [\[SWS_Dcm_01435\]](#).]

[SWS_Dcm_00859] Cancel active IO controls on security level change

Upstream requirements: [RS_Diag_04119](#)

[On any security level change, the `Dcm` shall stop all active IO controls according to [\[SWS_Dcm_01435\]](#) which are not support by the new security level anymore.]

[SWS_Dcm_01435] Dcm cancel IO control sequence

Upstream requirements: [RS_Diag_04119](#)

[If the `Dcm` needs to cancel an active IO control due to [\[SWS_Dcm_00858\]](#), [\[SWS_Dcm_00628\]](#) or [\[SWS_Dcm_00859\]](#), the `Dcm` shall do the following:

- For controlled data elements with `DcmDspDataUsePort` set to `USE_ECU_SIGNAL`: call to `IoHwAb_Dcm_<symbolic ECU signal name>()` with 'action' parameter set to `ReturnControlToECU`.

- For controlled data elements with `DcmDspDataUsePort` set to `USE_DATA_ASYNC_CLIENT_SERVER` or `USE_DATA_SYNC_CLIENT_SERVER` or `USE_DATA_ASYNC_CLIENT_SERVER_ERROR`: call the C/S interface operation `ReturnControlToECU`.
- For controlled data elements with `DcmDspDataUsePort` set to `USE_DATA_SYNC_FNC` or `USE_DATA_ASYNC_FNC` or `USE_DATA_ASYNC_FNC_ERROR` or `USE_DATA_SYNC_FNC_PROXY` or `USE_DATA_ASYNC_FNC_PROXY`: call the configured function `Xxx_ReturnControlToECU` (see parameter `DcmDspDataReturnControlToEcuFnc`)
- For controlled `DIDs` with is configured atomic S/R interfaces: update the data element `IOOperationRequest` with `inputOutputControlParameter = 0x00`, the `controlEnableMask = 0xFFFFFFFF1` and data element `underControl = 0x00`.

]

[SWS_Dcm_00640]*Upstream requirements:* [RS_Diag_04218](#)

[To serialize the required AUTOSAR data types (signed- and unsigned integer) from the request message (in case of `inputOutputControlParameter` is set to 'short-TermAdjustment') / into the response message of UDS Service InputOutputControlByIdentifier responses, the target endianness configured in `DcmDspDataEndianness` shall be considered for `DcmDspData` elements having `DcmDspDataUsePort` set to `USE_ECU_SIGNAL`. In case `DcmDspDataEndianness` is not present, the `DcmDspDataDefaultEndianness` shall be used instead.]

[SWS_Dcm_00682]*Upstream requirements:* [RS_Diag_04218](#)

[The `controlState` in the `ControlStatusRecord` for positive response message of `IoControl` service shall be retrieved using the associated `ReadData` operation/function/SenderReceiver after application processing on the IO control request is positively finalized.]

Beside the Client/Server interface, the `Dcm` provides the SenderReceiver interface `IOControlRequest_DID`. The `underControl` data element of this interface is calculated by the `Dcm` with one state bit for each data element identical to the `CEMR`. Applications can directly derive the active control enable status without the need to maintain internal states.

The bit-mask `underControl` contains the accumulated status about which data elements of this particular I/O is currently under diagnostic control. The normal operation state could be derived if the value of `underControl` is set to `0x00` (which is the initial value). Each set bit indicates a data element which is under diagnostic control via `FreezeCurrentState`, 'ResetToDefault' or 'ShortTermAdjustment'.

¹The size of the mask depends on the parameter `DcmDspDidControlMaskSize`

[SWS_Dcm_01436] Calculation of the underControl data element

Upstream requirements: [RS_Diag_04218](#)

[The [Dcm](#) shall calculate the underControl data element of the S/R interface [IOControlRequest_{DID}](#). The underControl is a bitfield of the same size than the [CEMR](#) of the controlled [DID](#). Each bit represents the same data element as in the [CEMR](#). A value of 0 indicates, that the corresponding data element is currently not controlled by the [Dcm](#), a value of 1 indicates that it is controlled. The initial value is 0, each control request for a data element with [inputOutputControlParameter](#) equal to ResetToDefault, FreezeCurrentState or ShortTermAdjustment will set the corresponding bit value to 1, and each control request with [inputOutputControlParameter](#) set to ReturnControlToECU will set the bit value to 0.]

With each I/O Control request a command [IOOperationRequest](#) is provided to the application to update the input or the respective output. [IOOperationRequest](#) contains the [inputOutputControlParameter](#), the [controlEnableMask](#) and in case of ShortTermAdjustment the [controlState](#).

To identify that previous operation has finished (e.g Write [IOControlRequest_{DID}](#)), the user can use the update flag mechanism from the [RTE](#).

The application needs to update their output values and finalizes the request with the response message [IOOperationRequest](#) to the [Dcm](#). The possible values are:

- 0x00 positive response (similar to E_OK)
- 0x10 generalReject
- 0x21 busyRepeatRequest
- 0x22 conditionsNotCorrect
- 0x26 FailurePreventsExecutionOfRequestedAction
- 0x31 requestOutOfRange
- 0x78 ResponsePending (similar to E_PENDING)

Based on the write trigger of the [SW-C](#) to [IOControlResponse_{DID}.IOOperationResponse](#), the [Dcm](#) will:

1. wait for final processing (0x78)
2. send a positive response message (0x00)
3. send a negative response message (all other values, except 0xFF)

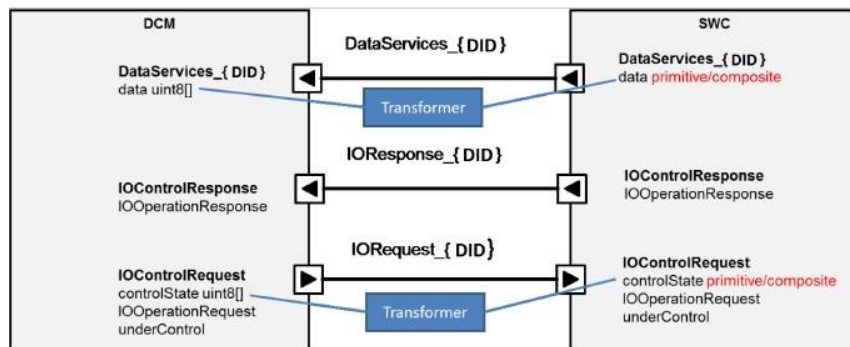


Figure 7.12: IO-Control with Sender/Receiver interfaces

[SWS_Dcm_01437] `inputOutputControlParameter` idle state

Upstream requirements: RS_Diag_04218

[The `inputOutputControlParameter` of data element of `IOOperationRequest` from S/R interface `IOControlRequest_{DID}` shall have an initial value of 0xFF. This value indicates the application, that the `Dcm` is currently not processing an UDS service to control this `DID`.]

[SWS_Dcm_01438] `inputOutputControlParameter` after processing an UDS `InputOutputControlByIdentifier (0x2F)` service

Upstream requirements: RS_Diag_04218

[If the `Dcm` is processing an `InputOutputControlByIdentifier` request with `inputOutputControlParameter` equal to `ResetToDefault`, `FreezeCurrentState` or `ShortTermAdjustment`, the `Dcm` shall set the `inputOutputControlParameter` of data element of `IOOperationRequest` from S/R interface `IOControlRequest_{DID}` to the idle state 0xFF after the application has set the `IOControlResponse_{DID}`. `IOOperationResponse` to 0x00 and before processing other `InputOutputControlByIdentifier` requests.]

Upon the `Dcm` writes `IOOperationRequest` of `IOControlRequest_{DID}` the SWC processes the IO control request. The SWC informs the `Dcm` about the current processing state by updating `IOControlResponse_{DID}`.`IOOperationResponse`.

[SWS_Dcm_01439] Positive response based on `IOOperationResponse`

Upstream requirements: RS_Diag_04218

[If the `Dcm` is processing an `InputOutputControlByIdentifier` request, it shall reply with a positive response, if the applications set `IOControlResponse_{DID}`.`IOOperationResponse` to 0x00.]

[SWS_Dcm_01440] Negative response based on IOOperationResponse

Upstream requirements: RS_Diag_04218

[If the Dcm is processing an InputOutputControlByIdentifier request, it shall reply with a negative response with the NRC IOControlResponse_{DID}.IOOperationResponse, if the applications set IOControlResponse_{DID}.IOOperationResponse a value different to 0x00 and 0x78.]

[SWS_Dcm_01441] RCRRP based on IOOperationResponse

Upstream requirements: RS_Diag_04218

[If the Dcm is processing an InputOutputControlByIdentifier request and the IOControlResponse_{DID}.IOOperationResponse has a value of 0x78, the Dcm shall wait until the IOControlResponse_{DID}.IOOperationResponse gets a value different to 0x78 and send RCRRP according to [SWS_Dcm_00024].]

Note: The use of the RTE functionality "IsUpdated" is a possible mechanism for the Dcm to detect a write from SW-C to the S/R data element.

[SWS_Dcm_01275] Common action for all inputOutputControlParameter operations with atomic S/R

Upstream requirements: RS_Diag_04218

[If the Dcm is processing an InputOutputControlByIdentifier request for a DID configured atomic S/R interface, the Dcm module shall update in the IOControlRequest_{DID} the data element IOOperationRequest with

1. controlEnableMask = controlEnableMaskRecord of the request message
2. inputOutputControlParameter = inputOutputControlParameter from the request message

]

The value 0xFF of the inputOutputControlParameter of the command IOOperationRequest is the 'idle' state. The values 0x00 (ReturnControlToECU), 0x01 (ResetToDefault), 0x02 (FreezeCurrentState) or 0x03 (ShortTermAdjustment) start the request processing and include the control option inputOutputControlParameter, controlEnableMask and controlState (for ShortTermAdjustment only).

[SWS_Dcm_01277] Additional action for InputOutputControl operations for ShortTermAdjustment with atomic S/R

Upstream requirements: RS_Diag_04218

[If the Dcm is processing an InputOutputControlByIdentifier request with inputOutputControlParameter equal to ShortTermAdjustment for a DID with configured atomic S/R interface, in addition to [SWS_Dcm_01275] the Dcm module shall update in the IOControlRequest_{DID} the data element controlState with content of the controlState from the request message.]

Note: The `controlState` is a separate data element that it can be optionally processed by a data transformer to transform the byte stream into a composite type (see Figure 7.12: IO-Control with Sender/Receiver interfaces).

An example of the `Dcm` S/R interaction is given in Figure 9.22, Figure 9.23 and Figure 9.24. For `ReturnControlToECU` the data from the request is provided to the application, the `Dcm` will continue to finalize the request after writing the data into the S/R ports. All other sub-functions will wait for the application providing the response `0x00` or an `NRC` code.

[SWS_Dcm_CONSTR_06048] Composite sub elements accessible only by read
[Composite sub elements can only be referred from Read `DID` i.e. Write and Control `DID` are not supported.]

[SWS_Dcm_CONSTR_06030] [The `ReturnControlToEcu` functionality is existing if at least one of the following parameters are activated : `DcmDspDidFreezeCurrentState` in `ECUC_Dcm_00624` : or `DcmDspDidResetToDefault` in `ECUC_Dcm_00623` : or `DcmDspDidShortTermAdjustment` in `ECUC_Dcm_00625` : .]

[SWS_Dcm_CONSTR_06059] Dependency for `DcmDspDataFreezeCurrentStateFnc` [`DcmDspDataFreezeCurrentStateFnc` shall be only present if:

- `DcmDspDataUsePort` is set to `USE_DATA_SYNC_FNC` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_ERROR` or
- `DcmDspDataUsePort` is set to `USE_DATA_SYNC_FNC_PROXY` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_PROXY`

]

[SWS_Dcm_CONSTR_06063] Dependency for `DcmDspDataResetToDefaultFnc`
[`DcmDspDataResetToDefaultFnc` shall be only present if:

- `DcmDspDataUsePort` is set to `USE_DATA_SYNC_FNC` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_ERROR` or
- `DcmDspDataUsePort` is set to `USE_DATA_SYNC_FNC_PROXY` or
- `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_PROXY`

]

[SWS_Dcm_CONSTR_06064] Dependency for DcmDspDidControlMaskSize [DcmDspDidControlMaskSize shall be only present if DcmDspDidControlMask is equal to DCM_CONTROLMASK_EXTERNAL or DCM_CONTROLMASK_INTERNAL.]

[SWS_Dcm_CONSTR_06081] Dependency for DcmDspDidControlMaskBitPosition [The value configured for DcmDspDidControlMaskBitPosition shall be lower than DcmDspDidControlMaskSize * 8.]

[SWS_Dcm_CONSTR_06065] Dependency for DcmDspDataReturnControlToEcuFnc [DcmDspDataReturnControlToEcuFnc shall be only present if:

- DcmDspDataUsePort is set to USE_DATA_SYNC_FNC or
- DcmDspDataUsePort is set to USE_DATA_ASYNC_FNC or
- DcmDspDataUsePort is set to USE_DATA_ASYNC_FNC_ERROR or
- DcmDspDataUsePort is set to USE_DATA_SYNC_FNC_PROXY or
- DcmDspDataUsePort is set to USE_DATA_ASYNC_FNC_PROXY

]

[SWS_Dcm_CONSTR_06066] Dependency for DcmDspDataShortTermAdjustmentFnc [DcmDspDataShortTermAdjustmentFnc shall be only present if:

- DcmDspDataUsePort is set to USE_DATA_SYNC_FNC or
- DcmDspDataUsePort is set to USE_DATA_ASYNC_FNC or
- DcmDspDataUsePort is set to USE_DATA_ASYNC_FNC_ERROR or
- DcmDspDataUsePort is set to USE_DATA_SYNC_FNC_PROXY or
- DcmDspDataUsePort is set to USE_DATA_ASYNC_FNC_PROXY

]

[SWS_Dcm_CONSTR_06082] Dependency for DcmDspDidControlMaskSize [DcmDspDidControlMaskSize larger than 4 shall be only allowed if DcmDspDataUsePort is set to USE_DATA_ASYNC_CLIENT_SERVER, USE_DATA_ASYNC_CLIENT_SERVER_ERROR or USE_DATA_SYNC_CLIENT_SERVER. Note: ControlEnableMask larger than 32 bits is a very rare use case. Therefore the Dcm supports only C/S interfaces to solve this use case.]

7.4.2.15 Service 0x31 - RoutineControl

[SWS_Dcm_00257] [The Dcm module shall implement the UDS Service RoutineControl (0x31) for subFunctions startRoutine, stopRoutine and requestsRoutineResults.]

A tester can use UDS Service 0x31 to start, stop or obtain the results of a routine identified by a 2-byte routineIdentifier. The Dcm module configuration contains a list of the routineIdentifiers (see configuration parameter `DcmDspRoutineIdentifier`) supported by the DCM. For each routineIdentifier, the Dcm configuration specifies:

- The function `Xxx_Start()` associated with this routineIdentifier (see configuration parameters `DcmDspStartRoutineFnc`, `DcmDspRoutineUsePort` and `DcmDspRoutineFncSignature`)
- List of signal available in the request and in the response (see configuration parameters `DcmDspStartRoutineIn` and `DcmDspStartRoutineOut`)
- The function `Xxx_Stop()` associated with this routineIdentifier (see configuration parameters `DcmDspStopRoutineFnc`, `DcmDspRoutineUsePort` and `DcmDspRoutineFncSignature`)
- List of signal available in the request and in the response (see configuration parameters `DcmDspStopRoutineIn` and `DcmDspStopRoutineOut`)
- The function `Xxx_RequestResults()` associated with this routineIdentifier (see configuration parameters `DcmDspRequestRoutineResultsFnc`, `DcmDspRoutineUsePort` and `DcmDspRoutineFncSignature`)
- List of signal available in the request and in the response (see configuration parameters `DcmDspRequestRoutineResultsIn` and `DcmDspRequestRoutineResultsOut`)

[SWS_Dcm_01442]

Upstream requirements: [RS_Diag_04224](#)

[If `DcmDspRoutineUsePort` is set to true, the Dcm shall call the corresponding operation of the C/S interfaces `RoutineServices_RoutineName` to process this routine.]

A routine handler processes the diagnostic routine control request. The Dcm passes the `routineControlOption` as input parameters to the routine handler. The routine processes the request and writes the result to the output parameters of the routine handler call. In case of shared Rx and Tx buffer, there are two cases where writing to the output parameters of the routine control can cause overwriting of the input parameters:

- a routine control uses arrays as output parameters (call by reference)
- a routine control writes to output parameters and returns `DCM_E_PENDING`. It is called again with `DcmOpstatus` set to `DCM_PENDING`. The input parameters are derived from the Rx buffer that was overwritten by the previous write to the output parameters

For efficient memory consumption it is controllable, if the Dcm applies further strategies to separate input and output parameters in those situations.

[SWS_Dcm_01580] Input and output consistency during C/S based routine handling

Upstream requirements: [RS_Diag_04224](#)

[If [DcmDspRoutineUsePort](#) is set to TRUE and the `ClientServerOperation.diagArgIntegrity` of that operation is also set to TRUE, the [Dcm](#) shall ensure that the routine handler in the C/S interfaces `RoutineServices_RoutineName` writing to the output parameters will not overwrite the input parameters.]

[SWS_Dcm_01581] "Routine argument integrity for C/S calls

Upstream requirements: [RS_Diag_04224](#)

[If [DcmDspRoutineUsePort](#) is set to true and [DcmDspRoutineInterfaceArgumentIntegrity](#) is configured, the [Dcm](#) shall create the corresponding operation of the C/S interface `RoutineServices_RoutineName` with `ClientServerOperation.diagArgIntegrity` set to [DcmDspRoutineInterfaceArgumentIntegrity](#).]

[SWS_Dcm_CONSTR_06097] Availability of [DcmDspRoutineInterfaceArgumentIntegrity](#) [[DcmDspRoutineInterfaceArgumentIntegrity](#) shall only be available if the corresponding [DcmDspRoutine](#) has [DcmDspRoutineUsePort](#) is set to true.]

[SWS_Dcm_01443]

Upstream requirements: [RS_Diag_04224](#)

[If [DcmDspRoutineUsePort](#) is set to false, the [Dcm](#) shall use the configured callout functions for routine operations.]

[SWS_Dcm_00568] [On reception of the UDS Service RoutineControl (0x31), the [Dcm](#) module shall check if the Routine is supported (see configuration parameter [DcmDspRoutine](#)) If not, the [Dcm](#) module shall send `NRC 0x31` (Request out of range).]

[SWS_Dcm_00569] [If a Routine is set as unused ([DcmDspRoutineUsed](#) set to FALSE), the [Dcm](#) shall consider the Routine as not supported (according to [\[SWS_Dcm_00568\]](#))]

[SWS_Dcm_01555] StartRoutine authentication check

Upstream requirements: [RS_Diag_04233](#)

[On reception of the UDS Service RoutineControl (0x31) with sub-function `startRoutine`, the [Dcm](#) shall check if the access to the requested routine identifier is authenticated and process the routine only if:

- for this start routine service a role is configured via [DcmDspStartRoutineRoleRef](#) and the verification according to [\[SWS_Dcm_01522\]](#) was successful or
- the active white list on that connection has one RID entry with sub-function access set to `startRoutine` that matches that service and sub-function.

]

[SWS_Dcm_01556] StopRoutine authentication check

Upstream requirements: [RS_Diag_04233](#)

[On reception of the UDS Service RoutineControl (0x31) with sub-function stopRoutine, the Dcm shall check if the access to the requested routine identifier is authenticated and process the routine only if:

- for this stop routine service a role is configured via [DcmDspStopRoutineRoleRef](#) and the verification according to [\[SWS_Dcm_01522\]](#) was successful or
- the active white list on that connection has one RID entry with sub-function access set to stopRoutine that matches that service and sub-function.

]

[SWS_Dcm_01557] RequestRoutineResult authentication check

Upstream requirements: [RS_Diag_04233](#)

[On reception of the UDS Service RoutineControl (0x31) with sub-function requestRoutineResult, the Dcm shall check if the access to the requested routine identifier is authenticated and process the routine only if:

- for this request routine results service a role is configured via [DcmDspRequestRoutineResultsRoleRef](#) and the verification according to [\[SWS_Dcm_01522\]](#) was successful or
- the active white list on that connection has one RID entry with sub-function access set to requestRoutineResults that matches that service and sub-function.

]

According to [\[SWS_Dcm_01537\]](#) the authentication checks are only executed if [DcmDspAuthentication](#) is configured. In case of a failed authentication the NRC handling is according to [\[SWS_Dcm_01544\]](#) and [\[SWS_Dcm_01551\]](#) applies.

The service RoutineControl (0x31) is of exotic nature and has both, a sub-function and an identifier. UDS defines a different behavior for this service. Permission checks for session and security are done on identifier level rather than on sub-function level.

[SWS_Dcm_CONSTR_06100] Same session and security settings for same routine identifier [All [DcmDspCommonAuthorization](#) that are referenced via [DcmDspStartRoutineCommonAuthorizationRef](#), [DcmDspStopRoutineCommonAuthorizationRef](#) or [DcmDspRequestRoutineResultsCommonAuthorizationRef](#) from a the same [DcmDspRoutine](#), shall have the same (identical) set of referenced [DcmDspCommonAuthorizationSecurityLevelRef](#) and [DcmDspCommonAuthorizationSessionRef](#).]

[SWS_Dcm_00570] [On reception of the UDS Service RoutineControl (0x31), the Dcm module shall check if the Routine can be executed in the current session (see configuration parameters [DcmDspStartRoutineCommonAuthorizationRef](#), [DcmDspStopRoutineCommonAuthorizationRef](#) and [DcmDspRequestRoutineResultsCommonAuthorizationRef](#)). If not, the Dcm module shall send a NRC 0x31 (Request Out of Range).]

[SWS_Dcm_00571] [On reception of the UDS Service RoutineControl (0x31), the Dcm module shall check if the Routine can be executed in the current security level (see configuration parameter [DcmDspStartRoutineCommonAuthorizationRef](#), [DcmDspStopRoutineCommonAuthorizationRef](#) and [DcmDspRequestRoutineResultsCommonAuthorizationRef](#)). If not, the Dcm module shall send NRC 0x33 (Security access denied).]

[SWS_Dcm_00869] [On reception of the UDS Service RoutineControl (0x31), the Dcm module shall check if the SubFunction to the corresponding Routine is supported (see existence of configuration container [DcmDspStopRoutine](#) for SubFunction 0x02; [DcmDspRequestRoutineResults](#) for SubFunction 0x03). If not, the Dcm module shall send NRC 0x12 (SubFunction not supported).]

[SWS_Dcm_01169] [On reception of the UDS Service RoutineControl (0x31) with SubFunction startRoutine, the Dcm module shall check if the Routine can be executed in the current mode condition (see configuration parameter [DcmDspStartRoutineCommonAuthorizationRef](#)). If not, the Dcm module shall send the calculated negative response code of the referenced [DcmModeRule](#).]

[SWS_Dcm_01170] [On reception of the UDS Service RoutineControl (0x31) with SubFunction stopRoutine, the Dcm module shall check if the Routine can be executed in the current mode condition (see configuration parameter [DcmDspStopRoutineCommonAuthorizationRef](#)). If not, the Dcm module shall send the calculated negative response code of the referenced [DcmModeRule](#).]

[SWS_Dcm_01171] [On reception of the UDS Service RoutineControl (0x31) with SubFunction requestRoutineResults, the Dcm module shall check if the Routine can be executed in the current mode condition (see configuration parameter [DcmDspRequestRoutineResultsCommonAuthorizationRef](#)). If not, the Dcm module shall send the calculated negative response code of the referenced [DcmModeRule](#).]

Routines have different input and output parameters depending on the routine configuration (e.g. [DcmDspStartRoutineIn](#) for input parameter for the routine start service). The signature of the called routine operations Xxx_Start, Xxx_Stop and Xxx_RequestResults is depending on this configuration. The defined parameters for input and output routine data are optional, and marked in brackets '[]' in the definition in [\[SWS_Dcm_01203\]](#), [\[SWS_Dcm_01204\]](#) and [\[SWS_Dcm_91013\]](#).

[SWS_Dcm_01360] [For each configured routine input signal in [DcmDspStartRoutineInSignal](#), [DcmDspStopRoutineInSignal](#) or [DcmDspRequestRoutineResultsInSignal](#) with a signal type unequal to `VARIABLE_LENGTH`, the optional parameter 'DcmDspRoutineSignalType dataIn_n' shall be provided in the corresponding operations in [\[SWS_Dcm_01203\]](#), [\[SWS_Dcm_01204\]](#) or [\[SWS_Dcm_91013\]](#).]

[SWS_Dcm_01361] [For a configured routine input signal in [DcmDspStartRoutineInSignal](#), [DcmDspStopRoutineInSignal](#) or [DcmDspRequestRoutineResultsInSignal](#) with a signal type equal to `VARIABLE_LENGTH` the optional parameter `const 'uint8 * dataInVar'` shall be provided in the corresponding operations in [\[SWS_Dcm_01203\]](#), [\[SWS_Dcm_01204\]](#) or [\[SWS_Dcm_91013\]](#).]

[SWS_Dcm_01362] [For each configured routine output signal in [DcmDspStartRoutineOutSignal](#), [DcmDspStopRoutineOutSignal](#) or [DcmDspRequestRoutineResultsOutSignal](#) with a signal type unequal to `VARIABLE_LENGTH` the optional parameter 'DcmDspRoutineSignalType dataOut_n' shall be provided in the corresponding operations in [\[SWS_Dcm_01203\]](#), [\[SWS_Dcm_01204\]](#) or [\[SWS_Dcm_91013\]](#).]

[SWS_Dcm_01363] [For a configured routine output signal in [DcmDspStartRoutineOutSignal](#), [DcmDspStopRoutineOutSignal](#) or [DcmDspRequestRoutineResultsOutSignal](#) with a signal type equal to `VARIABLE_LENGTH` the optional parameter `const 'uint8 * dataOutVar'` shall be provided in the corresponding operations in [\[SWS_Dcm_01203\]](#), [\[SWS_Dcm_01204\]](#) or [\[SWS_Dcm_91013\]](#).]

[SWS_Dcm_01364] [The optional in/out parameter `currentDataLength` in [\[SWS_Dcm_01203\]](#), [\[SWS_Dcm_01204\]](#) or [\[SWS_Dcm_91013\]](#) is always present if at least one of the routine input signal data or routine output signal data have a signal with routine type '`VARIABLE_LENGTH`'.]

Note: The '`currentDataLength`' parameter as in/out parameter contains the data length in bytes of the '`dataInVar`' while calling the operation and it returns the length in bytes of the '`dataOutVar`'. As '`dataInVar`' and '`dataOutVar`' are optional, '`currentDataLength`' is only present if at least one of this optional parameter is used.

[SWS_Dcm_00590] [When receiving a request for [UDS Service RoutineControl](#) (0x31) if all verifications have been successfully done (see [\[SWS_Dcm_00568\]](#), [\[SWS_Dcm_00570\]](#), [\[SWS_Dcm_00571\]](#)), the [Dcm](#) module shall split the `routineControlOptionRecord` received according of the list of input signal configured for this routine (see configuration parameters [DcmDspStartRoutineIn](#), [DcmDspStopRoutineIn](#), [DcmDspRequestRoutineResultsIn](#))]

[SWS_Dcm_01614] DRAFT [When receiving a request for [UDS Service RoutineControl](#) (0x31) if all verifications have been successfully done (see [\[SWS_Dcm_00568\]](#), [\[SWS_Dcm_00570\]](#), [\[SWS_Dcm_00571\]](#)) and [DcmDspRoutineFncSignature](#) is set to [ROUTINE_FNC_PROXY](#) the [Dcm](#) module shall pass the `routineControlOptionRecord` to the called function and send back the resulting `routineStatusRecord` via the

dataInOut parameter where the currentDataLength denotes the size of the passed routineControlOptionRecord and returned routineStatusRecord.]

[SWS_Dcm_00400] [When receiving a request for [UDS Service RoutineControl \(0x31\)](#) with subfunction startRoutine, if all verifications have been successfully done (see [\[SWS_Dcm_00568\]](#), [\[SWS_Dcm_00570\]](#), [\[SWS_Dcm_00571\]](#)), the [Dcm](#) module shall call the configured Xxx_Start() function passing the dataIn, calculated from routineControlOptionRecord (see [\[SWS_Dcm_00590\]](#)), and the dataOut reference according of the list of output signal configured for this routine (see configuration parameter [DcmDspStartRoutineOut](#)).]

[SWS_Dcm_00401] [Upon completing [\[SWS_Dcm_00400\]](#), when Xxx_Start() returns E_OK, the [Dcm](#) module shall reply with a positive response with the data returned by Xxx_Start() in the dataOut as routineStatusRecord (dataOut are merged according to the list of output signal configured for this routine (see configuration parameter [DcmDspStartRoutineOut](#))).]

[SWS_Dcm_01615] DRAFT [When receiving a request for [UDS Service RoutineControl \(0x31\)](#) with subfunction startRoutine, if all verifications have been successfully done (see [\[SWS_Dcm_00568\]](#), [\[SWS_Dcm_00570\]](#), [\[SWS_Dcm_00571\]](#)) and [DcmDspRoutineFncSignature](#) is set to [ROUTINE_FNC_PROXY](#), the [Dcm](#) module shall call the configured Xxx_Start() function passing the routineControlOptionRecord via (see [\[SWS_Dcm_01614\]](#)) and if Xxx_Start() returned E_OK reply the returned routineStatusRecord with a positive response.]

[SWS_Dcm_00402] [When receiving a request for [UDS Service RoutineControl \(0x31\)](#) with subfunction stopRoutine, if all verifications have been successfully done (see [\[SWS_Dcm_00568\]](#), [\[SWS_Dcm_00570\]](#), [\[SWS_Dcm_00571\]](#)), the [Dcm](#) module shall call the configured Xxx_Stop() function passing the dataIn, calculated from routineControlOptionRecord (see [\[SWS_Dcm_00590\]](#)), and the dataOut reference according of the list of output signal configured for this routine (see configuration parameter [DcmDspStopRoutineOut](#)).]

[SWS_Dcm_00403] [Upon completing [\[SWS_Dcm_00402\]](#), when Xxx_Stop() returns E_OK, the [Dcm](#) module shall reply with a positive response with the data returned by Xxx_Stop() in the dataOut as routineStatusRecord (dataOut are merged according to the list of output signal configured for this routine (see configuration parameter [DcmDspStopRoutineOut](#))).]

[SWS_Dcm_01616] DRAFT [When receiving a request for [UDS Service RoutineControl \(0x31\)](#) with subfunction stopRoutine, if all verifications have been successfully done (see [\[SWS_Dcm_00568\]](#), [\[SWS_Dcm_00570\]](#), [\[SWS_Dcm_00571\]](#)) and [DcmDspRoutineFncSignature](#) is set to [ROUTINE_FNC_PROXY](#), the [Dcm](#) module shall call the configured Xxx_Stop() function passing the routineControlOptionRecord via (see [\[SWS_Dcm_01614\]](#)) and if Xxx_Stop() returned E_OK reply the returned routineStatusRecord with a positive response.]

[SWS_Dcm_00404] [When receiving a request for UDS Service RoutineControl (0x31) with subfunction requestRoutineResults, if all verifications have been successfully done (see [SWS_Dcm_00568], [SWS_Dcm_00570], [SWS_Dcm_00571]), the Dcm module shall call the configured Xxx_RequestResults() function passing the dataIn, calculated from routineControlOptionRecord (see [SWS_Dcm_00590]) and provide the dataOut reference according of the list of output signal configured for this routine (see configuration parameter DcmDspRequestRoutineResultsOut).]

[SWS_Dcm_00405] [Upon completing [SWS_Dcm_00404], when Xxx_RequestResults() returns E_OK, the Dcm module shall reply with a positive response with the data returned by Xxx_RequestResults() in the dataOut as routineStatusRecord (dataOut are merged according to the list of output signal configured for this routine (see configuration parameter DcmDspRequestRoutineResultsOut)).]

[SWS_Dcm_01617] DRAFT [When receiving a request for UDS Service RoutineControl (0x31) with subfunction requestRoutineResults, if all verifications have been successfully done (see [SWS_Dcm_00568], [SWS_Dcm_00570], [SWS_Dcm_00571]) and DcmDspRoutineFncSignature is set to ROUTINE_FNC_PROXY, the Dcm module shall call the configured Xxx_RequestResults() function passing the routineControlOptionRecord via (see [SWS_Dcm_01614]) and if Xxx_RequestResults() returned E_OK, reply the returned routineStatusRecord with a positive response.]

[SWS_Dcm_00641] [To serialize the required AUTOSAR data types (signed- and unsigned integer) from the request message / into the response message of UDS Service RoutineControl, the target endianness configured in DcmDspRoutineSignalEndianness shall be considered for DcmDspRoutine signals having DcmDspRoutineSignalType set to fixed length (DcmDspRoutineSignalType set to other value than VARIABLE_LENGTH).]

[SWS_Dcm_CONSTR_06072] Dependency for DcmDspRoutineSignalEndianness [In case DcmDspRoutineSignalEndianness is not present, the DcmDspDataDefaultEndianness shall be used instead.]

A positive response of a routine control service allows the use of the optional "routine-Info" byte according to ISO 14229-1 [1]. If this shall be used, it can be provided by application via a standard routine output parameter (e.g. in case the routineInfo byte is not static and the result depends on some application provided state) or the Dcm can take the statically configure value from DcmDspRoutineInfoByte.

[SWS_Dcm_01135] Routine info byte support

Upstream requirements: RS_Diag_04224

[If DcmDspRoutineInfoByte is configured for a routine control, the Dcm shall add this configured value as static response parameter "routineInfo" to the positive response according to ISO 14229-1 [1].]

[SWS_Dcm_01139] [The Dcm shall follow the NRC handling for RoutineControlService according to ISO 14229-1 [1].]

[SWS_Dcm_01140] [On reception of the UDS Service RoutineControl (0x31), the Dcm module shall check the overall length of the request. If length of the request is wrong, the Dcm module shall send NRC 0x13 (Incorrect message length or invalid format) to the tester.]

[SWS_Dcm_01141] [The Dcm shall call the appropriate routine functions of the SWC after having performed the total length check and the Mode rules, security level and session checks (DcmDspStartRoutineCommonAuthorizationRef, DcmDspStopRoutineCommonAuthorizationRef and DcmDspRequestRoutineResultsCommonAuthorizationRef).]

Note: Subsequent checks have to be performed by the SWC.

7.4.2.15.1 Handling of OBD RID

The Dcm supports the so called OBD mirror functionality to use the same application interface for SAE-J1979 [4] Mode \$08 interfaces as for UDS RoutineControls in the OBD range. It is only applicable for RIDs within the range (E000-E0FF) (the one byte mode \$08 test ID). This mode is enabled with setting DcmDspEnableObdMirror to TRUE, which means that for the UDS RIDs in the range (E000-E0FF) the API Xxx_RequestControl() is used, even if the diagnostic request has a UDS based origin.

[SWS_Dcm_01330] [If DcmDspEnableObdMirror is set to true, an explicitly configured RID inside the OBD range (E000-E0FF) shall use the UDS interface.]

[SWS_Dcm_01331] [If DcmDspEnableObdMirror is set to false, all requests within the OBD RID range shall use the UDS interface.]

[SWS_Dcm_01332] [On reception of the UDS Service RoutineControl (0x31), for every requested RID inside the OBD range (E000-E0FF), the Dcm module shall handle the RID as defined for OBD Service \$08 (see [SWS_Dcm_00418], [SWS_Dcm_00947], [SWS_Dcm_00419], [SWS_Dcm_00420], [SWS_Dcm_00948], [SWS_Dcm_01192]) if DcmDspEnableObdMirror is set to true and RID not explicitly configured.]

[SWS_Dcm_01333] [On reception of the UDS Service RoutineControl (0x31), for every requested RID inside the OBD range (E000-E0FF) and usage of OBD interface, the Dcm shall use the routineInfo byte value from DcmDspRequestControlInfoByte in the response to the tester.]

[SWS_Dcm_01390] [On reception of an UDS Service RoutineControl (0x31) request with one or more "availability OBDTestIds" as parameter, the Dcm module shall respond with the corresponding supported (=configured) RIDs.]

[SWS_Dcm_01392] [On reception of an UDS Service RoutineControl (0x31) request with a OBDTestIds that is not an "availability OBDTestIds" and DcmDspEnableObdMirror is set to FALSE, the Dcm module shall invoke the configured Xxx_Start() function.]

[SWS_Dcm_01393] [As specified in [4, SAE J1979], unused data bytes shall be filled with \$00.]

[SWS_Dcm_CONSTR_06071] **Dependency for [DcmDspStartRoutineFnc](#), [DcmDspStopRoutineFnc](#), [DcmDspRequestRoutineResultsFnc](#), [DcmDspStartRoutineConfirmationFnc](#), [DcmDspStopRoutineConfirmationFnc](#)** [The following configuration parameters shall only be present if [DcmDspRoutineUsePort](#) is set to FALSE.

- [DcmDspStartRoutineFnc](#)
- [DcmDspStopRoutineFnc](#)
- [DcmDspRequestRoutineResultsFnc](#)
- [DcmDspStartRoutineConfirmationFnc](#)
- [DcmDspStopRoutineConfirmationFnc](#)

]

7.4.2.16 Service 0x3E - Tester Present

[SWS_Dcm_00251] [The [Dcm](#) module shall implement the Tester Present (service 0x3E, diagnostic communication and security) of the Unified Diagnostic Services for the subfunction values 0x00 and 0x80.]

[SWS_Dcm_01558] Skipping authentication check for tester present

Upstream requirements: [RS_Diag_04230](#)

[The Dcm shall process the UDS service 0x3E (TesterPresent) independently from the current authentication state.]

This service is used to keep one or multiple servers in a diagnostic session being different than the defaultSession.

7.4.2.17 Service 0x3D - WriteMemoryByAddress

[SWS_Dcm_00488] [The [Dcm](#) module shall implement the WriteMemoryByAddress (service 0x3D) of the Unified Diagnostic Services.]

This service is used to write data using a physical memory address.

[SWS_Dcm_00855] [On reception of the [UDS Service WriteMemoryByAddress](#) (0x3D), the [Dcm](#) shall check if the requested [AddressAndLengthFormatIdentifier](#) is supported (refer to configuration parameter [DcmDspSupportedAddressAndLengthFormatIdentifier](#)), Otherwise the [NRC 0x31](#) (requestOutOfRange) shall

be responded. In case the container `AddressAndLengthFormatIdentifier` is not present, the `Dcm` shall accept all possible `AddressAndLengthFormatIdentifiers`.]

[SWS_Dcm_00489] [On reception of the `UDS` Service `WriteMemoryByAddress` (0x3D), the `Dcm` shall check if the complete memory range to write to (from 'memoryAddress' parameter to 'memoryAddress + memorySize -1') is inside the allowed memory ranges (check of `DcmDspWriteMemoryRangeLow` and `DcmDspWriteMemoryRangeHigh` parameters for each `DcmDspWriteMemoryRangeInfo` container or `DcmDspWriteMemoryRangeByLabelLow` and `DcmDspWriteMemoryRangeByLabelHigh` parameters for each `DcmDspWriteMemoryRangeByLabelInfo` container). If not, the `Dcm` module shall send `NRC` 0x31 (Request out of range).]

[SWS_Dcm_00490] [On reception of the `UDS` Service `WriteMemoryByAddress` (0x3D), the `Dcm` shall check if the complete memory range (from 'memoryAddress' parameter to 'memoryAddress + memorySize -1') can be written in the current security level (see `DcmDspWriteMemoryRangeSecurityLevelRef`). If security level is not correct, the `Dcm` module shall send `NRC` 0x33 (securityAccessDenied).]

[SWS_Dcm_00825] [On reception of the `UDS` Service `WriteMemoryByAddress` (0x3D), the `Dcm` shall check if the complete memory range (from 'memoryAddress' parameter to 'memoryAddress + memorySize -1') can be written in the current mode condition (see `DcmDspWriteMemoryRangeModeRuleRef`). If mode condition is not correct, the `Dcm` module shall send the calculated negative response code of the referenced `dcmModeRule`.]

[SWS_Dcm_00491] [On reception of the `UDS` Service `WriteMemoryByAddress` (0x3D), and after verification of the validity of the request (see **[SWS_Dcm_00489]** and **[SWS_Dcm_00490]**) the `Dcm` module shall call the callout `Dcm_WriteMemory`.]

[SWS_Dcm_01052] [On reception of the `UDS` Service `WriteMemoryByAddress` (0x3D), if the request message contains different `MemoryIdValue` compare to the configured values in `DcmDspMemoryIdInfo` container, the `Dcm` shall send a `NRC` 0x31 (RequestOutOfRange).]

[SWS_Dcm_01056] [The configured ranges of memory address (`DcmDspReadMemoryRangeHigh` and `DcmDspReadMemoryRangeLow` or `DcmDspReadMemoryRangeByLabelHigh` and `DcmDspReadMemoryRangeByLabelLow`) shall not overlap each other.]

[SWS_Dcm_01358] [On reception of the `UDS` Service `WriteMemoryByAddress` (0x3D), the `Dcm` shall check if the complete memory range (from 'memoryAddress' parameter to 'memoryAddress + memorySize -1') can be written in the current session (see `DcmDspWriteMemoryRangeSessionLevelRef`). If the session is not correct, the `Dcm` module shall send `NRC` 0x31 (RequestOutOfRange).]

[SWS_Dcm_00643] [If the operation `Dcm_WriteMemory` returns `DCM_WRITE_FAILED`, the `Dcm` module shall send a negative response with `NRC` code equal to the parameter `ErrorCode` parameter value.]

[SWS_Dcm_00837]

Upstream requirements: [RS_Diag_04249](#)

[If the call to [Dcm_WriteMemory](#) returns DCM_WRITE_FORCE_RCRRP, the [Dcm](#) shall invoke the transmit request for RCR-RP (NRC 0x78 transmission) and the [Dcm](#) shall not realize further invocation of the operation till RCR-RP is transmitted.]

[SWS_Dcm_00838] [After transmit confirmation of a RCR-RP transmitted on the context of [\[SWS_Dcm_00837\]](#), the [Dcm](#) calls, from [Dcm_MainFunction](#) (due to call context), [Dcm_WriteMemory](#) again with OpStatus = DCM_FORCE_RCRRP_OK.]

7.4.2.18 Service 0x23 - ReadMemoryByAddress

This service is used to read data using a physical memory address.

[SWS_Dcm_00492] [The [Dcm](#) module shall implement the ReadMemoryByAddress (service 0x23) of the Unified Diagnostic Services.]

[SWS_Dcm_00853] [On reception of the [UDS](#) Service ReadMemoryByAddress (0x23), the [Dcm](#) shall check if the requested [AddressAndLengthFormatIdentifier](#) is supported (refer to configuration parameter [DcmDspSupportedAddressAndLengthFormatIdentifier](#)), Otherwise the NRC 0x31 (requestOutOfRange) shall be responded. In case the container [DcmDspAddressAndLengthFormatIdentifier](#) is not present, the [Dcm](#) shall accept all possible AddressAndLengthFormatIdentifiers.).]

[SWS_Dcm_00493] [On reception of the [UDS](#) Service ReadMemoryByAddress (0x23), the [Dcm](#) shall check if the complete memory range to read from (from 'memoryAddress' parameter to 'memoryAddress + memorySize -1') is inside the allowed memory ranges (check of [DcmDspReadMemoryRangeLow](#) and [DcmDspReadMemoryRangeHigh](#) parameters for each [DcmDspReadMemoryRangeInfo](#) container or [DcmDspReadMemoryRangeByLabelLow](#) and [DcmDspReadMemoryRangeByLabelHigh](#) parameters for each [DcmDspReadMemoryRangeByLabelInfo](#) container). If not, the [Dcm](#) module shall send NRC 0x31 (Request out of range).]

[SWS_Dcm_00494] [On reception of the [UDS](#) Service ReadMemoryByAddress (0x23), the [Dcm](#) shall check if the complete memory range (from 'memoryAddress' parameter to 'memoryAddress + memorySize -1') can be readen in the current security level (see [DcmDspReadMemoryRangeSecurityLevelRef](#)). If security level is not correct, the [Dcm](#) module shall send NRC 0x33 (securityAccessDenied).]

[SWS_Dcm_00826] [On reception of the [UDS](#) Service ReadMemoryByAddress (0x23), the [Dcm](#) shall check if the complete memory range (from 'memoryAddress' parameter to 'memoryAddress + memorySize -1') can be readen in the current mode condition (see [DcmDspReadMemoryRangeModeRuleRef](#)). If mode condition is not

correct, the `Dcm` module shall send calculated negative response code of the referenced `DcmModeRule`.]

[SWS_Dcm_00495] [On reception of the `UDS` Service `ReadMemoryByAddress` (0x23), and after verification of the validity of the request (see [SWS_Dcm_00493] and [SWS_Dcm_00494]) the `Dcm` module shall call the callout `Dcm_ReadMemory`.]

[SWS_Dcm_01053] [On reception of the `UDS` Service `ReadMemoryByAddress` (0x23), if the request message contains different `MemoryIdValue` compare to the configured values in `DcmDspMemoryIdInfo` container, the `Dcm` shall send a `NRC` 0x31 (`RequestOutOfRange`).]

[SWS_Dcm_01158] [The configured ranges of memory address (`DcmDspReadMemoryRangeHigh` and `DcmDspReadMemoryRangeLow` or `DcmDspReadMemoryRangeByLabelHigh` and `DcmDspReadMemoryRangeByLabelLow`) shall not overlap each other.]

[SWS_Dcm_01359] [On reception of the `UDS` Service `ReadMemoryByAddress` (0x23), the `Dcm` shall check if the complete memory range (from 'memoryAddress' parameter to 'memoryAddress + memorySize - 1') can be read in the current session (see `DcmDspReadMemoryRangeSessionLevelRef`). If the session is not correct, the `Dcm` module shall send `NRC` 0x31 (`RequestOutOfRange`).]

[SWS_Dcm_00644] [If the operation `Dcm_ReadMemory` returns `DCM_READ_FAILED`, the `Dcm` module shall send a negative response with `NRC` code equal to the parameter `ErrorCode` parameter value.]

[SWS_Dcm_00839]

Upstream requirements: [RS_Diag_04249](#)

[If the call to `Dcm_ReadMemory` returns `DCM_READ_FORCE_RCRRP`, the `Dcm` shall invoke the transmit request for RCR-RP (`NRC` 0x78 transmission) and the `Dcm` shall not realize further invocation of the operation till RCR-RP is transmitted.]

[SWS_Dcm_00840] [After transmit confirmation of a RCR-RP transmitted on the context of [SWS_Dcm_00839], the `Dcm` calls, from `Dcm_MainFunction` (due to call context), `Dcm_ReadMemory` again with `OpStatus` = `DCM_FORCE_RCRRP_OK`.]

7.4.2.19 Service 0x34 - RequestDownload

This service is used to request the start of a download process.

[SWS_Dcm_00496]

Upstream requirements: [RS_Diag_04033](#)

[The `Dcm` module shall implement the `RequestDownload` (service 0x34) of the Unified Diagnostic Services.]

[SWS_Dcm_00856] [On reception of the UDS ServiceRequestDownload (0x34), the Dcm shall check if the requested AddressAndLengthFormatIdentifier is supported (refer to configuration parameter DcmDspSupportedAddressAndLengthFormatIdentifier), Otherwise the NRC 0x31 (requestOutOfRange) shall be responded. In case the container AddressAndLengthFormatIdentifier is not present, the Dcm shall accept all possible AddressAndLengthFormatIdentifiers.]

[SWS_Dcm_01057] [On reception of the UDS ServiceRequestDownload (0x34), if the request message contains different MemoryIdValue compare to the configured values in DcmDspMemoryIdInfo container, the Dcm shall send a NRC 0x31 (RequestOutOfRange).]

[SWS_Dcm_01132] [NRC described in [SWS_Dcm_01662] shall be the responsibility of the callout function.]

[SWS_Dcm_01662] NRC managed by callout function for service 0x34 [

NRC	Use Case
0x31 requestOutOfRange	The specified dataFormatIdentifier is not valid.
0x70 uploadDownload-NotAccepted	An attempt to download to a server's memory cannot be accomplished due to some fault conditions. Note: this NRC will be handled by the callout only if mode rule is not used for this case

]

Note: the callout function can, if needed, return also other NRC but the ones above won't be treated by the Dcm module.

[SWS_Dcm_01591] Processing request download without RTE

Upstream requirements: RS_Diag_04033

[If DcmDspMemoryTransferUsePort is set to FALSE, the Dcm shall call Dcm_ProcessRequestDownload to process the request.]

[SWS_Dcm_00757] [If the operation Dcm_ProcessRequestDownload returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.]

[SWS_Dcm_01417]

Upstream requirements: RS_Diag_04033

[Upon calling Dcm_ProcessRequestDownload, the Dcm shall write the maximum possible buffer size into the BlockLength parameter.]

[SWS_Dcm_01418]

Upstream requirements: [RS_Diag_04033](#)

[If the function call [Dcm_ProcessRequestDownload](#) returns a requested buffer length larger than the supported buffer length of the current protocol connection, the [Dcm](#) shall report the [Det](#) error [DCM_E_INTERFACE_BUFFER_OVERFLOW](#) .]

For definition of [DCM_E_INTERFACE_BUFFER_OVERFLOW](#) see [[SWS_Dcm_00040](#)].

[SWS_Dcm_01419]

Upstream requirements: [RS_Diag_04033](#)

[If the function call [Dcm_ProcessRequestDownload](#) returns a requested buffer length smaller or equal than the supported buffer length of the current protocol connection, the [Dcm](#) shall return the [BlockLength](#) value within the [maxNumberOfBlockLength](#) parameter of the positive response.]

7.4.2.20 Service 0x35 - RequestUpload

This service is used to request the start of a upload process.

[SWS_Dcm_00499]

Upstream requirements: [RS_Diag_04033](#)

[The [Dcm](#) module shall implement the RequestUpload (service 0x35) of the Unified Diagnostic Services.]

[SWS_Dcm_00857] [On reception of the [UDS](#) RequestUpload (0x35), the [Dcm](#) shall check if the requested [AddressAndLengthFormatIdentifier](#) is supported (refer to configuration parameter [DcmDspSupportedAddressAndLengthFormatIdentifier](#)), Otherwise the [NRC](#) 0x31 (requestOutOfRange) shall be responded. In case the container [AddressAndLengthFormatIdentifier](#) is not present, the [Dcm](#) shall accept all possible [AddressAndLengthFormatIdentifiers](#).]

[SWS_Dcm_01055] [On reception of the [UDS](#) RequestUpload (0x35), if the request message contains different [MemoryIdValue](#) compare to the configured values in [DcmDspMemoryIdInfo](#) container, the [Dcm](#) shall send a [NRC](#) 0x31 (RequestOutOfRange).]

[SWS_Dcm_01133] [[NRC](#) described in [[SWS_Dcm_01663](#)] shall be the responsibility of the callout function.]

[SWS_Dcm_01663] NRC managed by callout function for service 0x35 [

NRC	Use Case
0x31 - requestOutOfRange	The specified dataFormatIdentifier is not valid.

NRC	Use Case
0x70 - uploadDownload-NotAccepted	An attempt to download to a server's memory cannot be accomplished due to some fault conditions. Note: this NRC will be handled by the callout only if mode rule is not used for this case

]

Note: the callout function can, if needed, return also other NRC but the ones above won't be treated by the Dcm module.

[SWS_Dcm_01592] Processing request upload without RTE

Upstream requirements: RS_Diag_04033

[If DcmDspMemoryTransferUsePort is set to FALSE, the Dcm shall call Dcm_ProcessRequestUpload to process the request.]

[SWS_Dcm_00758] [If the operation Dcm_ProcessRequestUpload returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.]

[SWS_Dcm_01420]

Upstream requirements: RS_Diag_04033

[Upon calling Dcm_ProcessRequestUpload, the Dcm shall write the maximum possible buffer size into the BlockLength parameter.]

[SWS_Dcm_01421]

Upstream requirements: RS_Diag_04033

[If the function call Dcm_ProcessRequestUpload returns a requested buffer length larger than the supported buffer length of the current protocol connection, the Dcm shall report the Det error DCM_E_INTERFACE_BUFFER_OVERFLOW .]

For definition of DCM_E_INTERFACE_BUFFER_OVERFLOW see [SWS_Dcm_01416].

[SWS_Dcm_01422]

Upstream requirements: RS_Diag_04033

[If the function call Dcm_ProcessRequestUpload returns a requested buffer length smaller or equal than the supported buffer length of the current protocol connection, the Dcm shall return the BlockLength value within the maxNumberOfBlockLength parameter of the positive response.]

7.4.2.21 Service 0x36 - TransferData

This service is used to transfer data during a download or upload process.

[SWS_Dcm_00502]

Upstream requirements: [RS_Diag_04033](#)

[The [Dcm](#) module shall implement the TransferData (service 0x36) of the Unified Diagnostic Services.]

To execute the transfer data request, it is mandatory to have a previous received RequestDownload or RequestUpload according to ISO14229-1 [1]. The [Dcm](#) will follow this requirements and track internally if such a request was received before the TransferData request. This is relevant to check if a [NRC](#) 0x24 sequence error has occurred and to reset the block sequence counter.

[SWS_Dcm_01620] Request sequence error check [The [Dcm](#) shall implement the requestSequenceErrorCheck (NRC 0x24) for RequestDownload, RequestUpload and RequestFileTransfer operations according to ISO14229-1 [1].]

The [Dcm](#) does not track if the data transfer is completed by the memory size parameter. The [NRC](#) 0x24 check for this has to be done in the application. This is a deviation to ISO14229-1 [1] where the [NRC](#) 0x24 check is done on only one place.

[SWS_Dcm_00645] Block sequence counter check [The [Dcm](#) shall realize the block sequence counter error verification according to ISO14229-1 [1] before the Read or Write Data calls are realized and return the [NRC](#) 0x73 if the block sequence counter of the request is wrong.]

[SWS_Dcm_00503] Processing transfer data write without [RTE](#)

Upstream requirements: [RS_Diag_04033](#)

[On reception of the [UDS](#) Service TransferData (0x36), if a download process is running (RequestDownload service has been previously received), the request format is correct and [DcmDspMemoryTransferUsePort](#) is set to FALSE, the [Dcm](#) module shall call the callout [Dcm_ProcessTransferDataWrite](#).]

[SWS_Dcm_00504] Processing transfer data read without [RTE](#)

Upstream requirements: [RS_Diag_04033](#)

[On reception of the [UDS](#) Service TransferData (0x36), if an upload process is running (RequestUpload service has been previously received), the request format is correct and [DcmDspMemoryTransferUsePort](#) is set to FALSE, the [Dcm](#) module shall call the callout [Dcm_ProcessTransferDataRead](#).]

[SWS_Dcm_01444] [On reception of the [UDS](#) Service TransferData (0x36), if a file download is running (RequestFileTransfer service has been previously received with 0x01 (AddFile) or 0x03 (ReplaceFile)) and the request format is correct, the [Dcm](#) module shall call the callout [Dcm_WriteFile](#)(.).]

[SWS_Dcm_01445] [On reception of the [UDS](#) Service TransferData (0x36), if a file or directory information upload is running (RequestFileTransfer service has been previously received with 0x04 (ReadFile) or 0x05 (ReadDir)) and the request format is correct, the [Dcm](#) module shall call the callout Dcm_ReadFileOrDir().]

[SWS_Dcm_01173] [NRCs described in [\[SWS_Dcm_01665\]](#) shall be the responsibility of the callout function.]

[SWS_Dcm_01665] NRC managed by callout function for service 0x36 [

NRC	Use Case
0x24 - requestSequenceError	only for the following conditions: If the RequestDownload or RequestUpload service is active, but the server has already received all data as determined by the memorySize parameter in the active RequestDownload or RequestUpload service
0x31 - requestOutOfRange	Only for the following conditions: The transferRequestParameterRecord contains additional control parameters (e.g. additional address information) and this control information is invalid. The transferRequestParameterRecord is not consistent with the server's memory alignment constraints
0x71 - transferDataSuspended	The data transfer operation was halted due to some fault.
0x72 - generalProgrammingFailure	If the server detects an error when finalizing the data transfer between the client and server (e.g., via an integrity check).
0x92 - voltageTooHigh	The voltage measured is higher than the maximum acceptable voltage for downloading data.
0x93 - voltageTooLow	The voltage measured is under the minimum acceptable voltage for downloading data.

]

Note: the callout function can, if needed, return also other NRCs but the ones above won't be treated by the [Dcm](#) module.

7.4.2.22 Service 0x37 - RequestTransferExit

This service is used to terminate a download or upload process.

[SWS_Dcm_00505]

Upstream requirements: [RS_Diag_04033](#)

[The [Dcm](#) module shall implement the RequestTransferExit (service 0x37) of the Unified Diagnostic Services.]

[SWS_Dcm_01134] [[NRC](#) described in [\[SWS_Dcm_01664\]](#) shall be the responsibility of the callout function.]

[SWS_Dcm_01664] NRC managed by callout function for service 0x37 [

NRC	Use Case
0x24 - requestSequenceError	The programming process is not completed when a request for this service is received.
0x72 - generalProgrammingFailure	If the server detects an error when finalizing the data transfer between the client and server (e.g., via an integrity check).

]

Note: the callout function can, if needed, return also other NRC but the ones above won't be treated by the Dcm module.

[SWS_Dcm_01593] Processing transfer exit without RTE

Upstream requirements: RS_Diag_04033

[If DcmDspMemoryTransferUsePort is set to FALSE, the Dcm shall call Dcm_ProcessRequestTransferExit to process the request.]

[SWS_Dcm_00759] [If the operation Dcm_ProcessRequestTransferExit returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.]

[SWS_Dcm_01594] Usage of API for RequestTransferExit processing

Upstream requirements: RS_Diag_04033

[If DcmDspMemoryTransferUsePort is set to false, the Dcm shall use the API Dcm_ProcessRequestTransferExit to process the RequestTransferExit request.]

[SWS_Dcm_01595] Usage of C/S interface for RequestTransferExit processing

Upstream requirements: RS_Diag_04033

[If DcmDspMemoryTransferUsePort is set to true, the Dcm shall use the operation ProcessRequestTransferExit of the C/S interface UploadDownloadServices to process the RequestTransferExit request.]

[SWS_Dcm_01596] Mapping of transferRequestParameter

Upstream requirements: RS_Diag_04033

[For Dcm_ProcessRequestTransferExit and operation ProcessRequestTransferExit of the C/S interface UploadDownloadServices the Dcm shall use:

- the size in bytes of the transferRequestParameterRecord from the request in the parameter transferRequestParameterRecordSize
- the transferRequestParameterRecord data of the request in the parameter transferRequestParameterRecord

]

[SWS_Dcm_01597] [For [Dcm_ProcessRequestTransferExit](#) and operation [ProcessRequestTransferExit](#) of the C/S interface [UploadDownloadServices](#) the [Dcm](#) shall use the size of the implementation data type [Dcm_RequestDataArrayType](#) as in-value for the in/out parameter [transferResponseParameterRecordSize](#).]

[SWS_Dcm_01598] Mapping of transferResponseParameter to the positive response

Upstream requirements: [RS_Diag_04033](#)

[For [Dcm_ProcessRequestTransferExit](#) and operation [ProcessRequestTransferExit](#) of the C/S interface [UploadDownloadServices](#) the [Dcm](#) shall use [transferResponseParameterRecordSize](#) bytes of the [transferResponseParameterRecord](#) as [transferResponseParameterRecord](#) of the positive response.]

7.4.2.23 Service 0x38 - RequestFileTransfer

[SWS_Dcm_01083]

Upstream requirements: [RS_Diag_04135](#)

[The [Dcm](#) module shall implement the [RequestFileTransfer](#) (service 0x38) of the Unified Diagnostic Services.]

This service is used to request the start of a file transfer process according to ISO-14229-1.

[SWS_Dcm_01446]

Upstream requirements: [RS_Diag_04135](#)

[If the [DcmRequestFileTransferUsePort](#) is set to TRUE the [Dcm](#) shall use C/S calls of the interface [RequestFileTransfer](#) for all [RequestFileTransfer](#) related callouts.]

[SWS_Dcm_01447]

Upstream requirements: [RS_Diag_04135](#)

[If the [DcmRequestFileTransferUsePort](#) is set to FALSE the [Dcm](#) shall use C function calls for all [RequestFileTransfer](#) related callouts.]

[SWS_Dcm_01448]

Upstream requirements: [RS_Diag_04135](#)

[If the [fileSizeParameterLength](#) parameter in the [RequestFileTransfer](#) request is present and outside the closed interval [0x01..0x08], the [Dcm](#) shall send a negative response with [NRC 0x31](#) ([RequestOutOfRange](#)).]

[SWS_Dcm_01449]

Upstream requirements: [RS_Diag_04135](#)

[If the modeOfOperation parameter in the RequestFileTransfer request is not supported (0x00 or greater than 0x05), the [Dcm](#) shall send a negative response with [NRC](#) 0x31 (RequestOutOfRange).]

[SWS_Dcm_01450]

Upstream requirements: [RS_Diag_04135](#)

[The [Dcm](#) shall process RequestFileTransfer according to [\[6\]](#) and, in case of modeOfOperation equal to 0x01 (AddFile) call XXX_ProcessRequestAddFile after the full length check.]

[SWS_Dcm_01451]

Upstream requirements: [RS_Diag_04135](#)

[The [Dcm](#) shall process RequestFileTransfer according to [\[6\]](#) and, in case of modeOfOperation equal to 0x02 (DeleteFile) call XXX_ProcessRequestDeleteFile after the full length check.]

[SWS_Dcm_01452]

Upstream requirements: [RS_Diag_04135](#)

[The [Dcm](#) shall process RequestFileTransfer according to [\[6\]](#) and, in case of modeOfOperation equal to 0x03 (ReplaceFile) call XXX_ProcessRequestReplaceFile after the full length check.]

[SWS_Dcm_01453]

Upstream requirements: [RS_Diag_04135](#)

[The [Dcm](#) shall process RequestFileTransfer according to [\[6\]](#) and, in case of modeOfOperation equal to 0x04 (ReadFile) call XXX_ProcessRequestReadFile after the full length check.]

[SWS_Dcm_01454]

Upstream requirements: [RS_Diag_04135](#)

[The [Dcm](#) shall process RequestFileTransfer according to [\[6\]](#) and, in case of modeOfOperation equal to 0x05 (ReadDir) call XXX_ProcessRequestReadDir after the full length check.]

[SWS_Dcm_01088]

Upstream requirements: [RS_Diag_04135](#)

[If any of the file transfer operations XXX_ProcessRequest<yyy> returns value E_NOT_OK, the [Dcm](#) module shall send a negative response with [NRC](#) code equal to the parameter ErrorCode parameter value.]

[SWS_Dcm_01455]

Upstream requirements: [RS_Diag_04135](#)

[The [Dcm](#) shall use the value configured in [DcmRequestFileTransferFileSizeOrDirInfoParameterLength](#) as value sent in [fileSizeOrDirInfoParameterLength](#) in the response.]

[SWS_Dcm_01090]

Upstream requirements: [RS_Diag_04135](#)

[The [Dcm](#) shall use the value configured in [DcmRequestFileTransferFileSizeOrDirInfoParameterLength](#) as number of bytes sent in [fileSizeUncompressedOrDirInfoLength](#) and [fileSizeCompressed](#) in the response.]

[SWS_Dcm_01456]

Upstream requirements: [RS_Diag_04135](#)

[The [Dcm](#) shall use the value configured in [DcmRequestFileTransferLengthFormatIdentifier](#) as value sent in [lengthFormatIdentifier](#) in the response.]

[SWS_Dcm_01091]

Upstream requirements: [RS_Diag_04135](#)

[The [Dcm](#) shall use the value configured in [DcmRequestFileTransferLengthFormatIdentifier](#) as number of bytes sent in [maxNumberOfBlockLength](#) in the response.]

[SWS_Dcm_01457]

Upstream requirements: [RS_Diag_04135](#)

[If the [maxNumberOfBlockLength](#) does not fit into the requested [lengthFormatIdentifier](#) number of bytes, the [Dcm](#) shall send [NRC 0x10](#) ([generalReject](#)).]

[SWS_Dcm_01458]

Upstream requirements: [RS_Diag_04135](#)

[If the [fileSizeUncompressedOrDirInfoLength](#) or [fileSizeCompressed](#) do not fit into the requested [fileSizeOrDirInfoParameterLength](#) number of bytes, the [Dcm](#) shall send [NRC 0x10](#) ([generalReject](#)).]

7.4.2.24 Service 0x85 - ControlDTCSetting

An external test tool can request an ECU to either disable or enable [DTC](#) storage in the ECUs error memory by sending a [UDS](#) Service 0x85 request with sub-function 0x01 ("ON") or 0x02 ("OFF").

The ControlDTCSetting service in ISO 14229-1:2020 [9] allows the optional parameter DTCSettingControlOptionRecord. The diagnostic server instance always supports

this parameter. This effectively means that a `ControlDTCSetting` service with or without `DTCSettingControlOptionRecord` is processed in the same way. As defined in [SWS_Dcm_01399] the `DTCSettingControlOptionRecord` is limited to the one and only `0xFFFFFFFF` value.

[SWS_Dcm_00249]

Upstream requirements: [RS_Diag_04159](#)

[The `Dcm` module shall implement `UDS` Service `ControlDTCSetting` (0x85) to enable or disable the storage of `DTCs` in the ECUs error memory.]

[SWS_Dcm_01399]

Upstream requirements: [RS_Diag_04159](#)

[If the `Dcm` receives a `ControlDTCSetting` (0x85) service with `DTCSettingControlOptionRecord` != `0xFFFFFFFF`, the `Dcm` shall send a `NRC` 0x31 (`RequestOutOfRange`).]

[SWS_Dcm_01063]

Upstream requirements: [RS_Diag_04115](#)

[On reception of `UDS` Service 0x85 with subfunction 0x01 (`DTCSettingType` "ON"), the `Dcm` shall call `Dem_EnableDTCSetting()` with `ClientId` = Client Id for this `Dcm` instance (see [DcmDemClientRef](#)).]

[SWS_Dcm_00783] [In case of `Dem_EnableDTCSetting` returns `E_OK` (see [SWS_Dcm_01063]), the `Dcm` shall invoke a mode switch of the `ModeDeclarationGroupPrototype` `DcmControlDTCSetting` by calling `SchM_Switch_<bsnp>_DcmControlDTCSetting` (`RTE_MODE_DcmControlDTCSetting_ENABLEDDTCSETTING`).]

[SWS_Dcm_00406]

Upstream requirements: [RS_Diag_04115](#)

[On reception of `UDS` Service 0x85 with subfunction 0x02 (`DTCSettingType` "OFF"), the `Dcm` shall call `Dem_DisableDTCSetting()` with `ClientId` = Client Id for this `Dcm` instance (see [DcmDemClientRef](#)).]

[SWS_Dcm_00784] [In case of `Dem_DisableDTCSetting` returns `E_OK` (see [SWS_Dcm_00406]), the `Dcm` shall invoke a mode switch of the `ModeDeclarationGroupPrototype` `DcmControlDTCSetting` by calling `SchM_Switch_<bsnp>_DcmControlDTCSetting` (`RTE_MODE_DcmControlDTCSetting_DISABLEDTCSETTING`).]

[SWS_Dcm_00751] [In case the `DTCSetting` is disabled and a transitions to default session or upon any diagnostic session change where the new session does not support `UDS` Service `ControlDTCSetting` anymore, the `Dcm` module shall call `Dem_EnableDTCSetting()` with the following parameters

- `ClientId`: Client Id for this `Dcm` instance (see [DcmDemClientRef](#))

and switch the mode `DcmControlDTCSetting` to `DCM_ENABLEDTCSETTING`.]

For some use-cases the `Dcm` may re-enable the `controlDTCSetting` due to external changed mode conditions:

[SWS_Dcm_00752] [In case the `DTCSetting` is disabled and at least one referenced arbitrary `ModeDeclarationGroupPrototypes` (see configuration parameter `DcmDspControlDTCSettingReEnableModeRuleRef`) for service `ControlDTCSetting` (0x85) with `DTCSettingType` "OFF" (0x02) are not fulfilled anymore, the `Dcm` module shall call `Dem_EnabledDTCSetting()` with the following parameters:

- `ClientId`: Client Id for this `Dcm` instance (see `DcmDemClientRef`)

and switch the mode `DcmControlDTCSetting` to `DCM_ENABLEDTCSETTING`]

Note: If at least one `ModeDeclarationGroupPrototypes` is configured (see configuration parameter `DcmDspControlDTCSettingReEnableModeRuleRef`) for service `ControlDTCSetting` (0x85) with `DTCSettingType` "OFF" (0x02), it is recommended to activate the condition check for the according sub-function 0x02 (see configuration parameter `DcmDsdSubServiceModeRuleRef`).

Note: This active observation of the referenced mode declaration groups can either be achieved by polling the mode condition in each `MainFunction` cycle or by attaching to the change notification of mode declaration group (`SchM` will trigger a `BSWEntity` in `Dcm` on changes of this mode declaration group).

7.4.2.25 Service 0x86 - ResponseOnEvent (RoE)

With the `UDS` Service `ResponseOnEvent` (0x86), a tester requests the server to start or stop transmission of responses initiated by a specified server event.

[SWS_Dcm_01625] Support of ResponseOnEvent

Upstream requirements: [RS_Diag_04160](#)

[The `Dcm` shall support the `UDS` service `ResponseOnEvent` (0x86) and all subfunctions of this service according to ISO 14229-1:2020 [9].]

[SWS_Dcm_01626] Not supported storageState bit

Upstream requirements: [RS_Diag_04160](#)

[If `DcmDspRoeStoreEventSupport` is set to false and a `RoE` request is received with the `storeEvent` bit set to 1, the `Dcm` shall send a `NRC 0x31` (`requestOutOfRange`).]

[SWS_Dcm_00894] Support of eventWindowTime

Upstream requirements: [RS_Diag_04160](#)

[The `Dcm` shall support a `eventWindowTime` set to `infiniteTimeToResponse` and `powerWindowTime` according to ISO 14229-1:2020 [9].]

[SWS_Dcm_01627] DTC status changed trigger

Upstream requirements: [RS_Diag_04160](#)

[The [Dcm](#) shall use calls to `Dcm_DemTriggerOnDTCStatus()` as trigger for changes of a DTC status.]

The [Dcm](#) leaves it open who is calling the `Dcm_DemTriggerOnDTCStatus()`. However, the Dcm API is designed to be compatible to the [Dem](#) module. If the [Dem](#) is used, it is recommended to map one Dem DTC status trigger notification to `Dcm_DemTriggerOnDTCStatus()`.

[SWS_Dcm_01410] No onDTCStatusChangedEvents during ClearDiagnosticInformation

Upstream requirements: [RS_Diag_04160](#)

[While the [Dcm](#) is processing a `ClearDiagnosticInformation` request, the [Dcm](#) shall ignore the callback `Dcm_DemTriggerOnDTCStatus`.]

[SWS_Dcm_00135] Discard serviceToRespondToMessage on transmission errors

Upstream requirements: [RS_Diag_04160](#)

[The [Dcm](#) shall discard all `serviceToRespondTo` messages that could not be send due to transmission errors.]

There is not queuing for failed RoE responses for later transmission.

[SWS_Dcm_01534] Authentication check for service to respond to

Upstream requirements: [RS_Diag_04230](#)

[On transmission of the service to respond to, the [Dcm](#) shall perform the service authentication checks and send a positive response only for services that have granted access to that connection.]

[SWS_Dcm_01628] ResponseOnEvent scheduler rate

Upstream requirements: [RS_Diag_04160](#)

[The [Dcm](#) shall implement the RoE scheduler according to ISO 14229-1:2020 [9] and map the configured scheduler rate `DcmDspRoeSchedulerRate` to `ResponseOnEventSchedulerRate` from ISO 14229-1:2020 [9].]

[SWS_Dcm_01629] Maximum number of changed DIDs per scheduler

Upstream requirements: [RS_Diag_04160](#)

[The [Dcm](#) shall provide the maximum number of changed DIDs per RoE scheduler by mapping `DcmDspRoeMaxNumChangeOfDataIdentifierEvents` to `MaxNumChangeOfDataIdentifierEvents` from ISO 14229-1:2020 [9].]

[SWS_Dcm_CONSTR_06104] [DcmDspRoeMaxNumChangeOfDataIdentifierEvents shall be configured, if the Dcm is configured to support service ResponseOnEvent with subfunction onChangeOfDataIdentifier.]

[SWS_Dcm_01630] Maximum number of comparison of value DIDs per scheduler

Upstream requirements: RS_Diag_04160

[The Dcm shall provide the maximum number of comparison of DIDs per RoE scheduler by mapping DcmDspRoeMaxNumChangeOfDataIdentifierEvents to MaxNumComparisonOfValueEvents from ISO 14229-1:2020 [9].]

[SWS_Dcm_CONSTR_06105] [DcmDspRoeMaxNumChangeOfDataIdentifierEvents shall be configured, if the Dcm is configured to support service ResponseOnEvent with subfunction onComparisonOfValues.]

[SWS_Dcm_01631] Maximum supported DID length

Upstream requirements: RS_Diag_04160

[The Dcm shall provide the maximum length of a DID as input for a ResponseOnEvent event by mapping DcmDspRoeMaxSupportedDIDLength to MaxSupportedDIDLength from ISO 14229-1:2020 [9].]

[SWS_Dcm_CONSTR_06106] [DcmDspRoeMaxSupportedDIDLength shall be configured, if the Dcm is configured to support service ResponseOnEvent with subfunction onChangeOfDataIdentifier or onComparisonOfValues.]

[SWS_Dcm_01632] Maximum number of changed DTCs per scheduler

Upstream requirements: RS_Diag_04160

[The Dcm shall provide the maximum number of DTC status change events by mapping DcmDspRoeMaxNumOfStoredDTCStatusChangedEvents to MaxNumberOfStoredDTCStatusChangedEvents from ISO 14229-1:2020 [9].]

[SWS_Dcm_CONSTR_06107] [DcmDspRoeMaxNumOfStoredDTCStatusChangedEvents shall be configured, if the Dcm is configured to support the service ResponseOnEvent with subfunction onDTCStatusChange.]

7.4.2.26 Service 0x87 - LinkControl

This service is used to gain bus bandwidth for diagnostic purposes.

The Service LinkControl (0x87) is user optional. There are different project specific use cases which are not handled in the default Dcm. One use case is to switch the bandwidth in application an other use case performs an OEM bootloader jump.

Therefore the service LinkControl needs to be implemented project specific as external service (refer to Chapter 8.8 Dcm as Service-Component).

7.4.3 OBD Services

7.4.3.1 Overview

The following table defines the **OBD** Services supported by the DCM.

Relevant OBD Service Identifier	Support in the DCM
\$01	Supported
\$02	Supported
\$03	Supported
\$04	Supported
\$06	Supported
\$07	Supported
\$08	Supported
\$09	Supported
\$0A	Supported

Table 7.3: Support for **OBD services in the DCM**

7.4.3.2 General behavior

In many cases, the **Dcm** protocol allows the bundling of several requests (for example several "PIDs") and the corresponding bundling of the responses. The descriptions of the behavior for the individual services do not explicitly consider this. As the **Dcm** needs to comply with **OBD** standard (as is defined through various requirements below), the **Dcm** might need to repeat the steps defined below to parse a request and assemble a valid response.

In a vehicle there can be 3 different types of **OBD** ECUs:

- Master ECU (one per vehicle)
- Primary ECU (several per vehicle)
- Dependent / Secondary ECUs (several per vehicle)

From the Basic Software point of view Dependent / Secondary ECUs doesn't need any specific **OBD** functionality. In Dependent / Secondary ECUs OBD-relevant information will not be stored in the Basic Software (e.g. no direct communication with the scan tool). The respective **OBD** functionality might be handled in Dependent / Secondary ECUs by a SWC.

The following **OBD** requirements are only valid for Master and Primary ECUs. If necessary the **OBD** requirements differentiate between Master and Primary Requirement.

The following table gives an overview about which **OBD** functionality must be supported in a Master ECU, Primary ECU or Dependent / Secondary ECU:

Functionality	Master ECU	Primary ECU	Dependent / Secondary ECU
OBD Scantool Communication	Yes	Yes	No

Table 7.4: Overview about [OBD](#) functionality in different [OBD](#) ECUs

[SWS_Dcm_00077]

Upstream requirements: [RS_Diag_04058](#)

[When calling the DEM module for [OBD](#) services, the [Dcm](#) module shall use the following values for the parameter DTCTOrigin:

Service \$0A uses DEM_DTC_ORIGIN_PERMANENT_MEMORY

All other services use DEM_DTC_ORIGIN_OBD_RELEVANT_MEMORY]

SAE-J1979 [4] states that the external test equipment shall not request [OBD](#) Service \$01/\$06/\$09 with a mixture of "availability identifiers" and "data containing identifiers". Therefore the AUTOSAR standard does not define the [Dcm](#) behavior in this situation.

7.4.3.3 Service \$01 - Request Current Powertrain Diagnostic Data

[SWS_Dcm_00243] [The [Dcm](#) module shall implement the [OBD](#) service \$01 (Request Current Powertrain diagnostic Data) in compliance to all provisions of the [OBD](#) standard.]

Using Service \$01, an external test tool can request an emission-related ECU to return PID-values or to return the supported PIDs. [OBD](#) reserves certain PIDs for the special purpose of obtaining the list of available PIDs in a certain range. These PIDs are called "availability PIDs" and are \$00, \$20, \$40, \$60, \$80, \$A0, \$C0 and \$E0.

The [Dcm](#) collects the [PID](#) information from 1 to n SW-Cs. This applies in particular for PIDs which contain several data values for potentially different sources. Example: PID\$83 reports Nox Sensor Data for sensor 1 and sensor 2 in one composed [PID](#) which might come from different SW-C.

The [Dcm](#) configuration defines the PIDs that are available on the ECU. The [Dcm](#) configuration defines for each such PID:

- The [PID](#) Identifier (see configuration parameter [DcmDspPidIdentifier](#))
- Indication of the [PID](#) is used or not (for postbuild configuration) (see configuration parameter [DcmDspPidUsed](#))
- The size of the [PID](#) (see configuration parameter [DcmDspPidSize](#))
- The supported information for this [PID](#) (see configuration parameter [DcmDsp-PidSupportInfo](#))

- List of data (`DcmDspPidData`) for the `PID` with the following configuration for every data
 - The length of the data associated with the `PID` (see configuration parameter `DcmDspPidDataByteSize`)
 - The position of the data in the `PID` (see configuration parameter `DcmDspPidByteOffset`)
 - The reference to the supported information container (see configuration parameter `DcmDspPidDataSupportInfo`)
 - The `Xxx_ReadData()` function that the `Dcm` must call to obtain the current value of the data or the name of the port that the `Dcm` uses to obtain the current value through the `RTE` from a `SW-C` (see configuration parameters `DcmDspPidDataReadFnc` and `DcmDspPidDataUsePort`)

[SWS_Dcm_00407] [On reception of an `OBd` Service \$01 request with only "availability `PIDs`" as parameter, the `Dcm` shall respond with the corresponding supported (=configured) `PIDs` encoded according to the `OBd` standard.]

To obtain the value for a `PID`, the `Dcm` uses the configured `Xxx_ReadData()` functions for every data of the `PID`. To provide `OBd` Service \$01, the `Dcm` relies on external functions that allow it to obtain the value of the `PIDs`. There is one such function per data of every `PID` that is needed by the `DCM`.

When using a `Xxx_ReadData()` function, the `Dcm` provides a buffer of the correct length, which is filled by the function with the data `PID` value.

[SWS_Dcm_00408] [On reception of an `OBd` Service \$01 request with only `PIDs` that are not "availability `PIDs`", the `Dcm` shall obtain the current value of these `PIDs` by invoking the configured `Xxx_ReadData()` functions for every data of the `PID` and shall return these values as response to Service \$01.]

The entity providing the actual implementation of the `Xxx_ReadData()` function for a specific signal of a `PID` might be a `SW-C` or another basic software module. The origin of the function is not known to the `Dcm` but is part of the `Dcm` configuration. Some `PIDs` are provided by the `DEM`. These `PIDs` are also explicitly configured in the `Dcm` configuration and it is the responsibility of a correct `Dcm` configuration to make the `Xxx_ReadData()` function point to the correct function provided by the `DEM`.

[SWS_Dcm_CONSTR_06069] **Dependency for `DcmDspPidDataReadFnc`** [`DcmDspPidDataReadFnc` shall be only present if `DcmDspPidDataUsePort` is set to `USE_DATA_SYNCH_FNC`.]

For certain `PIDs`, the `Dem` provides the function to obtain the `PID` value. Which `PIDs` come from the `Dem` are part of the `Dcm` configuration.

Note: For `PIDs` where `Dem` provides the function, `DcmDspPidDataUsePort` for that `PID` should be set to `USE_DATA_SYNCH_FNC` and `DcmDspPidDataReadFnc` shall

point to the function `Dem_DcmReadDataOfPID<NN>` where `<NN>` represents the Id of the `PID`.

The data byte A of the PIDs contain the support status of the subsequent data bytes. Since not all data values might be available due to the particular vehicle configuration (e.g. there is only a Nox-sensor 1 available in the vehicle in the example above), the `PID` response contains in this data byte A the information about the support status of the following data values. Note, that the PIDs always contain the same number of bytes - even if not all values are really available.

[SWS_Dcm_00621] [If a `PID` contains support information (presence of `DcmDspPidDataSupportInfo` container) the `Dcm` shall add the support information in the diagnostic response.]

[SWS_Dcm_00622] [If a `PID` contains support information (presence of `DcmDspPidDataSupportInfo` container) the `Dcm` shall calculate the support information value according to the available data for this PID: for every `DcmDspPidData` container existing for this PID, the associated support information bits, referenced in `DcmDspPidDataSupportInfo`, shall be set to one]

The response to the OBD-tester needs to be composed out of the available data values. Data bytes that are not provided by an `SW-C` need to be replaced with fill-byte to obtain a complete `PID` contents.

[SWS_Dcm_00623] [When responding to OBD Service \$01, the `Dcm` shall put fill-bytes between `DcmDspPidData` in the `PID` whenever content bytes are missing in order to fit to the `PID` size (see configuration parameter `DcmDspPidSize`).]

[SWS_Dcm_00944] [The `Dcm` shall set the fill bytes to 0x00.]

Note: If other fill-bytes than 0x00 are needed by legislation, the application has to provide the value of the fill-byte.

[SWS_Dcm_00718] [To serialize the required AUTOSAR data types (signed- and unsigned integer) into the response message of OBD Service \$01 responses the target endianness configured in `DcmDspPidDataEndianness` shall be considered for `DcmDspPidData` elements having `DcmDspPidDataUsePort` set to `USE_DATA_SENDER_RECEIVER` or `USE_DATA_SENDER_RECEIVER_AS_SERVICE`. In case `DcmDspPidDataEndianness` is not present, the `DcmDspDataDefaultEndianness` shall be used instead.]

[SWS_Dcm_CONSTR_06068] Dependency for `DcmDspPidDataEndianness` [In case `DcmDspPidDataEndianness` is not present, the `DcmDspDataDefaultEndianness` shall be used instead.]

7.4.3.4 Service \$02 - Request Power Train FreezeFrame Data

[SWS_Dcm_00244] [The [Dcm](#) shall implement [OBD](#) Service \$02 (Request Power Train FreezeFrame Data) in compliance to all provisions of the [OBD](#) standard.]

For OBD-relevant FreezeFrames AUTOSAR only supports frame 0, which is the minimum required by legislation.

[SWS_Dcm_00409] [The [Dcm](#) shall ignore all requests regarding record-numbers that are not 0]

[SWS_Dcm_00972] [On reception of an [OBD](#) Service \$02 request with a mixture of "availability [PIDs](#)" and not "availability [PIDs](#)", this request shall be ignored by the [Dcm](#).]

[SWS_Dcm_00973] [When responding to [OBD](#) Service \$02, the [Dcm](#) shall put fill-bytes between [DcmDspPidData](#) in the [PID](#) whenever content bytes are missing in order to fit to the [PID](#) size (see configuration parameter [DcmDspPidSize](#)).]

[SWS_Dcm_00974] [The [Dcm](#) shall set the fill bytes to 0x00.]

Note: If other fill-bytes than 0x00 are needed by legislation, the application has to provide the value of the fill-byte.

The following sections define how specific [PIDs](#) are handled by the [Dcm](#).

7.4.3.4.1 Service \$02 - PID\$02

An external tester can request the [DTC](#) that caused a FreezeFrame to be stored by using the Service \$02 with the [PID](#) value \$02.

[SWS_Dcm_00279]

Upstream requirements: [RS_Diag_04058](#)

[On reception of a request for Service \$02 with [PID](#) \$02, the [Dcm](#) shall call [Dem_DcmGetDTCofOBDFreezeFrame\(\)](#) with [FrameNumber](#) set to 0x00 to get the [DTC](#) number.]

The [Dem](#) module returns the corresponding [DTC](#). Note that this 2-byte [DTC](#) is packed into the 4-byte data returned by the call to [Dem_DcmGetDTCofOBDFreezeFrame \(\)](#). see [Dem](#) specification on how this is done.

[SWS_Dcm_01061] [If [Dem_DcmGetDTCofOBDFreezeFrame](#) returns [E_NOT_OK](#), the [Dcm](#) shall answer positively with \$0000 (indicates no stored freeze frame data).]

7.4.3.4.2 Service \$02 - availability PID

Using Service \$02, an external tester may request the supported PIDs for a specific freeze-frame by using the "availability PIDs".

[SWS_Dcm_00284] [On reception of a service \$02 request with an "availability PID", the Dcm shall respond with the corresponding supported (=configured) PIDs encoded according to the OBD standard.]

7.4.3.4.3 Service \$02 - other PIDs

Using Service \$02, an external tester may request the values of specific PIDs in specific FreezeFrames.

[SWS_Dcm_00286] [On reception of a service \$02 request with a PID that is not an "availability PID" and is not \$02, the Dcm shall call Dem_DcmReadDataOfOBDFreezeFrame() for every data of the PID with the following parameter values:

- PID = the PID received in the OBD request
- DestBuffer = a buffer in which the callee can write the value of the PID
- BufSize = the size of the DestBuffer, this must be at least equal to the size needed to store the value of the PID as configured in the DCM
- DataElementIndexOfPid = implicit index (from 0 to n) of the DataElement calculated by Dcm according to the order of the DataElement positions in the PID (see parameter DcmDspPidByteOffset)

]

Note that is not necessary for the Dcm module to lock or unlock the record updates of the Dem module.

[SWS_Dcm_00287] [Upon the completion of [SWS_Dcm_00286], the Dcm shall generate a response message including the respective PID, FreezeFrame Number and the associated data record for the requested FreezeFrame number.]

[SWS_Dcm_01252] [If Dem_DcmReadDataOfOBDFreezeFrame() returns E_NOT_OK and a single PID is requested, the Dcm shall not provide any answer.]

[SWS_Dcm_01253] [If Dem_DcmReadDataOfOBDFreezeFrame() returns E_NOT_OK and all PIDs from the requested multiple PID(s) are not supported, the Dcm shall not provide any answer.]

[SWS_Dcm_01254] [If `Dem_DcmReadDataOfOBDFreezeFrame()` returns `E_NOT_OK` and at least one `PID` from the requested multiple `PID(s)` is supported, the `Dcm` shall send a positive response including the data of the supported `PID(s)`.]

7.4.3.5 Service \$03 \$07 \$0A - Obtaining DTCs

[SWS_Dcm_00245] [The `Dcm` module shall implement `OBD` Service \$03 (Request emission-related diagnostic trouble codes) in compliance to all provisions of the `OBD` standard.]

[SWS_Dcm_00410] [The `Dcm` module shall implement `OBD` Service \$07 (Request Emission-Related Diagnostic Trouble Codes Detected during Current or Last Completed Driving Cycle) in compliance to all provisions of the `OBD` standard.]

[SWS_Dcm_00411] [The `Dcm` module shall implement `OBD` Service \$0A (Request Emission-Related Diagnostic Trouble Codes with Permanent Status) in compliance to all provisions of the `OBD` standard.]

An external test tool can request an emission-related ECU to report all stored, pending or permanent emission-related DTCs by sending the request \$03, \$07, \$0A respectively.

[SWS_Dcm_00289] [When receiving a request for `OBD` Service \$03, the `Dcm` module shall obtain from the DEM all DTCs in primary memory and with a "confirmed" status using the functions `Dem_SetDTCFilter()` and `Dem_GetNextFilteredDTC()`.]

Note: The `Dcm` module can get an indication of the number of records that will be found using `Dem_GetNextFilteredDTC()` by using `Dem_GetNumberOfFilteredDTC()`. This allows the implementation to calculate the total size of the response before cycling through the DTCs.

[SWS_Dcm_00412] [When receiving a request for `OBD` Service \$07, the `Dcm` module shall obtain from the DEM module all DTCs in primary memory with a "pending" status using the functions `Dem_SetDTCFilter()` and `Dem_GetNextFilteredDTC()`.]

Note: The `Dcm` module can get an indication of the number of records that will be found using `Dem_GetNextFilteredDTC()` by using `Dem_GetNumberOfFilteredDTC()`. This allows the implementation to calculate the total size of the response before cycling through the DTCs.

[SWS_Dcm_00330] [When receiving a request for `OBD` Service \$0A, the `Dcm` module shall obtain from the DEM all DTCs stored in permanent memory using the functions `Dem_SetDTCFilter()` and `Dem_GetNextFilteredDTC()`.]

Note: The [Dcm](#) module can get an indication of the number of records that will be found using `Dem_GetNextFilteredDTC()` by using `Dem_GetNumberOfFilteredDTC()`. This allows the implementation to calculate the total size of the response before cycling through the DTCs.

The following table illustrates the parameters the [Dcm](#) module must use when calling `Dem_SetDTCFilter()` in response to a request for [OBD](#) Service \$03, \$07 or \$0A.

Parameters to <code>Dem_SetDTCFilter</code>			
OBD Service	\$03	\$07	\$0A
ClientId	Client Id for this Dcm instance (see DcmDemClientRef)	Client Id for this Dcm instance (see DcmDemClientRef)	Client Id for this Dcm instance (see DcmDemClientRef)
DTCStatusMask	0x08 (confirmed bit set)	0x04 (pending bit set)	0x00
DTCFormat	DEM_DTC_FORMAT_OBD	DEM_DTC_FORMAT_OBD	DEM_DTC_FORMAT_OBD
DTCOrigin	DEM_DTC_ORIGIN_OBD_RELEVANT_MEMORY	DEM_DTC_ORIGIN_OBD_RELEVANT_MEMORY	DEM_DTC_ORIGIN_PERMANENT
FilterWithSeverity	FALSE	FALSE	FALSE
DTCSeverityMask	Not relevant	Not relevant	Not relevant
FilterForFaultDetectionCounter	FALSE	FALSE	FALSE

Table 7.5: `Dem_SetDTCFilter` Parameters

When using paged buffer mechanism, in some case, it's possible that the number of [DTC](#) matching the filter change between the calculation of the total size, needed for the first page transmission, and the sending of the further pages. For this reason, the requirement [[SWS_Dcm_00587](#)] and [[SWS_Dcm_00588](#)] shall be considered for the implementation of this service.

[SWS_Dcm_01227] [`Dem_GetNextFilteredDTC` returns `DEM_NO_SUCH_ELEMENT` and at least one matching element could be retrieved before, the [Dcm](#) shall send a positive response including these data elements and the number of DTCs.]

[SWS_Dcm_01228] [If `Dem_GetNextFilteredDTC` returns `DEM_NO_SUCH_ELEMENT` and no matching element could be retrieved before, the [Dcm](#) shall send a positive response with the number of DTCs set to 0x00.]

7.4.3.6 Service \$04 - Clear/reset emission-related diagnostic information

[SWS_Dcm_00246] [The [Dcm](#) module shall implement [OBD](#) Service \$04 (Clear/reset emission-related diagnostic information) in compliance to all provisions of the [OBD](#) standard.]

An external test tool can request an emission-related ECU to clear the error memory by sending the request \$04.

[SWS_Dcm_00004]

Upstream requirements: [RS_Diag_04058](#)

[When receiving a request for [OBD](#) Service \$04, the [Dcm](#) module shall call the interface `Dem_SelectDTC` with the following parameter values:

- `ClientId`: Client Id for this [Dcm](#) instance (see [DcmDemClientRef](#))
- `DTC` = `DEM_DTC_GROUP_ALL_DTCS`
- `DTCFormat` = `DEM_DTC_FORMAT_OBD`
- `DTCOrigin` = `DEM_DTC_ORIGIN_OBD_RELEVANT_MEMORY`

]

[SWS_Dcm_00413] [In case `Dem_ClearDTC()` returns `E_OK`, the [Dcm](#) module shall send a positive response.]

[SWS_Dcm_00703]

Upstream requirements: [RS_Diag_04249](#)

[In case `Dem_ClearDTC()` returns `DEM_PENDING`, the [Dcm](#) shall invoke `Dem_ClearDTC()` on next [Dcm_MainFunction](#) call again. It is up to the [Dcm](#) to send [NRC](#) 78 (`ResponsePending`) to respect the response behaviour]

[SWS_Dcm_00704] [In case `Dem_ClearDTC()` returns `DEM_CLEAR_FAILED`, the [Dcm](#) shall send a negative response `0x22` (`conditionsNotCorrect`).]

[SWS_Dcm_00967] [In case `Dem_ClearDTC()` returns `DEM_CLEAR_BUSY`, the [Dcm](#) shall send a negative response `0x22` (`ConditionsNotCorrect`).]

[SWS_Dcm_01067] [In case `Dem_ClearDTC()` returns `DEM_CLEAR_MEMORY_ERROR`, the [Dcm](#) module shall send a negative response `0x22` (`ConditionNotCorrect`).]

7.4.3.7 Service \$06 - Request On-Board Monitoring Test-results for Specific Monitored Systems

7.4.3.7.1 General requirements

[SWS_Dcm_00414] [The [Dcm](#) module shall implement [OBD](#) Service \$06 (Request On-Board Monitoring Test-results for Specific Monitored Systems) in compliance to all provisions of the [OBD](#) standard.]

Using Service \$06, an external test tool can request an emission-related ECU to return the DTR's associated with the OBDMID or to return the supported OBDMIDs. [OBD](#) reserves certain OBDMIDs for the special purpose of obtaining the list of supported

OBDMIDs in a certain range. These OBDMIDs are called "availability OBDMIDs" and are \$00, \$20, \$40, \$60, \$80, \$A0, \$C0 and \$E0.

A tester request for supported OBDMIDs may contain up to six (6) "availability OBDMIDs".

[SWS_Dcm_00956] [On reception of an OBD Service \$06 request with multiple non-availability OBDMIDs, the Dcm module shall ignore the request.]

7.4.3.7.2 Test results obtained via Dem interaction

The maintenance of the DTRs lies within the responsibility of the DEM. SW-Cs reporting DTRs use dedicated interfaces offered by the DEM. Upon requests from the tester the Dcm retrieves the information from the DEM using dedicated DEM interfaces. There is no direct interaction between the Dcm and SW-Cs.

[SWS_Dcm_00957] [On reception of an OBD Service \$06 request with only "availability OBDMID(s)" as parameter(s), the Dcm module shall obtain the supported OBDMIDs by calling the Dem interface Dem_DcmGetAvailableOBDMIDs() for each "availability OBDMID (\$00, \$20, ...)" contained within the request and concatenate the results within the response message according to ISO-15031-5 [2].]

[SWS_Dcm_00958] [On reception of an OBD Service \$06 request with an OBDMID that is not an "availability OBDMID", the Dcm module shall call the DEM interface Dem_DcmGetNumTIDsOfOBDMID() to obtain the TIDs available for the requested OBDMID and then recurrently call the interface Dem_DcmGetDTRData() for the number of reported TIDs to obtain the associated DTR data.]

7.4.3.8 Service \$08 - Request Control of On-Board System, Test or Component

[SWS_Dcm_00417] [The Dcm module shall implement OBD Service \$08 (Request Control of On-Board System, Test or Component) in compliance to all provisions of the OBD standard.]

Using Service \$08, an external test tool can control an on-board system, test or component using a TID. OBD reserves certain TIDs for the special purpose of obtaining the list of supported TIDs in a certain range. These TIDs are called "availability TIDs" and are \$00, \$20, \$40, \$60, \$80, \$A0, \$C0 and \$E0.

The Dcm module's configuration defines the TIDs that are available on the ECU for the purpose of OBD Service \$08. The configuration defines for each such TID (see configuration parameter DcmDspRequestControlTestId):

- the name of the port the Dcm uses to access the RequestControlServices interface (see configuration parameter DcmDspRequestControl)

- the number of bytes this function takes as input (see configuration parameter [DcmDspRequestControlInBufferSize](#))
- the number of bytes this function writes as output (see configuration parameter [DcmDspRequestControlOutBufferSize](#))

To provide [OBD](#) Service \$08, the [Dcm](#) relies on external functions configured per TID.

[SWS_Dcm_00418] [On reception of an [OBD](#) Service \$08 request with one or more "availability TIDs" as parameter, the [Dcm](#) module shall respond with the corresponding supported (=configured) TIDs.]

[SWS_Dcm_00947] [On reception of an [OBD](#) Service \$08 request "availability TIDs" together with other TIDs as parameter, the [Dcm](#) module shall ignore the request.]

[SWS_Dcm_00419] [On reception of an [OBD](#) Service \$08 request with a TID that is not an "availability TID", the [Dcm](#) module shall invoke the configured [Xxx_RequestControl\(\)](#) function with the following parameters values: InBuffer: data contained in the [OBD](#) request (the size of which must correspond to the size configured in the [Dcm](#) module's configuration) OutBuffer: space in which the RequestControl function can store its result (the size of the buffer is taken from the [Dcm](#) module's configuration)]

[SWS_Dcm_00420] [After the execution of [\[SWS_Dcm_00419\]](#), the [Dcm](#) module shall respond to the service request using the data stored by the RequestControl function in the OutBuffer.]

[SWS_Dcm_00948] [As specified in [\[4, SAE J1979\]](#), unused data bytes shall be filled with \$00.]

[SWS_Dcm_01192] [If [Xxx_RequestControl\(\)](#) doesn't return [E_OK](#), the [Dcm](#) shall return [NRC 0x22](#).]

7.4.3.9 Service \$09 - Request Vehicle Information

[SWS_Dcm_00421] [The [Dcm](#) module shall implement [OBD](#) Service \$09 (Request Vehicle Information) in compliance to all provisions of the [OBD](#) standard.]

Using Service \$09, an external test tool can request vehicle information or can obtain lists of supported vehicle information. [OBD](#) reserves certain InfoTypes for the special purpose of obtaining the list of supported InfoTypes in a certain range. These Infotypes are called "availability InfoTypes" and are \$00, \$20, \$40, \$60, \$80, \$A0, \$C0 and \$E0.

The [Dcm](#) module's configuration defines the InfoTypes and associated data that are available on one or several SW-C. The configuration defines for each such InfoType:

- The value of InfoType (see configuration parameter [DcmDspVehInfoInfoType](#))
- For every data of the InfoType:

- The position of this data in the InfoType (see configuration parameter `DcmDspVehInfoDataOrder`)
- the size of the value of the InfoType data (see configuration parameter `DcmDspVehInfoDataSize`)
- the function that the `Dcm` module must call to obtain the value for this InfoType data OR the port-name through which the `Dcm` module can obtain the value for this InfoType data (see configuration parameter `DcmDspVehInfoDataReadFnc` and `DcmDspVehInfoDataUsePort`).

To provide OBD Service \$09, the `Dcm` relies on external functions that allow it to obtain the value of an InfoType data. There is one such function per InfoType data that is needed by the DCM.

When invoking a `Xxx_GetInfotypeValueData()` function, the `Dcm` module provides a buffer of the correct size in which the value of the InfoType data can be stored. The entity providing the actual implementation of the `Xxx_GetInfotypeValueData()` function for a specific InfoType data might be a SW-C or another basic software module. The origin of the function is part of the `Dcm` module's configuration.

Certain InfoTypes needed by the `Dcm` to provide Service \$09 are provided by the DEM. This is handled in the `Dcm` configuration.

[SWS_Dcm_00422] [On reception of an OBD Service \$09 request with one or more "availability InfoTypes" as parameter, the `Dcm` module shall respond with the corresponding supported (=configured) InfoTypes.]

[SWS_Dcm_00423] [On reception of an OBD Service \$09 request for an InfoType that is not an "availability InfoType", the `Dcm` module shall obtain the value of this InfoType by invoking all the configured `Xxx_GetInfotypeValueData()` function for every data of this InfoType and shall return the value as response to Service \$09]

[SWS_Dcm_00684] [In case `DcmDspVehInfoNODIProvResp` is set to FALSE, in addition to collect the available InfoType value contributions from the individual SW-C, the `Dcm` shall compute the data byte `NofDataItems` in the diagnostic response, which defines the number of DataItems included in one InfoType.]

Note: The Calculation of the Calibration Identification (CAL-ID) and Calibration Verification Number (CVN) is not a BSW Task and will not handled within the DCM.

[SWS_Dcm_01167] [In case `DcmDspVehInfoNODIProvResp` is set to TRUE, the `Dcm` shall take over the value returned by the provider and report it as `NofDataItems` in the diagnostic response.]

[SWS_Dcm_CONSTR_06045] [In case the responsibility is on provider side (`DcmDspVehInfoNODIProvResp` is set to TRUE), only one `DcmDspVehInfoData` container shall be allowed.]

[SWS_Dcm_CONSTR_06046] [In case [DcmDspVehInfoDataUsePort](#) is set to FALSE and [DcmDspVehInfoDataReadFnc](#) is set to either [Dem_DcmGetInfoTypeValue08](#) or [Dem_DcmGetInfoTypeValue0B](#) then [DcmDspVehInfoNODIProvResp](#) shall be set to TRUE.]

Note : The integrator has to make sure that the buffer determined by the [DcmDspVehInfoDataSize](#) is sufficiently sized to receive the data returned by the provider of the data.

[SWS_Dcm_01191] [If [Xxx_GetInfoTypeValueData\(\)](#) doesn't return [E_OK](#) or [E_PENDING](#), the [Dcm](#) shall return [NRC 0x22](#).]

7.4.4 Interaction usecases

The [Dcm](#) shall be able to manage a jump to the bootloader / jump due to ECUReset request. Due to the diversity of possibility to realize this jump, this will be done using callout call.

7.4.4.1 Jump to Bootloader

4 different use cases have been identified for the jump to the bootloader, if all preconditions are fulfilled and assuming the 'suppressPosRspMsgIndicationBit' flag is set to 'false':

1. The application immediately sends a final positive response and then jumps to the bootloader
2. The application first sends a [NRC 0x78](#) response, then the final positive response and afterwards jumps to the bootloader
3. The application immediately jumps to the bootloader and the bootloader sends the final positive response
4. The application first sends a [NRC 0x78](#) response, then jumps to the bootloader and the bootloader sends the final positive response

Note: In case the 'suppressPosRspMsgIndicationBit' flag is set to 'true', use case '1' and use case '3' will not issue a positive response.

[SWS_Dcm_00532]

Upstream requirements: [RS_Diag_04098](#)

[On reception of service [DiagnosticSessionControl](#) if the provided session is used to jump to OEM bootloader (parameter [DcmDspSessionForBoot](#) set to [DCM_OEM_BOOT](#) or [DCM_OEM_BOOT_RESPAPP](#)) the [Dcm](#) shall prepare the jump to the OEM bootloader (see [\[SWS_Dcm_00535\]](#)) by triggering the mode switch of [ModeDeclarationGroupPrototype](#) [DcmEcuReset](#) to [JUMPTOBOOTLOADER](#).]

Note: By this mode switch the [Dcm](#) informs the BswM to prepare the jump to the boot-loader.

[SWS_Dcm_00592]

Upstream requirements: [RS_Diag_04098](#)

[On reception of service DiagnosticSessionControl if the provided session is used to jump to System Supplier bootloader (parameter [DcmDspSessionForBoot](#) set to DCM_SYS_BOOT or DCM_SYS_BOOT_RESPAPP) the [Dcm](#) shall prepare the jump to the System Supplier bootloader (see [\[SWS_Dcm_00535\]](#)) by triggering the mode switch of ModeDeclarationGroupPrototype DcmEcuReset to JUMPTOSYSSUPPLIERBOOTLOADER]

Note: By this mode switch the [Dcm](#) informs the BswM to prepare the jump to the boot-loader.

[SWS_Dcm_01164] [In case the service DiagnosticSessionControl implies an ECU reset, the [Dcm](#) shall ignore all further requests while that reset is being processed.]

[SWS_Dcm_00654]

Upstream requirements: [RS_Diag_04098](#), [RS_Diag_04249](#)

[In case the ModeDeclarationGroupPrototype DcmEcuReset is switched to mode JUMPTOBOOTLOADER or JUMPTOSYSSUPPLIERBOOTLOADER and the configuration parameter [DcmSendRespPendOnRestart](#) is set to TRUE, the [Dcm](#) shall trigger transmission of [NRC 0x78 - RCR-RP](#).]

Note: This final transmission of [NRC 0x78](#) before switching to Bootloader shall reload the P2* timeout in the client.

[SWS_Dcm_01175] [In case the ModeDeclarationGroupPrototype DcmEcuReset can not be switched JUMPTOBOOTLOADER or JUMPTOSYSSUPPLIERBOOTLOADER, the [Dcm](#) shall answer negatively to the request with [NRC 0x22](#) (Conditions not correct).]

[SWS_Dcm_00535]

Upstream requirements: [RS_Diag_04098](#)

[If the jump to bootloader is requested (see [\[SWS_Dcm_00532\]](#), [\[SWS_Dcm_00592\]](#), the configuration parameter [DcmSendRespPendOnRestart](#) is set to TRUE (see [\[SWS_Dcm_00654\]](#)) and the configuration parameter [DcmDspSessionForBoot](#) is set to DCM_OEM_BOOT or DCM_SYS_BOOT, the [Dcm](#) shall call [Dcm_SetProgConditions](#) after a successful transmission of [NRC 0x78](#) (Response pending).]

This will allow to store all relevant information prior to jumping to the bootloader.

Note: It is up to the software integrator to decide where to store that data. Usually it will be stored in non-volatile memory like e.g. data flash. It is also acceptable to "store" this data in a RAM section which is not initialized out of reset.

[SWS_Dcm_01163]

Upstream requirements: [RS_Diag_04098](#)

[In the context of a request to jump to the bootloader (see [\[SWS_Dcm_00532\]](#) and [\[SWS_Dcm_00592\]](#)), after [Dcm_SetProgConditions](#) returns E_OK according to [\[SWS_Dcm_00535\]](#), the [Dcm](#) shall trigger the mode switch of the ModeDeclarationGroupPrototype [DcmEcuReset](#) to EXECUTE.]

[SWS_Dcm_01177]

Upstream requirements: [RS_Diag_04098](#)

[If the jump to bootloader is requested (see [\[SWS_Dcm_00532\]](#), [\[SWS_Dcm_00592\]](#), the configuration parameter [DcmSendRespPendOnRestart](#) is set to TRUE (see [\[SWS_Dcm_00654\]](#)), and the configuration parameter [DcmDspSessionForBoot](#) is set to DCM_OEM_BOOT_RESPAPP or DCM_SYS_BOOT_RESPAPP, the [Dcm](#) shall initiate the final response after a successful transmission of [NRC 0x78](#) (Response pending).]

[SWS_Dcm_00995] [If the [NRC 0x78](#) (Response Pending) response in [\[SWS_Dcm_00535\]](#) is not sent successfully the [Dcm](#) shall cancel the current request.]

[SWS_Dcm_00997] [If the [NRC 0x78](#) (Response Pending) response in [\[SWS_Dcm_00535\]](#) is not sent successfully no jump to the bootloader shall be performed]

Note: If the [NRC 0x78](#) (Response Pending) response has not been sent correctly the [Dcm](#) will stay in the application and wait for the next request from the Client.

[SWS_Dcm_01178] [In case the ModeDeclarationGroupPrototype [DcmEcuReset](#) is switched to mode JUMPTOBOOTLOADER or JUMPTOSYSSUPPLIERBOOTLOADER, the configuration parameter [DcmSendRespPendOnRestart](#) is set to FALSE and the configuration parameter [DcmDspSessionForBoot](#) is set to DCM_OEM_BOOT_RESPAPP or DCM_SYS_BOOT_RESPAPP , the [Dcm](#) shall initiate the final response]

[SWS_Dcm_01179] [In case the final response has been successfully sent according to [\[SWS_Dcm_01177\]](#) or [\[SWS_Dcm_01178\]](#), the [Dcm](#) shall call [Dcm_SetProgConditions](#)]

[SWS_Dcm_01180] [If [Dcm_SetProgConditions](#) returns E_OK according to [\[SWS_Dcm_01179\]](#), the [Dcm](#) shall trigger the mode switch of the ModeDeclarationGroupPrototype [DcmEcuReset](#) to EXECUTE.]

[SWS_Dcm_01181] [If [Dcm_SetProgConditions](#) returns E_NOT_OK according to [\[SWS_Dcm_01179\]](#), the [Dcm](#) shall not request any reset, shall not perform the jump to bootloader, and shall not switch the ModeDeclarationGroupPrototype [DcmEcuReset](#) to EXECUTE.]

[SWS_Dcm_00720] [In case the ModeDeclarationGroupPrototype DcmEcuReset is switched to mode JUMPTOBOOTLOADER or JUMPTOSYSSUPPLIER-BOOTLOADER, the configuration parameter `DcmSendRespPendOnRestart` is set to FALSE and the configuration parameter `DcmDspSessionForBoot` it set to DCM_OEM_BOOT or DCM_SYS_BOOT, the Dcm shall call `Dcm_SetProgConditions` immediately. (see [\[SWS_Dcm_00532\]](#) and [\[SWS_Dcm_00592\]](#))]

[SWS_Dcm_00719] [If `Dcm_SetProgConditions` returns E_OK according to [\[SWS_Dcm_00720\]](#), the Dcm shall trigger the mode switch of the ModeDeclarationGroupPrototype DcmEcuReset to EXECUTE without sending a NRC 0x78 (Response pending).]

In case of [\[SWS_Dcm_00719\]](#), the exact response handling depends on the state of the 'suppressPosRspMsgIndicationBit' (TRUE or FALSE) in the request message.

[SWS_Dcm_00715] [If the jump to bootloader is requested (see [\[SWS_Dcm_00532\]](#) and [\[SWS_Dcm_00592\]](#)) and if the call to `Dcm_SetProgConditions` returns E_NOT_OK (see [\[SWS_Dcm_00535\]](#) and [\[SWS_Dcm_00720\]](#)), no further action shall be taken by the Dcm and negative response NRC 0x22 (Conditions not correct) shall be returned.]

7.4.4.2 Jump due to ECUReset

On reception of an ECUReset Service 0x11 request, if the configuration parameter `DcmResponseToEcuReset` is set to AFTER_RESET, the Dcm will set the ResponseRequired flag by calling `Dcm_SetProgConditions`.

[SWS_Dcm_01423] Answer to ECUReset request

Upstream requirements: [RS_Diag_04098](#)

[On reception of an ECUReset Service 0x11 request, if `DcmResponseToEcuReset` is set to AFTER_RESET, the Dcm shall answer to EcuReset service after the reset.]

[SWS_Dcm_01424] Answer to ECUReset request

Upstream requirements: [RS_Diag_04098](#)

[On reception of an ECUReset Service 0x11 request, if `DcmResponseToEcuReset` is set to BEFORE_RESET, the Dcm shall answer to EcuReset service before the reset.]

[SWS_Dcm_01425] Answer to ECUReset request

Upstream requirements: [RS_Diag_04098](#), [RS_Diag_04249](#)

[If the Dcm initiates a reset and `DcmSendRespPendOnRestart` is set to TRUE, the Dcm shall trigger transmission of NRC 0x78 (Response pending) before the reset.]

7.4.4.3 Jump from Bootloader / ECUReset

[SWS_Dcm_00536]

Upstream requirements: [RS_Diag_04098](#)

[At [Dcm](#) initialization, the [Dcm](#) shall call [Dcm_GetProgConditions](#) to know if the initialization is the consequence of a jump from the bootloader / ECUReset.]

Note: It is the responsibility of the software integrator to ensure that the data contained in [Dcm_ProgConditionsType](#) is valid when [Dcm_Init](#) is called. E.g. if this data is stored in non-volatile memory, it may take some time to make it available after an ECU reset. This has to be taken into account when designing the ECU's startup process.

[SWS_Dcm_00537] [If the initialization of the [Dcm](#) is the consequence of a jump from the bootloader / ECUReset (see [\[SWS_Dcm_00536\]](#), the [Dcm](#) shall call [ComM_DCM_ActiveDiagnostic\(NetworkId\)](#) to request the ComManager for the full communication mode.]

[SWS_Dcm_00767]

Upstream requirements: [RS_Diag_04098](#)

[When the ComM reports full communication to the Dcm, the [Dcm](#) shall send the Response to the Service Id passed in the [Dcm_ProgConditionsType](#).]

[SWS_Dcm_00768] [If the initialization of the [Dcm](#) is the consequence of a jump from the bootloader (see [\[SWS_Dcm_00536\]](#) and the application is updated by an FLASH download ([Dcm_ProgConditionsType.ApplUpdated](#) == True), the [Dcm](#) shall call [BswM_Dcm_ApplicationUpdated\(\)](#) to notify the BswM that the application was updated.]

7.4.4.4 Flags management

7.4.4.4.1 Jump to Bootloader

[SWS_Dcm_01182] [On reception of a [UDS](#) Service 0x10 request (Diagnostic Session Control) with subfunction 0x02 (Start Programming Session), the [Dcm](#) shall set the [ReprogrammingRequest](#) flag and, if indicated for this service, the ResponseRequired flag by calling [Dcm_SetProgConditions](#).]

[SWS_Dcm_01183] [Depending on configuration parameter [DcmDspSessionForBoot](#), either the application shall send the positive response (if [suppressPosRspMs-gIndicationBit](#) = FALSE) or after an ECU reset, when the bootloader is started, it shall send a response and clear the ResponseRequired flag. In either case, the bootloader shall clear the [ReprogrammingRequest](#) flag.]

[SWS_Dcm_01185] [In case that, during jump to Bootloader, the `Dcm_SetProgConditions` API returns `E_NOT_OK`, a DET error shall be reported `DCM_E_SET_PROG_CONDITIONS_FAIL` and normal functionality shall resume.]

7.4.4.2 Jump from Bootloader

After successful reprogramming of the application software, the bootloader will update the *ApplUpdated* flag and the *ResponseRequired* flags.

After an ECU reset, when the newly programmed application is started for the first time, the `Dcm` will read the *ApplUpdated* and *ResponseRequired* flag by calling `Dcm_GetProgConditions`. During this function call the *ApplUpdated* and *ResponseRequired* flags are cleared by the integration code.

7.5 Error notification

The Default Error Tracer module is just help for BSW development and integration. It must not be contained inside the production code. The `API` is defined, but the functionality can be chosen and implemented according to the development needs (e.g. errors count, send error information via a serial interface to an external logger, and so on).

7.6 Synchronous and Asynchronous implementation

The `Dcm` can access data using an R-Port requiring either a synchronous or an asynchronous `ClientServerInterface DataServices_{Data}`. In the `Dcm` SWS, the parameter `DcmDspDataUsePort` is set to `USE_DATA_SYNCH_CLIENT_SERVER` or `USE_DATA_ASYNC_CLIENT_SERVER` or `USE_DATA_ASYNC_CLIENT_SERVER_ERROR`.

In case of `USE_DATA_SYNCH_CLIENT_SERVER`, the interface shall be compatible with the `Dem` interface "`DataServices_{Data}`" (no `OpStatus` parameter). The parameter `OpStatus` and return parameter `DCM_E_PENDING` shall only be available in case of `USE_DATA_ASYNC_CLIENT_SERVER` or `USE_DATA_ASYNC_CLIENT_SERVER_ERROR`.

Note: a `Dcm` implementation using `AsynchronousServerCallPoint` or `SynchronousServerCallPoint` when calling service processors is completely an implementation decision. This only indicates that the operation uses the status of the operation to allow an asynchronous processing by the `SW-C` (initiating a request, checking if a request is still pending, or canceling a pending request, see [SWS_Dcm_00686]).

There is no correlation to the operation signature (i.e. existence of `OpStatus` parameter and `DCM_E_PENDING` return code) that demands `AsynchronousServerCallPoint` or `SynchronousServerCallPoint` usage.

[SWS_Dcm_01187] [If an asynchronous interface is used, the [Dcm](#) shall consider the Output data (OUT) only valid after the last call to the interface that returns E_OK.]

[SWS_Dcm_01188] [If an asynchronous interface is used, the [Dcm](#) shall consider the OUT-parameter ErrorCode only valid after the last call to the interface that returns E_NOT_OK]

Note : The "last call" to the interface is the call that returns with a value that indicates that processing has finished, i.e. E_OK or E_NOT_OK.

Note : INOUT parameter are a combination of the requirements above, i.e. on each call of the interface the parameters shall have a valid in-value, and the [Dcm](#) considers the out-value valid only after the last call of the interface.

[SWS_Dcm_01189] [If an asynchronous interface is used, the [Dcm](#) shall provide the values originating from the request for the Input data (IN) on every call to the interface.]

Note: Requirements [\[SWS_Dcm_01187\]](#), [\[SWS_Dcm_01188\]](#) and [\[SWS_Dcm_01189\]](#) do not apply for functions where a deviant behaviour is explicitly specified.

7.7 DID configuration

The configuration of the [Dcm](#) contains a list of supported DIDs which can be configured in two ways:

- The individual [DID](#) configuration, which required one connection (either via a port or a c-function) per configured data element of the respective [DID](#) to access to the data (reading, writing and controlling). The interface DataServices should be used for each [DID](#) in this case.
- The [DID](#) range configuration, used to handle a set of DIDs sharing the same behavior uniformly in one [SW-C](#) with only one port-connection. The interface DataServices_DIDRange_{Range} should be used in this case. Using this configuration allows an interface optimization. The following parameters shall be configured in order to use the DIDRange optimization: [DcmDspDidRangeIdentifierLowerLimit](#) and [DcmDspDidRangeIdentifierUpperLimit](#) which delimited the range of the DIDs. [DcmDspDidRangeMaxDataLength](#) and [DcmDspDidRangeHasGaps](#).

[SWS_Dcm_01174] [If [DcmVinRef](#) is configured then the VIN shall be fetched once by the [Dcm](#) during startup by calling [Dcm_GetVin](#).]

7.7.1 Individual DID

The individual DID can be configured in `DcmDspDid`. A unique DID identifier is configured on 2 bytes in `DcmDspDidIdentifier`. In case the DID refers to other DIDs, the link between them can be configured in `DcmDspDidRef`². Each DID allows to access to signal data values (by reading and/or writing). The signal reference (to `DcmDspData`) and the position of the data in the diagnostic answer (for reading) or request (for writing) can be configured in `DcmDspDidSignal`.

The configuration of the data belonging to the DID can be provided in the container `DcmDspData` (ECUC_Dcm_00869). This container collects the following information:

- The Data endianness of the data belonging to the DID (ECUC_Dcm_00986: `DcmDspDataEndianness`)
- The length and the type of the data (ECUC_Dcm_00605: `DcmDspDataByteSize`, ECUC_Dcm_00985 : `DcmDspDataType`)
- The interface to be used to access to the data (ECUC_Dcm_00713: `DcmDspDataUsePort`)
- The interfaces to the application in order to :
 - Check if the DID can be accessed in the current conditions. This can be achieved by checking for each `DataElement` if the conditions to read the data are satisfied (ECUC_Dcm_00677: `DcmDspDataConditionCheckReadFnc` and ECUC_Dcm_00955: `DcmDspDataConditionCheckReadFncUsed`)
 - Request to freeze the current state of an `IOControl` (ECUC_Dcm_00674: `DcmDspDataFreezeCurrentStateFnc`)
 - Get the scaling information of the DID. This can be achieved by getting the scaling information for each `DataElement` (ECUC_Dcm_00676: `DcmDspDataGetScalingInfoFnc`)
 - Request the data length of a `DataElement` (ECUC_Dcm_00671: `DcmDspDataReadDataLengthFnc`)
 - Read a certain ECU signal (ECUC_Dcm_00824: `DcmDspDataReadEcuSignal`).
 - Access in reading or writing to the data (ECUC_Dcm_00669 : `DcmDspDataReadFnc`, ECUC_Dcm_00670: `DcmDspDataWriteFnc`)
 - Request to reset an `IOControl` to default value (ECUC_Dcm_00673 : `DcmDspDataResetToDefaultFnc`)
 - Request to return control to ECU of an `IOControl` (ECUC_Dcm_00672 : `DcmDspDataReturnControlToEcuFnc`)

²Overview of `DcmDspDid` container is described in chapter 10.2.5.8.1

- Request to adjust the IO signal (ECUC_Dcm_00675 : [DcmDspDataShortTermAdjustmentFnc](#))

It is also possible to configure an alternative diagnosis representation via ECUC_Dcm_00993: [DcmDspDiagnosisScaling](#).

The following example shows how to configure the containers [DcmDspDid](#) and [DcmDspData](#) for an individual [DID](#) 0xF080. This configuration allows access to a byte of data via synchronous C APIs [WriteDID_F080](#) (for writing) and [ReadDID_F080](#) (for reading).

- [DcmDspDidIdentifier](#)=0xF080
- [DcmDspDidByteOffset](#)=0
- [DcmDspDidDataRef](#)=[DcmDspData_F080](#)
- [DcmDspDataByteSize](#)=8
- [DcmDspDataType](#)=[UINT8_N](#)
- [DcmDspDataUsePort](#)=[USE_DATA_SYNCH_FNC](#)
- [DcmDspDataWriteFnc](#)=[WriteDID_F080](#)
- [DcmDspDataReadFnc](#)=[ReadDID_F080](#)

7.7.2 DID ranges

[DID](#) ranges are in general the same as the 'normal' [DID](#) read and write function, except that the [DID](#) is also passed as a parameter. This allows to treat the [DID](#) range in a switch/case in the read or the write function.

The ranges can be applied for reading ([ReadDataByIdentifier](#) 0x22) and writing ([WriteDataByIdentifier](#) 0x2E) [DIDs](#).

The ranges can be configured in ECUC_Dcm_00937 : [DcmDspDidRange](#). Each configured range is by default accessible by service 0x22 and 0x2E. In case the range should be limited to reading or writing, the referenced [DcmDspDidInfo](#) container should be defined accordingly.

It is also possible to define gaps within the range ([DcmDspDidRangeHasGaps](#)). By activating this feature, the [Dcm](#) invokes each time a [DID](#) is requested within the configured range, the operation [IsDidAvailable](#) has to check the current availability. And as the [DIDs](#) of the specified range can have different length, the length of the longest [DID](#) has to be configured ([DcmDspDidRangeMaxDataLength](#)) in order to reserve enough buffer passed to the respective function.

In general, the range functionality can also be used for a single [DID](#) if you specifically want to pass the [DID](#) as a parameter. Then lower [DID](#) and upper [DID](#) should be the same.

ReadDidRangeDataLength operation allows to request the application to return the data length of a [DID](#) Range.

[SWS_Dcm_CONSTR_06020] Definition of allowed [DID](#) access [Any defined range shall only reference via [DcmDspDidRangeInfoRef](#). The sub-containers [DcmDspDidControl](#) and [DcmDspDidDefineinDcmDspDidInfo](#) shall not be used] .]

[SWS_Dcm_CONSTR_06021] DID ranges cannot be mapped on DDDIDs, because service 0x2C DDDID does not support the range feature. Practically [DcmDspDidRangeIdentifierLowerLimit](#) and [DcmDspDidRangeIdentifierUpperLimit](#) should not include DIDs of the range 0xF200 till 0xF3FF. [Any defined range shall only reference [DcmDspDidInfo](#) via [DcmDspDidRangeInfoRef](#), having set [DcmDspDidDynamicallyDefined](#) == False.]

7.8 Secure Coding

Secure coding or secure variant coding offers the functionality to update, change, or replace parameters in the server in a secure way. Secure variant coding is an AUTOSAR standardized functionality that allows manufacturers to control which data can be written into the ECU. Only signed data is accepted and can be written into the server. In typical environments, the data is signed by the manufacturer's backend. The manufacturer can control which data can be written.

From a high level this sequence consists of:

- Bringing in a certificate that authenticates the content that shall be written
- Writing the parameter data
- Provide a signature of the data that matches the certificate public key
- Update the data in the server.

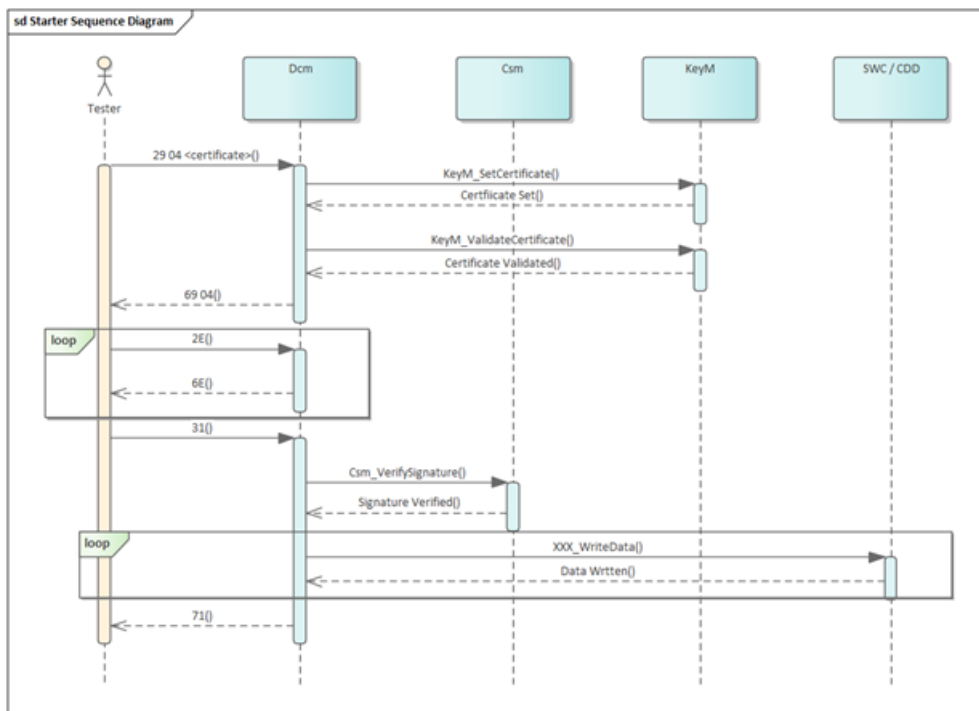


Figure 7.13: Secure Coding Sequence

The secure coding sequence is always started by transmitting a certificate using the UDS service Authentication with the sub-function transmitCertificate.

[SWS_Dcm_01684] Starting the secure coding sequence

Upstream requirements: [RS_Diag_04272](#)

[Upon receiving a certificate via the UDS service transmitCertificate with a certificateEvaluationId mapped to a `DcmDspAuthenticationTransmitCertificate` with `DcmDspAuthenticationTransmitCertificateFunction` that is set to = SECURE_CODING, the `Dcm` shall (re-)start the secure coding mechanism and use this certificate to validate the secure coding data.]

Special attention needs to be given to the mechanism to buffer the coding data until it is protected. The secure coding procedure can contain multiple consecutively sent WriteDataByIdentifier requests. The written data can only be applied and forwarded to the application of the signature calculated over all the data is valid. In the meantime, the data is buffered. There are various approaches for resource-efficient buffer handling. AUTOSAR does not have the goal of providing solutions for all possible strategies. Rather two strategies are taken into consideration:

1. Buffer all coding data within the `Dcm` prior to persisting. This approach works fine for smaller write data but scales less well for larger coding data.
2. This strategy is an outlook to use the BndM module. It might be part of future AUTOSAR releases. The BndM allows direct a/b partition NV memory access and does not require a further RAM buffer.

The `Dcm` derives the used strategy directly from the configured used port interfaces `DcmDspDataUsePort` for the `DID` data elements. In the current version only strategy 1) is supported which means that all configured port interfaces are valid with the exception of [\[SWS_Dcm_CONSTR_06108\]](#).

[SWS_Dcm_CONSTR_06108] Not supported port interfaces for secure coding
[A `DcmDspDid` that is referenced by `DcmSecureCodingDataIdentifier` shall not have a `DcmDspData` with `DcmDspDataUsePort` set to `USE_ECU_SIGNAL` or `USE_BLOCK_ID`.]

The data that is stored during the secure coding process is a sequential chain of stored data identifiers as shown in Figure 7.13. For each data identifier, the 2-byte data identifier number followed by the data record of the data identifier is stored. This is sequentially repeated for each secure data identifier. The signature validation is run over the entire buffer.

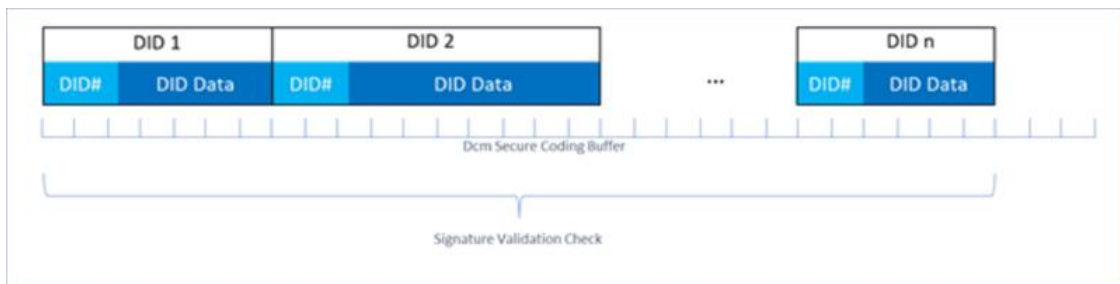


Figure 7.14: Dcm Secure Coding Buffer Layout

The data identifier number in the signature check ensures that the written data matches the intended write operation of the manufacturer and avoids misuse to a data record as data of a different `DID`. The signature validation over the entire buffer only allows one explicit order of `DIDs` in which the data has to be written. Even if the order might be irrelevant for coding data for some users, the order needs to be kept.

[SWS_Dcm_01685] Write Secure Coding DID for started secure coding sequence

Upstream requirements: [RS_Diag_04272](#)

[If the `Dcm` receives a write data by identifier request for a data identifier referenced by `DcmSecureCodingDataIdentifier` and the secure coding sequence has been started according to [\[SWS_Dcm_01684\]](#), the `Dcm` shall sequentially store the data in the buffer defined by `DcmSecureCodingDataBufferSize` and store the 2 byte `DID` number followed by the data in it.]

[SWS_Dcm_CONSTR_06109] Static length of secure coding `DIDs` [A `DcmDspDid` that is referenced by `DcmSecureCodingDataIdentifier` shall have a static length. This means that the `DcmDspData` of that `DID` shall not have any `DcmDspDataType` set to `UINT8_DYN`.]

With [\[SWS_Dcm_01685\]](#) the `Dcm` stores the data in a temporary local buffer that is not visible to the outside. Applications have therefore no access to the data at this

point. This also implies that the configured callouts in `DcmDspDataUsePort` are not yet invoked.

Apart from this local data storage, the Dcm processes a secure coding `DID` diagnostic request in the same way as any other diagnostic request. In particular, all service validation checks are executed.

[SWS_Dcm_01686] Write Secure Coding `DID` without a started secure coding sequence

Upstream requirements: [RS_Diag_04272](#)

[If the Dcm receives a write data by identifier request for a data identifier referenced by `DcmSecureCodingDataIdentifier` and the secure coding sequence has not been started according to [\[SWS_Dcm_01684\]](#), the Dcm shall return an NRC 0x22 (condition-`sNotCorrect`) as a response to the write data by identifier request.]

[SWS_Dcm_01687] Write Secure Coding `DID` buffer overflow

Upstream requirements: [RS_Diag_04272](#)

[If the Dcm is storing secured coding `DID` data within the buffer provided by `DcmSecureCodingDataBufferSize` and the data of the `DID` written to it exceeds the configured size, the Dcm shall send an NRC 0x22 (`conditionsNotCorrect`) and stop the secure coding sequence.]

The secure coding sequence is finalized by a dedicated routine control service. It is responsible for verifying the authenticity of the written data and triggers the final data write operations.

[SWS_Dcm_01688] Secure Coding validation trigger on inactive secure coding sequence

Upstream requirements: [RS_Diag_04272](#)

[If the Dcm receives the routine control service referred by `DcmSecureCodingValidationRoutine` and the secure coding sequence has not been started according to [\[SWS_Dcm_01684\]](#), the Dcm shall return an NRC 0x22 (`conditionsNotCorrect`).]

[SWS_Dcm_01689] Secure Coding validation trigger without written data

Upstream requirements: [RS_Diag_04272](#)

[If the Dcm receives the routine control service referred by `DcmSecureCodingValidationRoutine` and no secure data according to [\[SWS_Dcm_01685\]](#) has been written so far, the Dcm shall return an NRC 0x22 (`conditionsNotCorrect`).]

[SWS_Dcm_CONSTR_06110] Secure Coding validation routine input parameter

[A `DcmDspRoutine` that is referenced by `DcmSecureCodingValidationRoutine` shall only have a single configured `DcmDspStartRoutineInSignal` with `DcmDspRoutineSignalType` set to `UINT8_N`.]

[SWS_Dcm_CONSTR_06111] Secure Coding validation routine output parameter

[A `DcmDspRoutine` that is referenced by `DcmSecureCodingValidationRoutine`

shall only have a single configured `DcmDspStartRoutineOutSignal` with `DcmDspRoutineSignalType` set to `UINT8`.]

A secure coding validation routine control service is processed entirely `Dcm` internal without calling an SWC via C/S calls or a CDD via a C-Function.

[SWS_Dcm_CONSTR_06112] Secure Coding validation routine is synchronous

[A `DcmDspRoutine` that is referenced by `DcmSecureCodingValidationRoutine` shall not have a configured `DcmDspStopRoutine` and `DcmDspRequestRoutineResults`.]

[SWS_Dcm_01690] Dcm internal processing of secure coding validation routine

Upstream requirements: [RS_Diag_04272](#)

[If the `Dcm` receives the routine control service referred by `DcmSecureCodingValidationRoutine`, the `Dcm` shall process the routine control inside the `Dcm` and does neither create nor call the external C-Function interface [[SWS_Dcm_01203](#)].]

[SWS_Dcm_01691] Authentication of secure coding data

Upstream requirements: [RS_Diag_04272](#)

[If the `Dcm` receives the routine control service referred by `DcmSecureCodingValidationRoutine` it shall verify if the signature matches the authentication provided by the secure coding certificate by using the crypto job configured by `DcmSecureCodingAuthenticationJobRef` with a signature obtained by the one and only routine control request parameter and the secure coding data buffer as data to verify.]

[SWS_Dcm_CONSTR_06113] Require asynchronous signature verification [`DcmSecureCodingAuthenticationJobRef` is only valid if the referenced `CsmJob` itself:

- has a `CsmProcessingMode` set to `CRYPTO_PROCESSING_ASYNC`
- references a `CsmPrimitive` with an aggregated `CsmSignatureVerify`.

]

The crypto job from [[SWS_Dcm_01691](#)] securely authenticates the data that has been written into the secure data buffer. This is done by verifying this data with the signature that is provided by the client as a request parameter of the secure coding authentication routine control service with the public key of the certificate. The certificate's public key is mandatory input data for the verification. In the AUTOSAR architecture, the crypto job directly accesses the public key of the certificate. The integrator of the `Csm` and `KeyM` modules links the public key to the crypto job.

A successful authentication of the secure data ensures from a security point of view, that the client is authenticated to write exactly that data to the corresponding data identifiers. After this successful authentication, the `Dcm` will call the same configured write APIs as it would do for write operations without the secure coding procedure. It can be seen as a delay of the write operations to the point that all write data is authenticated.

For the application, it is a write operation and it does not need to distinguish between a normal or secured coding write operation.

[SWS_Dcm_01692] Successful authenticated secure coding data

Upstream requirements: [RS_Diag_04272](#)

[If the secure coding data has been successfully validated according to [\[SWS_Dcm_01691\]](#), the [Dcm](#) shall call the configured write operations for all secured coding data identifiers in the secure data buffer.]

The secure coding sequence can consist of more than one written data identifier. From the [Dcm](#) side, there is no transaction system in place or any kind of rollback if a write operation fails. In case of a failed write operation, the secure coding sequence is canceled and a failure indication is sent as a parameter in the positive response of the secure coding validation routine control service. It is the application's responsibility to design a sufficient stable mechanism to deal with this situation.

[SWS_Dcm_01693] Successfully completed secure coding procedure

Upstream requirements: [RS_Diag_04272](#)

[If all the called write operations from [\[SWS_Dcm_01691\]](#) have been completed successfully, the [Dcm](#) shall set the positive response routine parameter [DcmDspStartRoutineOutSignal](#) of the [DcmSecureCodingValidationRoutine](#) to the value of 0x00 to indicate a successful completion of the secure coding procedure and send a positive response for the [DcmSecureCodingValidationRoutine](#).]

[SWS_Dcm_01694] Failed secure coding write

Upstream requirements: [RS_Diag_04272](#)

[If a called write operation from [\[SWS_Dcm_01691\]](#) has failed, the [Dcm](#) shall cancel all pending other secure coding write operations and set the positive response routine parameter [DcmDspStartRoutineOutSignal](#) of the [DcmSecureCodingValidationRoutine](#) to the value of 0x01 to indicate a failed secure coding write operation and send a positive response for the [DcmSecureCodingValidationRoutine](#).]

[SWS_Dcm_01695] Wrong signature for secure coding data

Upstream requirements: [RS_Diag_04272](#)

[If the crypto job to authenticate the secure data has failed, the [Dcm](#) shall set the positive response routine parameter [DcmDspStartRoutineOutSignal](#) of the [DcmSecureCodingValidationRoutine](#) to the value of 0x02 to indicate a signature validation failure and send a positive response for the [DcmSecureCodingValidationRoutine](#).]

[SWS_Dcm_01696] Behavior on too long signature validation [If the execution time of the routine control behind the [DcmSecureCodingValidationRoutine](#) exceeds the maximum number of allowed response times due to [\[SWS_Dcm_00120\]](#), the [Dcm](#) shall report the Runtime Error DCM_E_INTERFACE_TIMEOUT.]

7.9 Startup behavior

[SWS_Dcm_00334] [Dcm_Init shall initialize all Dcm global variables with the values of the configuration]

7.10 Error Classification

This section describes how the Dcm module has to treat the several error classes that may happen during the life cycle of the Dcm module.

Diagnostic-Communication-Errors are handled directly in the ISO-Protocols by NRCs.

[SWS_Dcm_00044]

Upstream requirements: SRS_BSW_00369

[The error values shall be the unique for all error types. The Dcm shall use only the values given in this chapter.]

Chapter [8, General Specification of Basic Software Modules] 7.2 “Error Handling” describes the error handling of the Basic Software in detail. Above all, it constitutes a classification scheme consisting of five error types which may occur in BSW modules.

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

7.10.1 Development Errors

The errors and exceptions described in [SWS_Dcm_00040] shall be detectable by the Dcm module depending on its build version (development/production mode).

[SWS_Dcm_00040] Definition of development errors in module Dcm

Upstream requirements: SRS_BSW_00337

[

Type of error	Related error code	Error value
The Dcm is getting called with an invalid input parameter value or the Dcm has called an function and this function returns an invalid out parameter or return value.	DCM_E_INVALID_VALUE	0x02
Interface return-value is out of range	DCM_E_INTERFACE_RETURN_VALUE	0x02
Internal:Dcm not initialized	DCM_E_UNINIT	0x05
Dcm API function with invalid input parameter	DCM_E_PARAM	0x06
Dcm API service invoked with NULL POINTER as parameter	DCM_E_PARAM_POINTER	0x07
Dcm initialisation failed	DCM_E_INIT_FAILED	0x08





Type of error	Related error code	Error value
Storing the ProgConditions failed	DCM_E_SET_PROG_CONDITIONS_FAIL	0x09

]

7.10.2 Runtime Errors

The errors and exceptions described in [SWS_Dcm_01416] shall be detectable by the Dcm module depending on its build version (development/production mode).

[SWS_Dcm_01416] Definition of runtime errors in module Dcm

Upstream requirements: [SRS_BSW_00452](#)

[

Type of error	Related error code	Error value
Interface: Timeout occurred during interaction with another module (e.g. maximum number of response pending is reached, refer to [SWS_Dcm_00120])	DCM_E_INTERFACE_TIMEOUT	0x01
Interface: Boundary check of buffers provided by the Dcm failed during interaction with another module (application, Dem, PduR, etc.)	DCM_E_INTERFACE_BUFFER_OVERFLOW	0x03

]

7.10.3 Production Errors

There are no production errors.

7.10.4 Extended Production Errors

There are no extended production errors.

7.11 Security Events

[SWS_Dcm_01589]

Upstream requirements: [RS_Ids_00810](#)

[If security event reporting has been enabled for the Dcm module ([DcmEnableSecurityEventReporting](#) = true) the respective security events shall be reported to the IdsM via the interfaces defined in AUTOSAR_SWS_BSWGeneral.]

[SWS_Dcm_01590] Security events for Dcm

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

Name	Description	ID
SEV_UDS_SECURITY_ACCESS_NEEDED	Tester has sent a diagnostic request without meeting the server's security level requirements for that service. NRC 0x33 (securityAccessDenied) was returned.	100
SEV_UDS_AUTHENTICATION_NEEDED	A diagnostic request was received while the required authentication to execute this service is not given. NRC 0x34 (authenticationRequired) was returned.	101
SEV_UDS_SECURITY_ACCESS_SUCCESSFUL	Successful unlocked the ECU (via Security Access SID 0x27)	102
SEV_UDS_SECURITY_ACCESS_FAILED	Unlocking of the ECU (via Security Access SID 0x27) failed	103
SEV_UDS_AUTHENTICATION_SUCCESSFUL	Successfully authenticated (via Authentication SID 0x29)	104
SEV_UDS_AUTHENTICATION_FAILED	Authentication (via Authentication SID 0x29) failed	105
SEV_UDS_WRITE_DATA_SUCCESSFUL	Diagnostic data identifier has been written by SID 0x2E WriteDataByIdentifier	106
SEV_UDS_WRITE_DATA_FAILED	Change of Diagnostic data identifier has been requested by SID 0x2E WriteDataByIdentifier, but failed	107
SEV_UDS_WRITE_MEMORY_SUCCESSFUL	Data has been written into memory by SID 0x3D WriteMemoryByAddress	108
SEV_UDS_WRITE_MEMORY_FAILED	Writing of Data into memory has been requested by SID 0x3D WriteMemoryByAddress, but failed	109
SEV_UDS_REQUEST_UP_DOWNLOAD_SUCCESSFUL	An upload / download sequence has been requested successfully with SID 0x34 or SID 0x35	110
SEV_UDS_REQUEST_UP_DOWNLOAD_FAILED	An upload / download sequence has been requested with SID 0x34 or SID 0x35, but failed	111
SEV_UDS_REQUEST_FILE_TRANSFER_SUCCESSFUL	A file transfer sequence has been requested successfully with SID 0x38.	112
SEV_UDS_REQUEST_FILE_TRANSFER_FAILED	A file transfer sequence has been requested with SID 0x38, but failed	113
SEV_UDS_COMMUNICATION_CONTROL_SUCCESSFUL	The control of a communication has been requested by service SID 0x28 CommunicationControl successfully.	114
SEV_UDS_COMMUNICATION_CONTROL_FAILED	The control of a communication has been requested by service SID 0x28 CommunicationControl, but failed.	115
SEV_UDS_CLEAR_DTC_SUCCESSFUL	DTC information has been cleared by SID 0x14 ClearDiagnosticInformation.	116
SEV_UDS_CLEAR_DTC_FAILED	Clearing DTC information has been requested by SID 0x14 ClearDiagnosticInformation, but failed.	117
SEV_UDS_CONTROL_DTC_SETTING_SUCCESSFUL	The control of a DTC setting has been requested by service SID 0x85 ControlDTCSetting successfully.	118
SEV_UDS_CONTROL_DTC_SETTING_FAILED	Control of DTC setting has been requested by service SID 0x85 ControlDTCSetting, but failed.	119
SEV_UDS_ECU_RESET_SUCCESSFUL	ECU has been reset by SID 0x11 ECUReset.	120
SEV_UDS_ECU_RESET_FAILED	ECU Reset has been requested by SID 0x11 ECUReset, but failed.	121
SEV_UDS_ROUTINE_CONTROL_SUCCESSFUL	The control of a routine has been requested by service SID 0x31 RoutineControl successfully.	122
SEV_UDS_ROUTINE_CONTROL_FAILED	The control of a routine has been requested by service SID 0x31 RoutineControl, but failed.	123
SEV_UDS_IO_CONTROL_SUCCESSFUL	IOControl operation has been requested by service SID 0x2F InputOutputControlByIdentifier successfully.	124





Name	Description	ID
SEV_UDS_IO_CONTROL_FAILED	IOControl operation has been requested by service SID 0x2F InputOutputControlByIdentifier, but failed.	125

]

In situations when a single diagnostic request can result in more than one security event, the [Dcm](#) can report more than one security event.

Example: The diagnostic service 0x2E (WriteDataByIdentifier) as rejected with NRC 0x33 (SecurityAccessDenied) and the [Dcm](#) is configured to report SEV_UDS_SECURITY_ACCESS_NEEDED and SEV_UDS_WRITE_DATA_FAILED. In this case the two security events SEV_UDS_SECURITY_ACCESS_NEEDED and SEV_UDS_WRITE_DATA_FAILED are reported.

[SWS_Dcm_01702] Reporting security access denied to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service request is not allowed in the current security level, which results in a negative response with NRC 0x33 (securityAccessDenied), the Dcm shall report a security event SEV_UDS_SECURITY_ACCESS_NEEDED to IdsM (see [\[11\]](#) and table [\[SWS_Dcm_01703\]](#)) with the context data given in [\[SWS_Dcm_01703\]](#).]

[SWS_Dcm_01703] Security event context data definition: SEV_UDS_SECURITY_ACCESS_NEEDED

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_SECURITY_ACCESS_NEEDED	
ID	100	
Description	Tester has sent a diagnostic request without meeting the server's security level requirements for that service. NRC 0x33 (securityAccessDenied) was returned.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
SID	uint8	
Subfunction	uint8	255: is filled in case the service is without Subfunction
DataIdentifier	uint16	65535: is filled in case the service is without DID
RoutineIdentifier	uint16	65535: is filled in case the service is without RID
ClientSourceAddress	uint16	

]

[SWS_Dcm_01704] Reporting authentication required to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service request is not having the required authentication which results in a negative response with NRC 0x34 (authenticationRequired), the Dcm shall report a security event SEV_UDS_AUTHENTICATION_NEEDED to IdsM (see [\[11\]](#) and table [\[SWS_Dcm_01705\]](#)) with the context data given in [\[SWS_Dcm_01705\]](#).]

[SWS_Dcm_01705] Security event context data definition: SEV_UDS_AUTHENTICATION_NEEDED*Status:* DRAFT*Upstream requirements:* [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_AUTHENTICATION_NEEDED	
ID	101	
Description	A diagnostic request was received while the required authentication to execute this service is not given. NRC 0x34 (authenticationRequired) was returned.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
SID	uint8	
Subfunction	uint8	255: is filled in case the service is without Subfunction
DataIdentifier	uint16	65535: is filled in case the service is without DID
RoutineIdentifier	uint16	65535: is filled in case the service is without RID
ClientSourceAddress	uint16	

]

[SWS_Dcm_01706] Reporting successful security access to IdsM*Upstream requirements:* [RS_Ids_00810](#)

[In case a diagnostic service SecurityAccess with subfunction CompareKey is received, which successfully unlocks the requested security access type, the Dcm shall report a security event SEV_UDS_SECURITY_ACCESS_SUCCESSFUL to IdsM (see [11] and table [SWS_Dcm_01707]) with the context data given in [SWS_Dcm_01707].]

[SWS_Dcm_01707] Security event context data definition: SEV_UDS_SECURITY_ACCESS_SUCCESSFUL*Status:* DRAFT*Upstream requirements:* [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_SECURITY_ACCESS_SUCCESSFUL	
ID	102	
Description	Successful unlocked the ECU (via Security Access SID 0x27)	
Context Data Version	1	
Context Data	Data Type	Allowed Values
Subfunction	uint8	
ClientSourceAddress	uint16	

]

[SWS_Dcm_01708] Reporting failed security access to IdsM*Upstream requirements:* [RS_Ids_00810](#)

[In case a diagnostic service SecurityAccess is received, which results in a negative response, the Dcm shall report a security event SEV_UDS_SECURITY_ACCESS_FAILED to IdsM (see [11] and table [SWS_Dcm_01709]) with the context data given in [SWS_Dcm_01709].]

[SWS_Dcm_01709] Security event context data definition: SEV_UDS_SECURITY_ACCESS_FAILED*Status:* DRAFT*Upstream requirements:* [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_SECURITY_ACCESS_FAILED	
ID	103	
Description	Unlocking of the ECU (via Security Access SID 0x27) failed	
Context Data Version	1	
Context Data	Data Type	Allowed Values
Subfunction	uint8	
ClientSourceAddress	uint16	
NegativeResponseCode	uint8	

]

[SWS_Dcm_01710] Reporting successful authentication to IdsM*Upstream requirements:* [RS_Ids_00810](#)

[In case a diagnostic service Authentication with subfunction proofOfOwnership is received, which is successful, the Dcm shall report a security event SEV_UDS_AUTHENTICATION_SUCCESSFUL to IdsM (see [11] and table [SWS_Dcm_01711]) with the context data given in [SWS_Dcm_01711].]

[SWS_Dcm_01711] Security event context data definition: SEV_UDS_AUTHENTICATION_SUCCESSFUL*Status:* DRAFT*Upstream requirements:* [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_AUTHENTICATION_SUCCESSFUL	
ID	104	
Description	Successfully authenticated (via Authentication SID 0x29)	
Context Data Version	1	
Context Data	Data Type	Allowed Values
Subfunction	uint8	
ClientSourceAddress	uint16	

]

[SWS_Dcm_01712] Reporting failed authentication to IdsM*Upstream requirements:* [RS_Ids_00810](#)

[In case a diagnostic service Authentication is received, which results in a negative response, the Dcm shall report a security event SEV_UDS_AUTHENTICATION_FAILED to IdsM (see [11] and table [SWS_Dcm_01713]) with the context data given in [SWS_Dcm_01713].]

[SWS_Dcm_01713] Security event context data definition: SEV_UDS_AUTHENTICATION_FAILED

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_AUTHENTICATION_FAILED	
ID	105	
Description	Authentication (via Authentication SID 0x29) failed	
Context Data Version	1	
Context Data	Data Type	Allowed Values
Subfunction	uint8	
ClientSourceAddress	uint16	
NegativeResponseCode	uint8	

]

[SWS_Dcm_01714] Reporting successful writing of data to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service WriteDataByIdentifier is received, which results in a positive response, the Dcm shall report a security event SEV_UDS_WRITE_DATA_SUCCESSFUL to IdsM (see [11] and table [SWS_Dcm_01715]) with the context data given in [SWS_Dcm_01715].]

[SWS_Dcm_01715] Security event context data definition: SEV_UDS_WRITE_DATA_SUCCESSFUL

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_WRITE_DATA_SUCCESSFUL	
ID	106	
Description	Diagnostic data identifier has been written by SID 0x2E WriteDataByIdentifier	
Context Data Version	1	
Context Data	Data Type	Allowed Values
DID	uint16	
ClientSourceAddress	uint16	

]

[SWS_Dcm_01716] Reporting writing of data failed to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service WriteDataByIdentifier is received, which results in a negative response, the Dcm shall report a security event SEV_UDS_WRITE_DATA_FAILED to IdsM (see [11] and table [SWS_Dcm_01717]) with the context data given in [SWS_Dcm_01717].]

[SWS_Dcm_01717] Security event context data definition: SEV_UDS_WRITE_DATA_FAILED

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_WRITE_DATA_FAILED	
ID	107	
Description	Change of Diagnostic data identifier has been requested by SID 0x2E WriteDataByIdentifier, but failed	
Context Data Version	1	
Context Data	Data Type	Allowed Values
DID	uint16	
ClientSourceAddress	uint16	
NegativeResponseCode	uint8	

]

[SWS_Dcm_01718] Reporting successful up download to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service RequestDownload or RequestUpload is received, which results in a positive response, the Dcm shall report a security event SEV_UDS_REQUEST_UP_DOWNLOAD_SUCCESSFUL to IdsM (see [11] and table [SWS_Dcm_01719]) with the context data given in [SWS_Dcm_01719].]

[SWS_Dcm_01719] Security event context data definition: SEV_UDS_REQUEST_UP_DOWNLOAD_SUCCESSFUL

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_REQUEST_UP_DOWNLOAD_SUCCESSFUL	
ID	110	
Description	An upload / download sequence has been requested successfully with SID 0x34 or SID 0x35	
Context Data Version	1	
Context Data	Data Type	Allowed Values
SID	uint8	
MemoryAddress	uint32	
MemorySize	uint32	
ClientSourceAddress	uint16	

]

[SWS_Dcm_01720] Reporting up download failed to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service RequestDownload or RequestUpload is received, which results in a negative response, the Dcm shall report a security event SEV_UDS_REQUEST_UP_DOWNLOAD_FAILED to IdsM (see [11] and table [SWS_Dcm_01719]) with the context data given in [SWS_Dcm_01719].]

[SWS_Dcm_01721] Security event context data definition: SEV_UDS_REQUEST_UP_DOWNLOAD_FAILED*Status:* DRAFT*Upstream requirements:* [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_REQUEST_UP_DOWNLOAD_FAILED	
ID	111	
Description	An upload / download sequence has been requested with SID 0x34 or SID 0x35, but failed	
Context Data Version	1	
Context Data	Data Type	Allowed Values
SID	uint8	
MemoryAddress	uint32	
MemorySize	uint32	
ClientSourceAddress	uint16	
NegativeResponseCode	uint8	

]

[SWS_Dcm_01722] Reporting successful file transfer to IdsM*Upstream requirements:* [RS_Ids_00810](#)

[In case a diagnostic service RequestFileTransfer is received, which results in a positive response, the Dcm shall report a security event SEV_UDS_REQUEST_FILE_TRANSFER_SUCCESSFUL to IdsM (see [11] and table [SWS_Dcm_01723]) with the context data given in [SWS_Dcm_01723].]

[SWS_Dcm_01723] Security event context data definition: SEV_UDS_REQUEST_FILE_TRANSFER_SUCCESSFUL*Status:* DRAFT*Upstream requirements:* [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_REQUEST_FILE_TRANSFER_SUCCESSFUL	
ID	112	
Description	A file transfer sequence has been requested successfully with SID 0x38.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
ModeOfOperation	uint8	AddFile (0x01) DeleteFile (0x02) ReplaceFile (0x03) ReadFile (0x04) ReadDir (0x05) ResumeFile (0x06)
FilePathAndName	uint8 [50]	Each byte of this parameter is encoded in ASCII format.
ClientSourceAddress	uint16	

]

[SWS_Dcm_01724] Reporting file transfer failed to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service RequestFileTransfer is received, which results in a negative response, the Dcm shall report a security event SEV_UDS_REQUEST_FILE_TRANSFER_FAILED to IdsM (see [11] and table [SWS_Dcm_01725]) with the context data given in [SWS_Dcm_01725].]

[SWS_Dcm_01725] Security event context data definition: SEV_UDS_REQUEST_FILE_TRANSFER_FAILED

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

SEV Name	SEV_UDS_REQUEST_FILE_TRANSFER_FAILED	
ID	113	
Description	A file transfer sequence has been requested with SID 0x38, but failed	
Context Data Version	1	
Context Data	Data Type	Allowed Values
ModeOfOperation	uint8	AddFile (0x01) DeleteFile (0x02) ReplaceFile (0x03) ReadFile (0x04) ReadDir (0x05) ResumeFile (0x06)
FilePathAndName	uint8 [50]	Each byte of this parameter is encoded in ASCII format.
ClientSourceAddress	uint16	
NegativeResponseCode	uint8	

[SWS_Dcm_01726] Reporting successful communication control to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service CommunicationControl is received, which results in a positive response, the Dcm shall report a security event SEV_UDS_COMMUNICATION_CONTROL_SUCCESSFUL to IdsM (see [11] and table [SWS_Dcm_01727]) with the context data given in [SWS_Dcm_01727].]

[SWS_Dcm_01727] Security event context data definition: SEV_UDS_COMMUNICATION_CONTROL_SUCCESSFUL

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

SEV Name	SEV_UDS_COMMUNICATION_CONTROL_SUCCESSFUL	
ID	114	
Description	The control of a communication has been requested by service SID 0x28 Communication Control successfully.	
Context Data Version	1	





SEV Name	SEV_UDS_COMMUNICATION_CONTROL_SUCCESSFUL	
Context Data	Data Type	Allowed Values
Subfunction	uint8	
ClientSourceAddress	uint16	

]

[SWS_Dcm_01728] Reporting communication control failed to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service CommunicationControl is received, which results in a negative response, the Dcm shall report a security event SEV_UDS_COMMUNICATION_CONTROL_FAILED to IdsM (see [11] and table [SWS_Dcm_01729]) with the context data given in [SWS_Dcm_01729].]

[SWS_Dcm_01729] Security event context data definition: SEV_UDS_COMMUNICATION_CONTROL_FAILED

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_COMMUNICATION_CONTROL_FAILED	
ID	115	
Description	The control of a communication has been requested by service SID 0x28 Communication Control, but failed.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
Subfunction	uint8	
ClientSourceAddress	uint16	
NegativeResponseCode	uint8	

]

[SWS_Dcm_01730] Reporting successful clearing of dtc to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service ClearDiagnosticInformation is received, which results in a positive response, the Dcm shall report a security event SEV_UDS_CLEAR_DTC_SUCCESSFUL to IdsM (see [11] and table [SWS_Dcm_01731]) with the context data given in [SWS_Dcm_01731].]

[SWS_Dcm_01731] Security event context data definition: SEV_UDS_CLEAR_DTC_SUCCESSFUL

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_CLEAR_DTC_SUCCESSFUL	
ID	116	
Description	DTC information has been cleared by SID 0x14 ClearDiagnosticInformation.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
GroupOfDTC	uint8 [3]	given in the format: HighByte, MiddleByte, LowByte
MemorySelection	uint16	0x0001: PrimaryMemory 0x01XX: XX is the address of the UserDefinedMemory
ClientSourceAddress	uint16	

]

[SWS_Dcm_01732] Reporting clearing of dtc failed to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service ClearDiagnosticInformation is received, which results in a negative response, the Dcm shall report a security event SEV_UDS_CLEAR_DTC_FAILED to IdsM (see [11] and table [SWS_Dcm_01733]) with the context data given in [SWS_Dcm_01733].]

[SWS_Dcm_01733] Security event context data definition: SEV_UDS_CLEAR_DTC_FAILED

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_CLEAR_DTC_FAILED	
ID	117	
Description	Clearing DTC information has been requested by SID 0x14 ClearDiagnosticInformation, but failed.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
GroupOfDTC	uint8 [3]	given in the format: HighByte, MiddleByte, LowByte
MemorySelection	uint16	0x0001: PrimaryMemory 0x01XX: XX is the address of the UserDefinedMemory
ClientSourceAddress	uint16	
NegativeResponseCode	uint8	

]

[SWS_Dcm_01734] Reporting successful control of dtc to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service ControlDTCSetting is received, which results in a positive response, the Dcm shall report a security event

SEV_UDS_CONTROL_DTC_SETTING_SUCCESSFUL to IdsM (see [11] and table [SWS_Dcm_01735]) with the context data given in [SWS_Dcm_01735].]

[SWS_Dcm_01735] Security event context data definition: SEV_UDS_CONTROL_DTC_SETTING_SUCCESSFUL

Status: DRAFT

Upstream requirements: RS_Ids_00810

[

SEV Name	SEV_UDS_CONTROL_DTC_SETTING_SUCCESSFUL	
ID	118	
Description	The control of a DTC setting has been requested by service SID 0x85 ControlDTCSetting successfully.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
Subfunction	uint8	
ClientSourceAddress	uint16	

]

[SWS_Dcm_01736] Reporting control of dtc failed to IdsM

Upstream requirements: RS_Ids_00810

[In case a diagnostic service ControlDTCSetting is received, which results in a negative response, the Dcm shall report a security event SEV_UDS_CONTROL_DTC_SETTING_FAILED to IdsM (see [11] and table [SWS_Dcm_01737]) with the context data given in [SWS_Dcm_01737].]

[SWS_Dcm_01737] Security event context data definition: SEV_UDS_CONTROL_DTC_SETTING_FAILED

Status: DRAFT

Upstream requirements: RS_Ids_00810

[

SEV Name	SEV_UDS_CONTROL_DTC_SETTING_FAILED	
ID	119	
Description	Control of DTC setting has been requested by service SID 0x85 ControlDTCSetting, but failed.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
Subfunction	uint8	
ClientSourceAddress	uint16	
NegativeResponseCode	uint8	

]

[SWS_Dcm_01738] Reporting successful reset of ECU to IdsM

Upstream requirements: RS_Ids_00810

[In case a diagnostic service ECUReset is received, which results in a positive response, the Dcm shall report a security event

SEV_UDS_ECU_RESET_SUCCESSFUL to IdsM (see [11] and table [SWS_Dcm_01739]) with the context data given in [SWS_Dcm_01739].]

[SWS_Dcm_01739] Security event context data definition: SEV_UDS_ECU_RESET_SUCCESSFUL

Status: DRAFT

Upstream requirements: RS_Ids_00810

[

SEV Name	SEV_UDS_ECU_RESET_SUCCESSFUL	
ID	120	
Description	ECU has been reset by SID 0x11 ECUReset.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
Subfunction	uint8	
ClientSourceAddress	uint16	

]

[SWS_Dcm_01740] Reporting reset of ECU failed to IdsM

Upstream requirements: RS_Ids_00810

[In case a diagnostic service ECUReset is received, which results in a negative response, the Dcm shall report a security event SEV_UDS_ECU_RESET_FAILED to IdsM (see [11] and table [SWS_Dcm_01741]) with the context data given in [SWS_Dcm_01741].]

[SWS_Dcm_01741] Security event context data definition: SEV_UDS_ECU_RESET_FAILED

Status: DRAFT

Upstream requirements: RS_Ids_00810

[

SEV Name	SEV_UDS_ECU_RESET_FAILED	
ID	121	
Description	ECU Reset has been requested by SID 0x11 ECUReset, but failed.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
Subfunction	uint8	
ClientSourceAddress	uint16	
NegativeResponseCode	uint8	

]

[SWS_Dcm_01742] Reporting successful routine control to IdsM

Upstream requirements: RS_Ids_00810

[In case a diagnostic service RoutineControl is received, which results in a positive response, the Dcm shall report a security event SEV_UDS_ROUTINE_CONTROL_SUCCESSFUL to IdsM (see [11] and table [SWS_Dcm_01743]) with the context data given in [SWS_Dcm_01743].]

[SWS_Dcm_01743] Security event context data definition: SEV_UDS_ROUTINE_CONTROL_SUCCESSFUL

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_ROUTINE_CONTROL_SUCCESSFUL	
ID	122	
Description	The control of a routine has been requested by service SID 0x31 RoutineControl successfully.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
RID	uint16	
Subfunction	uint8	
ClientSourceAddress	uint16	

]

[SWS_Dcm_01744] Reporting routine control failed to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service RoutineControl is received, which results in a negative response, the Dcm shall report a security event SEV_UDS_ROUTINE_CONTROL_FAILED to IdsM (see [11] and table [SWS_Dcm_01745]) with the context data given in [SWS_Dcm_01745].]

[SWS_Dcm_01745] Security event context data definition: SEV_UDS_ROUTINE_CONTROL_FAILED

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_ROUTINE_CONTROL_FAILED	
ID	123	
Description	The control of a routine has been requested by service SID 0x31 RoutineControl, but failed.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
RID	uint16	
Subfunction	uint8	
ClientSourceAddress	uint16	
NegativeResponseCode	uint8	

]

[SWS_Dcm_01746] Reporting successful IO control to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service IOControl is received, which results in a positive response, the Dcm shall report a security event SEV_UDS_IO_CONTROL_SUCCESSFUL to IdsM (see [11] and table [SWS_Dcm_01747]) with the context data given in [SWS_Dcm_01747].]

[SWS_Dcm_01747] Security event context data definition: SEV_UDS_IO_CONTROL_SUCCESSFUL*Status:* DRAFT*Upstream requirements:* [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_IO_CONTROL_SUCCESSFUL	
ID	124	
Description	IOControl operation has been requested by service SID 0x2F InputOutputControlBy Identifier successfully.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
DID	uint16	
inputOutputControlParameter	uint8	
ClientSourceAddress	uint16	

]

[SWS_Dcm_01748] Reporting IO control failed to IdsM*Upstream requirements:* [RS_Ids_00810](#)

[In case a diagnostic service IOControl is received, which results in a negative response, the Dcm shall report a security event SEV_UDS_IO_CONTROL_FAILED to IdsM (see [11] and table [SWS_Dcm_01749]) with the context data given in [SWS_Dcm_01749].]

[SWS_Dcm_01749] Security event context data definition: SEV_UDS_IO_CONTROL_FAILED*Status:* DRAFT*Upstream requirements:* [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_IO_CONTROL_FAILED	
ID	125	
Description	IOControl operation has been requested by service SID 0x2F InputOutputControlBy Identifier, but failed.	
Context Data Version	1	
Context Data	Data Type	Allowed Values
DID	uint16	
inputOutputControlParameter	uint8	
ClientSourceAddress	uint16	
NegativeResponseCode	uint8	

]

[SWS_Dcm_01750] Reporting successful Memory Write operation to IdsM*Upstream requirements:* [RS_Ids_00810](#)

[In case a diagnostic service WriteMemoryByAddress is received, which results in a positive response, the Dcm shall report a security event SEV_UDS_WRITE_MEMORY_SUCCESSFUL to IdsM (see [11] and table [SWS_Dcm_01751]) with the context data given in [SWS_Dcm_01751].]

[SWS_Dcm_01751] Security event context data definition: SEV_UDS_WRITE_MEMORY_SUCCESSFUL

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_WRITE_MEMORY_SUCCESSFUL	
ID	108	
Description	Data has been written into memory by SID 0x3D WriteMemoryByAddress	
Context Data Version	1	
Context Data	Data Type	Allowed Values
MemoryAddress	uint32	
MemorySize	uint32	
ClientSourceAddress	uint16	

]

[SWS_Dcm_01752] Reporting Write Memory failed to IdsM

Upstream requirements: [RS_Ids_00810](#)

[In case a diagnostic service WriteMemoryByAddress is received, which results in a negative response, the Dcm shall report a security event SEV_UDS_WRITE_MEMORY_FAILED to IdsM (see [11] and table [SWS_Dcm_01753]) with the context data given in [SWS_Dcm_01753].]

[SWS_Dcm_01753] Security event context data definition: SEV_UDS_WRITE_MEMORY_FAILED

Status: DRAFT

Upstream requirements: [RS_Ids_00810](#)

[

SEV Name	SEV_UDS_WRITE_MEMORY_FAILED	
ID	109	
Description	Writing of Data into memory has been requested by SID 0x3D WriteMemoryByAddress, but failed	
Context Data Version	1	
Context Data	Data Type	Allowed Values
MemoryAddress	uint32	
MemorySize	uint32	
ClientSourceAddress	uint16	
NegativeResponseCode	uint8	

]

8 API specification

This section defines:

- The syntax and semantics of the functions that are provided and required from other BSW modules. These take the form of "C"-APIs.
- The syntax and semantics of a subset of those functions which are used by software-components through the RTE. These take the form of descriptions using the concepts of the Software-Component Template.

8.1 Imported types

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

[SWS_Dcm_00333] Definition of imported datatypes of module Dcm

Module	Header File	Imported Type
BndM	bndm.h	BndM_BlockIdType
	bndm.h	BndM_ResultType
	bndm_externals.h	BndM_Block{BlockId.Shortname}Type
Comtype	ComStack_Types.h	BufReq_ReturnType
	ComStack_Types.h	NetworkHandleType
	ComStack_Types.h	PduIdType
	ComStack_Types.h	PduInfoType
	ComStack_Types.h	PduLengthType
	ComStack_Types.h	RetryInfoType
	ComStack_Types.h	TpDataStateType
Csm	Rte_Csm_Type.h	Crypto_OperationModeType
	Rte_Csm_Type.h	Crypto_ResultType
	Rte_Csm_Type.h	Crypto_VerifyResultType
Dem	Dem.h	Dem_DTCRequestType
	Dem.h	Dem_DTCSeverityType
	Dem.h	Dem_DTCTranslationFormatType
	Rte_Dem_Type.h	Dem_DTCFormatType
	Rte_Dem_Type.h	Dem_DTCOriginType
	Rte_Dem_Type.h	Dem_UdsStatusByteType
IdsM	IdsM.h	IdsM_SecurityEventIdType
	Rte_IdsM_Type.h	IdsM_TimestampDataType
KeyM	KeyM.h	KeyM_CertDataType
	KeyM.h	KeyM_CertElementIteratorType
	Rte_KeyM_Type.h	KeyM_CertElementIdType
	Rte_KeyM_Type.h	KeyM_CertificateIdType





Module	Header File	Imported Type
	Rte_KeyM_Type.h	KeyM_CertificateStatusType
NvM	Rte_NvM_Type.h	NvM_BlockIdType
Std	Std_Types.h	Std_ReturnType
	Std_Types.h	Std_VersionInfoType

]

8.2 Type definitions

The [Dcm](#) module shall ensure that implementation-specific types are not "visible" outside of Dcm. Otherwise, the complete architecture would be corrupted.

This section lists the types which are defined by the [Dcm](#) SWS.

8.2.1 Dcm_CommunicationModeType

[SWS_Dcm_00981] Definition of datatype **Dcm_CommunicationModeType** [

Name	Dcm_CommunicationModeType		
Kind	Type		
Derived from	uint8		
Range	DCM_ENABLE_RX_TX_NORM	0x00	Enable the Rx and Tx for normal communication
	DCM_ENABLE_RX_DISABLE_TX_NORM	0x01	Enable the Rx and disable the Tx for normal communication
	DCM_DISABLE_RX_ENABLE_TX_NORM	0x02	Disable the Rx and enable the Tx for normal communication
	DCM_DISABLE_RX_TX_NORMAL	0x03	Disable Rx and Tx for normal communication
	DCM_ENABLE_RX_TX_NM	0x04	Enable the Rx and Tx for network management communication
	DCM_ENABLE_RX_DISABLE_TX_NM	0x05	Enable Rx and disable the Tx for network management communication
	DCM_DISABLE_RX_ENABLE_TX_NM	0x06	Disable the Rx and enable the Tx for network management communication
	DCM_DISABLE_RX_TX_NM	0x07	Disable Rx and Tx for network management communication
	DCM_ENABLE_RX_TX_NORM_NM	0x08	Enable Rx and Tx for normal and network management communication
	DCM_ENABLE_RX_DISABLE_TX_NORM_NM	0x09	Enable the Rx and disable the Tx for normal and network management communication





	DCM_DISABLE_RX_ENABLE_TX_NORM_NM	0x0A	Disable the Rx and enable the Tx for normal and network management communication
	DCM_DISABLE_RX_TX_NORM_NM	0x0B	Disable Rx and Tx for normal and network management communication
Description	–		
Available via	Dcm.h		

]

8.2.2 Dcm_ConfigType

[SWS_Dcm_00982] Definition of datatype Dcm_ConfigType [

Name	Dcm_ConfigType	
Kind	Structure	
Elements	Implementation specific	
	Type	–
	Comment	–
Description	This type defines a data structure for the post build parameters of the DCM . At initialization the DCM gets a pointer to a structure of this type to get access to its configuration data, which is necessary for initialization.	
Available via	Dcm.h	

]

8.2.3 Dcm_ReturnReadMemoryType

[SWS_Dcm_00985] Definition of datatype Dcm_ReturnReadMemoryType [

Name	Dcm_ReturnReadMemoryType		
Kind	Type		
Derived from	uint8		
Range	DCM_READ_OK	0x00	Reading has been done
	DCM_READ_PENDING	0x01	Reading is pending, another call is request to finalize the reading
	DCM_READ_FAILED	0x02	Reading has failed
	DCM_READ_FORCE_RCRRP	0x03	Reading is pending, the Response pending transmission starts immediately
Description	Return values of Callout Dcm_ReadMemory		
Available via	Dcm.h		

]

8.2.4 Dcm_ReturnWriteMemoryType

[SWS_Dcm_00986] Definition of datatype Dcm_ReturnWriteMemoryType [

Name	Dcm_ReturnWriteMemoryType		
Kind	Type		
Derived from	uint8		
Range	DCM_WRITE_OK	0x00	Writing has been done
	DCM_WRITE_PENDING	0x01	Writing is pending, another called is requested
	DCM_WRITE_FAILED	0x02	The writing has failed
	DCM_WRITE_FORCE_RCRP	0x03	Writing is pending, the Response pending transmission starts immediately
Description	Return type of callout Dcm_WriteMemory		
Available via	Dcm.h		

8.2.5 Dcm_EcuStartModeType

[SWS_Dcm_00987] Definition of datatype Dcm_EcuStartModeType [

Name	Dcm_EcuStartModeType		
Kind	Type		
Derived from	uint8		
Range	DCM_COLD_START	0x00	The ECU starts normally
	DCM_WARM_START	0x01	The ECU starts from a bootloader jump
Description	Allows the DCM to know if a diagnostic response shall be sent in the case of a jump from bootloader		
Available via	Dcm.h		

8.2.6 Dcm_ProgConditionsType

[SWS_Dcm_00988] Definition of datatype Dcm_ProgConditionsType [

Name	Dcm_ProgConditionsType		
Kind	Structure		
Elements	ConnectionId		
	Type	uint16	
	Comment	Unique id of the connection on which the request has been received	
	TesterAddress		
	Type	uint16	





	Comment	Source address of the received request if meta data is enabled, otherwise the value as configured in DcmDslProtocolRxTesterSourceAddr
	Sid	
	Type	uint8
	Comment	Service identifier of the received request
	SubFnclId	
	Type	uint8
	Comment	Identifier of the received subfunction
	ReprogrammingRequest	
	Type	boolean
	Comment	Set to true in order to request reprogramming of the ECU.
	ApplUpdated	
	Type	boolean
	Comment	Indicate whether the application has been updated or not.
	ResponseRequired	
	Type	boolean
	Comment	Set to true in case the flashloader or application shall send a response.
Description	Used in Dcm_SetProgConditions() to allow the integrator to store relevant information prior to jumping to bootloader / jump due to ECUReset request.	
Available via	Dcm.h	

]

8.2.7 Dcm_MsgItemtype

[SWS_Dcm_00989] Definition of datatype Dcm_MsgItemtype [

Name	Dcm_MsgItemtype
Kind	Type
Derived from	uint8
Description	Base type for diagnostic message item
Available via	Dcm.h

]

8.2.8 Dcm_MsgType

[SWS_Dcm_00990] Definition of datatype Dcm_MsgType [

Name	Dcm_MsgType
Kind	Pointer
Type	Dcm_MsgItemtype *
Description	Base type for diagnostic message (request, positive or negative response)





Available via	Dcm.h

]

8.2.9 Dcm_MsgLenType

[SWS_Dcm_00991] Definition of datatype Dcm_MsgLenType [

Name	Dcm_MsgLenType
Kind	Type
Derived from	uint32
Description	Length of diagnostic message (request, positive or negative response). The maximum length is dependent of the underlying transport protocol/media.
Available via	Dcm.h

]

8.2.10 Dcm_MsgAddInfoType

Please note that the following table describes a struct type definition - including its struct items "elements".

[SWS_Dcm_00992] Definition of datatype Dcm_MsgAddInfoType [

Name	Dcm_MsgAddInfoType	
Kind	Structure	
Elements	reqType	
	Type	bit
	Comment	(Pos LSB+0) 0 = physical request 1 = functional request
	suppressPosResponse	
	Type	bit
	Comment	Position LSB+1 0 = no (do not suppress) 1 = yes (no positive response will be sent)
Description	Additional information on message request. Datastructure: Bitfield	
Available via	Dcm.h	

]

8.2.11 Dcm_IdContextType

[SWS_Dcm_00993] Definition of datatype Dcm_IdContextType [

Name	Dcm_IdContextType
Kind	Type
Derived from	uint8
Description	This message context identifier can be used to determine the relation between request and response confirmation.
Available via	Dcm.h

]

8.2.12 Dcm_MsgContextType

Please note that the following table describes a struct type definition - including its struct items "elements".

[SWS_Dcm_00994] Definition of datatype Dcm_MsgContextType [

Name	Dcm_MsgContextType	
Kind	Structure	
Elements	reqData	
	Type	Dcm_MsgType
	Comment	Request data, starting directly after service identifier (which is not part of this data)
	reqDataLen	
	Type	Dcm_MsgLenType
	Comment	Request data length (excluding service identifier)
	resData	
	Type	Dcm_MsgType
	Comment	Positive response data, starting directly after service identifier (which is not part of this data).
	resDataLen	
	Type	Dcm_MsgLenType
	Comment	Positive response data length (excluding service identifier)
	msgAddInfo	
	Type	Dcm_MsgAddInfoType
	Comment	Additional information about service request and response (see: Dcm_MsgAddInfo)
	resMaxDataLen	
	Type	Dcm_MsgLenType





	Comment	The maximal length of a response is restricted by the size of the buffer. The buffer size can depend on the diagnostic protocol identifier which is assigned to this message, e. g. an OBD protocol id can obtain other properties than the enhanced diagnostic protocol id. The resMaxData Len is a property of the diagnostic protocol assigned by the DSL. The value does not change during communication. It cannot be implemented as a constant, because it can differ between different diagnostic protocols.
	idContext	
	Type	Dcm_IdContextType
	Comment	This message context identifier can be used to determine the relation between request and response confirmation. This identifier can be stored within the application at request time, so that the response can be assigned to the original request. Background: Within the confirmation, the message context is no more valid, all message data is lost. You need an additional information to determine the request to which this confirmation belongs.
	dcmRxPduld:	
	Type	PduldType
	Comment	Pdu identifier on which the request was received. The Pduld of the request can have consequences for message processing. E. g. an OBD request will be received on the OBD Pduld and will be processed slightly different than an enhanced diagnostic request received on the physical
Description	This data structure contains all information which is necessary to process a diagnostic message from request to response and response confirmation.	
Available via	Dcm.h	

]

8.2.13 Dcm_ExtendedOpStatusType

[SWS_Dcm_91015] Definition of datatype Dcm_ExtendedOpStatusType [

Name	Dcm_ExtendedOpStatusType		
Kind	Type		
Derived from	uint8		
Range	DCM_INITIAL	0x00	Indicates the initial call to the operation
	DCM_PENDING	0x01	Indicates that a pending return has been done on the previous call of the operation
	DCM_CANCEL	0x02	Indicates that the Dcm requests to cancel the pending operation
	DCM_FORCE_RCRRP_OK	0x03	Confirm a response pending transmission Variation
	DCM_POS_RESPONSE_SENT	0x04	Indicates that a positive response has been sent successfully
	DCM_POS_RESPONSE_FAILED	0x05	Indicates that a positive response has not been sent successfully
	DCM_NEG_RESPONSE_SENT	0x06	Indicates that a negative response has been sent successfully



△

	DCM_NEG_RESPONSE_FAILED	0x07	Indicates that a negative response has not been sent successfully
Description	–		
Available via	Dcm.h		

8.3 Function definitions

This section defines the functions provided for other modules.

8.3.1 Functions provided for other BSW components

8.3.1.1 Dcm_Init

[SWS_Dcm_00037] Definition of API function Dcm_Init

Upstream requirements: [SRS_BSW_00438](#), [SRS_BSW_00101](#)

Service Name	Dcm_Init		
Syntax	<pre>void Dcm_Init (const Dcm_ConfigType* ConfigPtr)</pre>		
Service ID [hex]	0x01		
Sync/Async	Synchronous		
Reentrancy	Non Reentrant		
Parameters (in)	ConfigPtr	Pointer to configuration set in Variant Post-Build.	
Parameters (inout)	None		
Parameters (out)	None		
Return value	None		
Description	Service for basic initialization of DCM module.		
Available via	Dcm.h		

The call of this service is mandatory before using the [Dcm](#) module for further processing.

8.3.1.2 Dcm_GetVersionInfo

[SWS_Dcm_00065] Definition of API function Dcm_GetVersionInfo

Upstream requirements: [SRS_BSW_00407](#), [SRS_BSW_00482](#), [SRS_BSW_00003](#)

Service Name	Dcm_GetVersionInfo	
Syntax	void Dcm_GetVersionInfo (Std_VersionInfoType* versionInfo)	
Service ID [hex]	0x24	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	versionInfo	Pointer to where to store the version information of this module.
Return value	None	
Description	Returns the version information of this module	
Available via	Dcm.h	

8.3.1.3 Dcm_DemTriggerOnDTCStatus

[SWS_Dcm_00614] Definition of API function Dcm_DemTriggerOnDTCStatus [

Service Name	Dcm_DemTriggerOnDTCStatus	
Syntax	<pre>Std_ReturnType Dcm_DemTriggerOnDTCStatus (uint32 DTC, Dem_UdsStatusByteType DTCStatusOld, Dem_UdsStatusByteType DTCStatusNew)</pre>	
Service ID [hex]	0x2B	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	DTC	This is the DTC the change trigger is assigned to.
	DTCStatusOld	DTC status before change
	DTCStatusNew	DTC status after change
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: this value is always returned.
Description	Triggers on changes of the UDS status byte. Allows to trigger on ROE Event for subservice On DTCStatusChanged.	
Available via	Dcm_Dem.h	

]

8.3.1.4 Dcm_GetVin

[SWS_Dcm_00950] Definition of API function Dcm_GetVin [

Service Name	Dcm_GetVin	
Syntax	<pre>Std_ReturnType Dcm_GetVin (uint8* Data)</pre>	
Service ID [hex]	0x07	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	Data	Pointer to where to store the VIN
Return value	Std_ReturnType	E_OK: The Data pointer has been filled with valid VIN E_NOT_OK: The default VIN will be used in the DoIP
Description	Function to get the VIN (as defined in SAE J1979-DA)	
Available via	Dcm.h	

]

Note: After fetching the VIN, the [Dcm](#) can offer the data to all users without worrying that the data is unavailable if a user asks for it. This is necessary because the VIN could not be fetched synchronously for all settings of DcmDspDidDataUsePort.

Note: The invocation of functions BndM_WriteStart and BndM_WriteFinalize are not part of [Dcm](#) Specification. The functions are called via project specific CDD implementation.

8.3.1.5 Dcm_BndMWriteBlockFinish

[SWS_Dcm_91088] Definition of API function Dcm_BndMWriteBlockFinish

Upstream requirements: [RS_Diag_04243](#)

[

Service Name	Dcm_BndMWriteBlockFinish	
Syntax	<pre>void Dcm_BndMWriteBlockFinish (BndM_BlockIdType BlockId, BndM_ResultType result)</pre>	
Service ID [hex]	0x7a	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	BlockId	–
	result	–
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	





Description	Called by BndM to indicate that a block write operation has finished.
Available via	Dcm.h

└

8.3.2 Functions provided to BSW modules and to SW-Cs

The functions defined in this section can also be used by SW-Cs through the RTE.

8.3.2.1 Dcm_SetDeauthenticatedRole

[SWS_Dcm_91069] Definition of API function Dcm_SetDeauthenticatedRole

Service Name	Dcm_SetDeauthenticatedRole	
Syntax	<pre>Std_ReturnType Dcm_SetDeauthenticatedRole (uint16 connectionId, const Dcm_AuthenticationRoleType* deauthenticatedRole)</pre>	
Service ID [hex]	0x79	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	connectionId	Unique connection identifier identifying the connection for which a deauthenticated roles is set.
	deauthenticatedRole	New deauthenticated role that is assigned to that connection
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: this value is always returned.
Description	Sets a new role used in deauthenticated state for that connection. The set role is valid until the connection switches into authenticated state or the ECU is reset.	
Available via	Dcm.h	

└

8.3.2.2 Dcm_GetSecurityLevel

[SWS_Dcm_00338] Definition of API function Dcm_GetSecurityLevel

Upstream requirements: [RS_Diag_04005](#), [RS_Diag_04011](#)

└

Service Name	Dcm_GetSecurityLevel	
Syntax	<pre>Std_ReturnType Dcm_GetSecurityLevel (Dcm_SecLevelType* SecLevel)</pre>	
Service ID [hex]	0x0d	





Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	SecLevel	Active Security Level value Conversion formula to calculate SecurityLevel out of tester requested SecurityAccessType parameter: $\text{SecurityLevel} = (\text{SecurityAccessType} + 1) / 2$ Content of SecurityAccessType is according to "securityAccessType" parameter of SecurityAccess request (see [11])
Return value	Std_ReturnType	E_OK: this value is always returned.
Description	This function provides the active security level value.	
Available via	Dcm.h	

8.3.2.3 Dcm_GetSesCtrlType

[SWS_Dcm_00339] Definition of API function Dcm_GetSesCtrlType

Upstream requirements: [RS_Diag_04006](#), [RS_Diag_04011](#)

Service Name	Dcm_GetSesCtrlType	
Syntax	<pre>Std_ReturnType Dcm_GetSesCtrlType (Dcm_SesCtrlType* SesCtrlType)</pre>	
Service ID [hex]	0x06	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	SesCtrlType	Active Session Control Type value Content is according to "diagnosticSessionType" parameter of DiagnosticSessionControl request (see [11])
Return value	Std_ReturnType	E_OK: this value is always returned.
Description	This function provides the active session control type value.	
Available via	Dcm.h	

8.3.2.4 Dcm_ResetToDefaultSession

[SWS_Dcm_00520] Definition of API function Dcm_ResetToDefaultSession [

Service Name	Dcm_ResetToDefaultSession	
Syntax	<pre>Std_ReturnType Dcm_ResetToDefaultSession (void)</pre>	
Service ID [hex]	0x2a	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: this value is always returned.
Description	The call to this function allows the application to reset the current session to Default session. Example: Automatic termination of an extended diagnostic session upon exceeding of a speed limit.	
Available via	Dcm.h	

]

8.3.2.5 Dcm_SetActiveDiagnostic

[SWS_Dcm_01068] Definition of API function Dcm_SetActiveDiagnostic [

Service Name	Dcm_SetActiveDiagnostic	
Syntax	<pre>Std_ReturnType Dcm_SetActiveDiagnostic (boolean active)</pre>	
Service ID [hex]	0x56	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	active	If false Dcm shall not call ComM_DCM_ActiveDiagnostic(). If true Dcm will call ComM_DCM_ActiveDiagnostic().
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: this value is always returned.
Description	Allows to activate and deactivate the call of ComM_DCM_ActiveDiagnostic() function.	
Available via	Dcm.h	

]

8.4 Callback notifications

This section defines the functions provided for lower layer BSW modules.

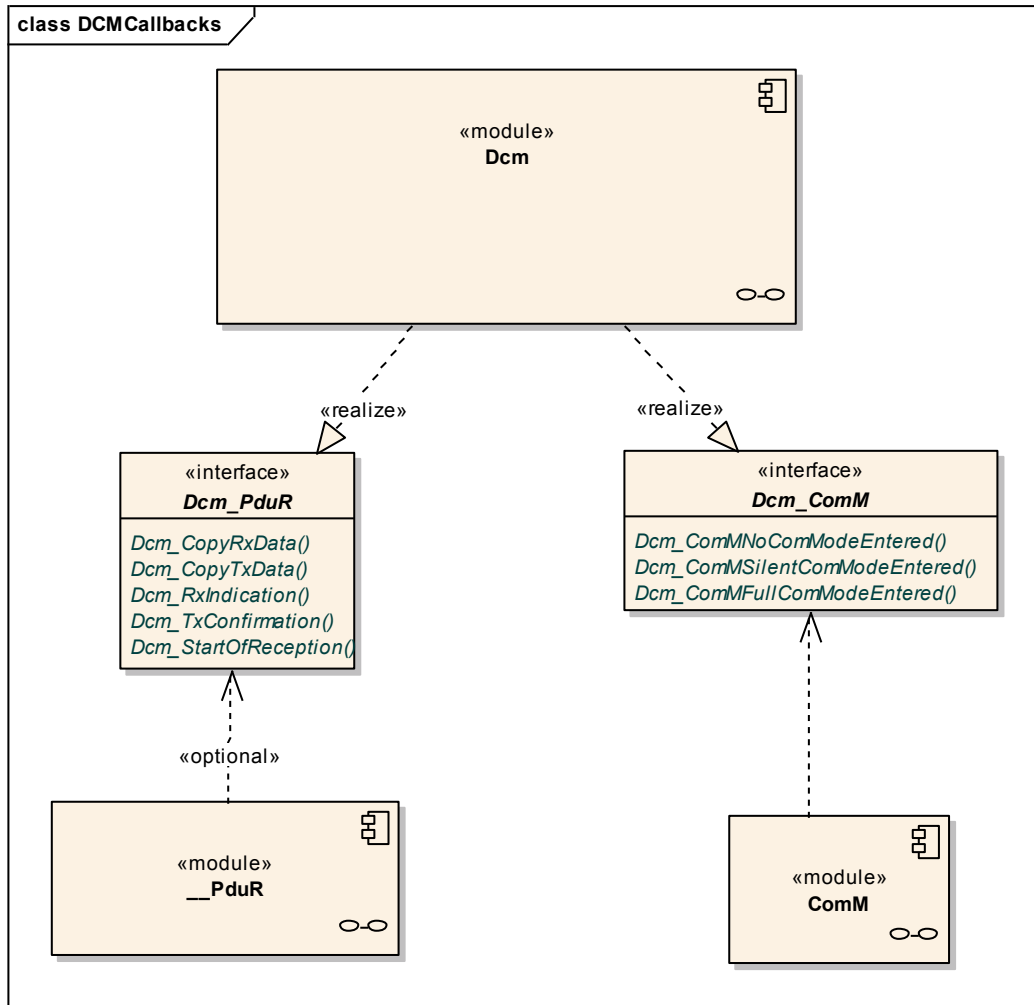


Figure 8.1: Overview of the callbacks provided by the DCM

8.4.1 Dcm_StartOfReception

[SWS_Dcm_00094] Definition of callback function Dcm_StartOfReception [

Service Name	Dcm_StartOfReception	
Syntax	<pre> BufReq_ReturnType Dcm_StartOfReception (PduIdType id, const PduInfoType* info, PduLengthType TpSduLength, PduLengthType* bufferSizePtr) </pre>	
Service ID [hex]	0x46	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	id	Identification of the I-PDU.





	info	Pointer to a PduInfoType structure containing the payload data (without protocol information) and payload length of the first frame or single frame of a transport protocol I-PDU reception, and the MetaData related to this PDU. If neither first/single frame data nor MetaData are available, this parameter is set to NULL_PTR.
	TpSduLength	Total length of the N-SDU to be received.
Parameters (inout)	None	
Parameters (out)	bufferSizePtr	Available receive buffer in the receiving module. This parameter will be used to compute the Block Size (BS) in the transport protocol module.
Return value	BufReq_ReturnType	BUFREQ_OK: Connection has been accepted. bufferSizePtr indicates the available receive buffer; reception is continued. If no buffer of the requested size is available, a receive buffer size of 0 shall be indicated by bufferSizePtr. BUFREQ_E_NOT_OK: Connection has been rejected; reception is aborted. bufferSizePtr remains unchanged. BUFREQ_E_OVFL: No buffer of the required length can be provided; reception is aborted. bufferSizePtr remains unchanged.
Description	This function is called at the start of receiving an N-SDU. The N-SDU might be fragmented into multiple N-PDUs (FF with one or more following CFs) or might consist of a single N-PDU (SF). If the service is successful and BUFREQ_OK is returned, the service provides the currently available maximum buffer size. This function might be called in interrupt context.	
Available via	Dcm.h	

]

By the function [Dcm_StartOfReception](#) the receiver (e.g. DCM) is also informed implicitly about a first frame reception or a single frame reception. If the function [Dcm_StartOfReception](#) returns a return value not equal to BUFREQ_OK, the values of the out parameters are not specified and should not be evaluated by the caller.

8.4.2 Dcm_CopyRxData

[SWS_Dcm_00556] Definition of API function Dcm_CopyRxData [

Service Name	Dcm_CopyRxData	
Syntax	<pre>BufReq_ReturnType Dcm_CopyRxData (PduIdType id, const PduInfoType* info, PduLengthType* bufferSizePtr)</pre>	
Service ID [hex]	0x44	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	id	Identification of the received I-PDU.
	info	Provides the source buffer (SduDataPtr) and the number of bytes to be copied (SduLength). An SduLength of 0 can be used to query the current amount of available buffer in the upper layer module. In this case, the SduDataPtr may be a NULL_PTR.
Parameters (inout)	None	
Parameters (out)	bufferSizePtr	Available receive buffer after data has been copied.





Return value	BufReq_ReturnType	BUFREQ_OK: Data copied successfully BUFREQ_E_NOT_OK: Data was not copied because an error occurred.
Description	This function is called to provide the received data of an I-PDU segment (N-PDU) to the upper layer. Each call to this function provides the next part of the I-PDU data. The size of the remaining buffer is written to the position indicated by bufferSizePtr. This function might be called in interrupt context.	
Available via	Dcm.h	

]

8.4.3 Dcm_TpRxIndication

[SWS_Dcm_00093] Definition of API function Dcm_TpRxIndication [

Service Name	Dcm_TpRxIndication	
Syntax	<pre>void Dcm_TpRxIndication (PduIdType id, Std_ReturnType result)</pre>	
Service ID [hex]	0x45	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	id	Identification of the received I-PDU.
	result	E_OK: The I-PDU was received successfully. E_NOT_OK: Reception of the I-PDU failed.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Called after an I-PDU has been received via the TP API, the result indicates whether the transmission was successful or not. This function might be called in interrupt context.	
Available via	Dcm.h	

]

8.4.4 Dcm_CopyTxData

[SWS_Dcm_00092] Definition of API function Dcm_CopyTxData [

Service Name	Dcm_CopyTxData	
Syntax	<pre>BufReq_ReturnType Dcm_CopyTxData (PduIdType id, const PduInfoType* info, const RetryInfoType* retry, PduLengthType* availableDataPtr)</pre>	
Service ID [hex]	0x43	
Sync/Async	Synchronous	





Reentrancy	Reentrant	
Parameters (in)	id	Identification of the transmitted I-PDU.
	info	Provides the destination buffer (SduDataPtr) and the number of bytes to be copied (SduLength). If not enough transmit data is available, no data is copied by the upper layer module and BUFREQ_E_BUSY is returned. The lower layer module may retry the call. An SduLength of 0 can be used to indicate state changes in the retry parameter or to query the current amount of available data in the upper layer module. In this case, the SduDataPtr may be a NULL_PTR.
	retry	This parameter is used to acknowledge transmitted data or to retransmit data after transmission problems. If the retry parameter is a NULL_PTR, it indicates that the transmit data can be removed from the buffer immediately after it has been copied. Otherwise, the retry parameter must point to a valid RetryInfoType element. If TpDataState indicates TP_CONFENDING, the previously copied data must remain in the TP buffer to be available for error recovery. TP_DATACONF indicates that all data that has been copied before this call is confirmed and can be removed from the TP buffer. Data copied by this API call is excluded and will be confirmed later. TP_DATA_RETRY indicates that this API call shall copy previously copied data in order to recover from an error. In this case TxTpDataCnt specifies the offset in bytes from the current data copy position.
Parameters (inout)	None	
Parameters (out)	availableDataPtr	Indicates the remaining number of bytes that are available in the upper layer module's Tx buffer. availableDataPtr can be used by TP modules that support dynamic payload lengths (e.g. FrIsoTp) to determine the size of the following CFs.
Return value	BufReq_ReturnType	BUFREQ_OK: Data has been copied to the transmit buffer completely as requested. BUFREQ_E_BUSY: Request could not be fulfilled, because the required amount of Tx data is not available. The lower layer module may retry this call later on. No data has been copied. BUFREQ_E_NOT_OK: Data has not been copied. Request failed.
Description	This function is called to acquire the transmit data of an I-PDU segment (N-PDU). Each call to this function provides the next part of the I-PDU data unless retry->TpDataState is TP_DATA_RETRY. In this case the function restarts to copy the data beginning at the offset from the current position indicated by retry->TxTpDataCnt. The size of the remaining data is written to the position indicated by availableDataPtr. This function might be called in interrupt context.	
Available via	Dcm.h	

]

8.4.5 Dcm_TpTxConfirmation

[SWS_Dcm_00351] Definition of API function Dcm_TpTxConfirmation [

Service Name	Dcm_TpTxConfirmation
Syntax	<pre>void Dcm_TpTxConfirmation (PduIdType id, Std_ReturnType result)</pre>
Service ID [hex]	0x48
Sync/Async	Synchronous





Reentrancy	Reentrant	
Parameters (in)	id	Identification of the transmitted I-PDU.
	result	E_OK: The I-PDU was transmitted successfully. E_NOT_OK: Transmission of the I-PDU failed.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This function is called after the I-PDU has been transmitted on its network, the result indicates whether the transmission was successful or not. This function might be called in interrupt context.	
Available via	Dcm.h	

]

8.4.6 Dcm_TxConfirmation

[SWS_Dcm_01092] Definition of API function Dcm_TxConfirmation [

Service Name	Dcm_TxConfirmation	
Syntax	<pre>void Dcm_TxConfirmation (PduIdType TxPduId, Std_ReturnType result)</pre>	
Service ID [hex]	0x40	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	TxPduId	ID of the PDU that has been transmitted.
	result	E_OK: The PDU was transmitted. E_NOT_OK: Transmission of the PDU failed.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU. This function might be called in interrupt context.	
Available via	Dcm.h	

]

8.4.7 Dcm_ComM_NoComModeEntered

[SWS_Dcm_00356] Definition of callback function Dcm_ComM_NoComModeEntered [

Service Name	Dcm_ComM_NoComModeEntered	
Syntax	<pre>void Dcm_ComM_NoComModeEntered (uint8 NetworkId)</pre>	





Service ID [hex]	0x21	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	NetworkId	Identifier of the network concerned by the mode change
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This call informs the Dcm module about a ComM mode change to COMM_NO_COMMUNICATION.	
Available via	Dcm_ComM.h	

]

8.4.8 Dcm_ComM_SilentComModeEntered

[SWS_Dcm_00358] Definition of callback function Dcm_ComM_SilentComModeEntered [

Service Name	Dcm_ComM_SilentComModeEntered	
Syntax	<pre>void Dcm_ComM_SilentComModeEntered (uint8 NetworkId)</pre>	
Service ID [hex]	0x22	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	NetworkId	Identifier of the network concerned by the mode change
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This call informs the Dcm module about a ComM mode change to COMM_SILENT_COMMUNICATION.	
Available via	Dcm_ComM.h	

]

8.4.9 Dcm_ComM_FullComModeEntered

[SWS_Dcm_00360] Definition of callback function Dcm_ComM_FullComModeEntered [

Service Name	Dcm_ComM_FullComModeEntered	
Syntax	<pre>void Dcm_ComM_FullComModeEntered (uint8 NetworkId)</pre>	
Service ID [hex]	0x23	
Sync/Async	Synchronous	





Reentrancy	Reentrant	
Parameters (in)	NetworkId	Identifier of the network concerned by the mode change
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This call informs the Dcm module about a ComM mode change to COMM_FULL_COMMUNICATION.	
Available via	Dcm_ComM.h	

]

8.4.10 Dcm_CsmAsyncJobFinished

[SWS_Dcm_91076] Definition of configurable interface Dcm_CsmAsyncJobFinished [

Service Name	Dcm_CsmAsyncJobFinished	
Syntax	<pre>void Dcm_CsmAsyncJobFinished (uint32 jobId, Crypto_ResultType result)</pre>	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	jobId	JobId provided by the Csm, indicating the job that has finished.
	result	Return value of the asynchronous job
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Can be called from Csm upon finishing an asynchronous job processing. The integrator will configure this name as callback function within the Csm ECUC configuration for asynchronous jobs. Only one such callback is available, the Dcm detects the job that has finished by evaluating the job parameter. API availability: This API will be available only if ({ecuc(Dcm/Dcm Dsp/DcmDspAuthentication/DcmDspAuthenticationConnection) != null}).	
Available via		

]

8.4.11 Dcm_KeyMAsyncCertificateVerifyFinished

[SWS_Dcm_91077] Definition of configurable interface Dcm_KeyMAsyncCertificateVerifyFinished [

Service Name	Dcm_KeyMAsyncCertificateVerifyFinished	
Syntax	<pre>Std_ReturnType Dcm_KeyMAsyncCertificateVerifyFinished (KeyM_CertificateIdType CertID, KeyM_CertificateStatusType Result)</pre>	





Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	CertID	Certificate identifier that has finished the verification
	Result	Return value of the asynchronous job
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: this value is always returned
Description	Can be called from Key upon finishing an asynchronous certificate verification. The integrator will configure this name as callback function within the KeyM ECUC configuration for asynchronous jobs. Only one such callback is available, the Dcm detects the certificate that has finished by evaluating the certId parameter. API availability: This API will be available only if ({ecuc(Dcm/DcmDsp/DcmDspAuthentication/DcmDspAuthenticationConnection) != null}).	
Available via		

]

8.5 Callout Definitions

Callouts are pieces of code that have to be added to the [Dcm](#) during ECU integration. The content of most callouts is hand-written code, for some callouts the [Dcm](#) configuration tool shall generate a default implementation that is manually edited by the integrator. Conceptually, these callouts belong to the ECU Firmware.

Since callouts are no services of the [Dcm](#) they do not have an assigned Service ID. Note: The Autosar architecture doesn't provide the possibility to access the ECU memory using a physical address. This realized using BlockId which identified a memory block.

According to that, the [Dcm](#) is not able to fully support the implementation of ISO14229-1 [1]services which request a physical memory access. Therefore, the [Dcm](#) define callout to realize this kind of memory access. This callout implementation could be simply realized by defining a mapping between the BlockId and the physical memory address.

8.5.1 Dcm_ReadMemory

[SWS_Dcm_00539] Definition of configurable interface Dcm_ReadMemory [

Service Name	Dcm_ReadMemory	
Syntax	<pre> Dcm_ReturnReadMemoryType Dcm_ReadMemory (Dcm_OpStatusType OpStatus, uint8 MemoryIdentifier, uint32 MemoryAddress, uint32 MemorySize, uint8* MemoryData, Dcm_NegativeResponseCodeType* ErrorCode) </pre>	
Service ID [hex]	0x7c	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL DCM_PENDING DCM_CANCEL DCM_FORCE_RCRRP_OK
	MemoryIdentifier	Identifier of the Memory Block (e.g. used if memory section distinguishing is needed) Note: If it's not used this parameter shall be set to 0.
	MemoryAddress	Starting address of server memory from which data is to be retrieved.
	MemorySize	Number of bytes in the MemoryData
Parameters (inout)	None	
Parameters (out)	MemoryData	Data read (Points to the diagnostic buffer in DCM)
	ErrorCode	If the operation Dcm_ReadMemory returns value DCM_READ_FAILED, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Dcm_ReturnReadMemoryType	DCM_READ_OK: read was successful DCM_READ_FAILED: read was not successful DCM_READ_PENDING: read is not yet finished DCM_READ_FORCE_RCRRP: reading is pending, the Response pending transmission starts immediately
Description	The Dcm_ReadMemory callout is used to request memory data identified by the parameter memoryAddress and memorySize from the UDS request message. This service is needed for the implementation of UDS services: • ReadMemoryByAddress • RequestUpload • ReadDataByIdentifier (in case of Dynamical DID defined by memory address) • TransferData	
Available via	Dcm_Externals.h	

]

8.5.2 Dcm_WriteMemory

[SWS_Dcm_00540] Definition of configurable interface Dcm_WriteMemory [

Service Name	Dcm_WriteMemory	
Syntax	<pre> Dcm_ReturnWriteMemoryType Dcm_WriteMemory (Dcm_OpStatusType OpStatus, uint8 MemoryIdentifier, uint32 MemoryAddress, uint32 MemorySize, const uint8* MemoryData, Dcm_NegativeResponseCodeType* ErrorCode) </pre>	
Service ID [hex]	0x27	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL DCM_PENDING DCM_CANCEL DCM_FORCE_RCRRP_OK
	MemoryIdentifier	Identifier of the Memory Block (e.g. used by WriteDataByIdentifier service). Note: If it's not used this parameter shall be set to 0.
	MemoryAddress	Starting address of server memory in which data is to be copied. Note: If it's not used (e.g. if the data is compressed) this parameter shall be set to 0.
	MemorySize	Number of bytes in MemoryData
	MemoryData	Data to write (Points to the diagnostic buffer in DCM)
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Dcm_WriteMemory returns value DCM_WRITE_FAILED, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Dcm_ReturnWriteMemoryType	DCM_WRITE_OK: write was successful DCM_WRITE_FAILED: write was not successful DCM_WRITE_PENDING: write is not yet finished DCM_WRITE_FORCE_RCRRP: writing is pending, the Response pending transmission starts immediately
Description	The Dcm_WriteMemory callout is used to write memory data identified by the parameter memoryAddress and memorySize. This service is needed for the implementation of UDS services : • WriteMemoryByAddress • RequestDownload • TransferData	
Available via	Dcm_Externals.h	

]

Note : The callout implementation shall take care of the following points :

- When writing data in NVRAM, take care to keep the consistency with data in the mirror RAM
- When writing data in memory, take care that a SW-C won't overwrite the data. Maybe the SW-C should be informed of this writing

8.5.3 Dcm_SetProgConditions

[SWS_Dcm_00543] Definition of configurable interface Dcm_SetProgConditions

Service Name	Dcm_SetProgConditions	
Syntax	<pre>Std_ReturnType Dcm_SetProgConditions (Dcm_OpStatusType OpStatus, const Dcm_ProgConditionsType* ProgConditions)</pre>	
Service ID [hex]	0x61	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	OpStatus DCM_INITIAL DCM_PENDING DCM_CANCEL DCM_FORCE_RCRRP_OK
	ProgConditions	Conditions on which the jump to bootloader has been requested
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Conditions have correctly been set E_NOT_OK: Conditions cannot be set DCM_E_PENDING: Conditions set is in progress, a further call to this API is needed to end the setting DCM_E_FORCE_RCRRP: Application requests the transmission of a response Response Pending (NRC 0x78)
Description	The Dcm_SetProgConditions callout allows the integrator to store relevant information prior to jumping to bootloader / jump due to ECUReset request. The context parameter are defined in Dcm_ProgConditionsType.	
Available via	Dcm_Externals.h	

Note: In case the SecurityAccess AttemptCounter needs to be shared between application and bootloader in addition to the ProgConditionStructure the current value can be retrieved via the [API Xxx_GetSecurityAttemptCounter](#) (see chapter 7.4.4 Interaction)

8.5.4 Dcm_GetProgConditions

[SWS_Dcm_00544] Definition of configurable interface Dcm_GetProgConditions

Service Name	Dcm_GetProgConditions	
Syntax	<pre>Dcm_EcuStartModeType Dcm_GetProgConditions (Dcm_ProgConditionsType * ProgConditions)</pre>	
Service ID [hex]	0x62	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	





Parameters (out)	ProgConditions	Conditions on which the jump from the bootloader has been requested
Return value	Dcm_EcuStartModeType	–
Description	The Dcm_GetProgConditions callout is called upon Dcm initialization and allows to determine if a response (\$50 or \$51) has to be sent. The context parameters are defined in Dcm_ProgConditionsType.	
Available via	Dcm_Externals.h	

]

8.5.5 Dcm_ProcessRequestAddFile

[SWS_Dcm_91078] Definition of configurable interface Dcm_ProcessRequestAddFile

Upstream requirements: [RS_Diag_04135](#)

[

Service Name	Dcm_ProcessRequestAddFile	
Syntax	<pre>Std_ReturnType Dcm_ProcessRequestAddFile (Dcm_OpStatusType OpStatus, uint16 filePathAndNameLength, const uint8* filePathAndName, uint8 dataFormatIdentifier, uint64 fileSizeUncompressed, uint64 fileSizeCompressed, uint64* maxNumberOfBlockLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x72	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRRP_OK: All In-parameters are set to 0x00.
	filePathAndNameLength	Defines the length in bytes for the parameter filePathAndName.
	filePathAndName	Defines the file system location of the server where the file which shall be added, deleted, replaced or read from depending on the parameter modeOfOperation parameter. In addition this parameter includes the file name of the file which shall be added, deleted, replaced or read as part of the file path.
	dataFormatIdentifier	This data-parameter is a one byte value with each nibble encoded separately. The high nibble specifies the "compression Method", and the low nibble specifies the "encryptingMethod". The value 0x00 specifies that neither compressionMethod nor encryptingMethod is used. Values other than 0x00 are vehicle manufacturer specific.
	fileSizeUncompressed	Defines the size of the uncompressed file to be download in bytes.
	fileSizeCompressed	Defines the size of the compressed file to be downloaded in bytes.
Parameters (inout)	None	





Parameters (out)	maxNumberOfBlockLength	Max number of bytes to be included in each TransferData request excluding the SID and the blockSequenceCounter.
	ErrorCode	If the operation Dcm_ProcessRequestAddFile returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful E_NOT_OK: Request was not successful DCM_E_PENDING: Request is not yet finished DCM_E_FORCE_RCRRP: Application request the transmission of a response Response Pending (NRC 0x78)
Description	Callout function. DCM shall call this function to start a RequestFileTransfer process with mode OfOperation equal to 0x01 (AddFile).	
Available via	Dcm_Externals.h	

]

8.5.6 Dcm_ProcessRequestDeleteFile

[SWS_Dcm_91079] Definition of configurable interface Dcm_ProcessRequestDeleteFile

Upstream requirements: [RS_Diag_04135](#)

[

Service Name	Dcm_ProcessRequestDeleteFile	
Syntax	<pre>Std_ReturnType Dcm_ProcessRequestDeleteFile (Dcm_OpStatusType OpStatus, uint16 filePathAndNameLength, const uint8* filePathAndName, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x73	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRRP_OK: All In-parameters are set to 0x00.
	filePathAndNameLength	Defines the length in bytes for the parameter filePathAndName.
	filePathAndName	Defines the file system location of the server where the file which shall be added, deleted, replaced or read from depending on the parameter modeOfOperation parameter. In addition this parameter includes the file name of the file which shall be added, deleted, replaced or read as part of the file path.
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Dcm_ProcessRequestDeleteFile returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.





Return value	Std_ReturnType	E_OK: Request was successful E_NOT_OK: Request was not successful DCM_E_PENDING: Request is not yet finished DCM_E_FORCE_RCRRP: Application request the transmission of a response Response Pending (NRC 0x78)
Description	Callout function. DCM shall call this function to start a RequestFileTransfer process with mode OfOperation equal to 0x02 (DeleteFile).	
Available via	Dcm_Externals.h	

]

8.5.7 Dcm_ProcessRequestReplaceFile

[SWS_Dcm_91080] Definition of configurable interface Dcm_ProcessRequestReplaceFile

Upstream requirements: [RS_Diag_04135](#)

[

Service Name	Dcm_ProcessRequestReplaceFile	
Syntax	<pre>Std_ReturnType Dcm_ProcessRequestReplaceFile (Dcm_OpStatusType OpStatus, uint16 filePathAndNameLength, const uint8* filePathAndName, uint8 dataFormatIdentifier, uint64 fileSizeUncompressed, uint64 fileSizeCompressed, uint64* maxNumberOfBlockLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x74	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRRP_OK: All In-parameters are set to 0x00.
	filePathAndNameLength	Defines the length in bytes for the parameter filePathAndName.
	filePathAndName	Defines the file system location of the server where the file which shall be added, deleted, replaced or read from depending on the parameter modeOfOperation parameter. In addition this parameter includes the file name of the file which shall be added, deleted, replaced or read as part of the file path.
	dataFormatIdentifier	This data-parameter is a one byte value with each nibble encoded separately. The high nibble specifies the "compression Method", and the low nibble specifies the "encryptingMethod". The value 0x00 specifies that neither compressionMethod nor encryptingMethod is used. Values other than 0x00 are vehicle manufacturer specific.
	fileSizeUncompressed	Defines the size of the uncompressed file to be download in bytes.
	fileSizeCompressed	Defines the size of the compressed file to be downloaded in bytes.





Parameters (inout)	None	
Parameters (out)	maxNumberOfBlockLength	Max number of bytes to be included in each TransferData request excluding the SID and the blockSequenceCounter.
	ErrorCode	If the operation Dcm_ProcessRequestReplaceFile returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful E_NOT_OK: Request was not successful DCM_E_PENDING: Request is not yet finished DCM_E_FORCE_RCRP: Application request the transmission of a response Response Pending (NRC 0x78)
Description	Callout function. DCM shall call this function to start a RequestFileTransfer process with mode OfOperation equal to 0x03 (ReplaceFile).	
Available via	Dcm_Externals.h	

]

8.5.8 Dcm_ProcessRequestReadFile

[SWS_Dcm_91081] Definition of configurable interface Dcm_ProcessRequestReadFile

Upstream requirements: [RS_Diag_04135](#)

[

Service Name	Dcm_ProcessRequestReadFile	
Syntax	<pre>Std_ReturnType Dcm_ProcessRequestReadFile (Dcm_OpStatusType OpStatus, uint16 filePathAndNameLength, const uint8* filePathAndName, uint8 dataFormatIdentifier, uint64 fileSizeUncompressed, uint64 fileSizeCompressed, uint64* maxNumberOfBlockLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x75	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRP_OK: All In-parameters are set to 0x00.
	filePathAndNameLength	Defines the length in bytes for the parameter filePathAndName.
	filePathAndName	Defines the file system location of the server where the file which shall be added, deleted, replaced or read from depending on the parameter modeOfOperation parameter. In addition this parameter includes the file name of the file which shall be added, deleted, replaced or read as part of the file path.





	dataFormatIdentifier	This data-parameter is a one byte value with each nibble encoded separately. The high nibble specifies the "compression Method", and the low nibble specifies the "encryptingMethod". The value 0x00 specifies that neither compressionMethod nor encryptingMethod is used. Values other than 0x00 are vehicle manufacturer specific.
Parameters (inout)	None	
Parameters (out)	fileSizeUncompressed	Defines the size of the uncompressed file to be uploaded in bytes.
	fileSizeCompressed	Defines the size of the compressed file to be uploaded in bytes.
	maxNumberOfBlock Length	Max number of bytes to be included in each TransferData response excluding the SID and the blockSequenceCounter.
	ErrorCode	If the operation Dcm_ProcessRequestReadFile returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful E_NOT_OK: Request was not successful DCM_E_PENDING: Request is not yet finished DCM_E_FORCE_RCRRP: Application request the transmission of a response Response Pending (NRC 0x78)
Description	Callout function. DCM shall call this function to start a RequestFileTransfer process with mode OfOperation equal to 0x04 (ReadFile).	
Available via	Dcm_Externals.h	

]

8.5.9 Dcm_ProcessRequestReadDir

[SWS_Dcm_91082] Definition of configurable interface Dcm_ProcessRequestReadDir

Upstream requirements: [RS_Diag_04135](#)

[

Service Name	Dcm_ProcessRequestReadDir	
Syntax	<pre>Std_ReturnType Dcm_ProcessRequestReadDir (Dcm_OpStatusType OpStatus, uint16 filePathAndNameLength, const uint8* filePathAndName, uint64* dirInfoLength, uint64* maxNumberOfBlockLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x76	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRRP_OK: All In-parameters are set to 0x00.
	filePathAndNameLength	Defines the length in bytes for the parameter filePathAndName.





	filePathAndName	Defines the file system location of the server where the file which shall be added, deleted, replaced or read from depending on the parameter modeOfOperation parameter. In addition this parameter includes the file name of the file which shall be added, deleted, replaced or read as part of the file path.
Parameters (inout)	None	
Parameters (out)	dirInfoLength	Defines the size of directory information to be uploaded in bytes.
	maxNumberOfBlockLength	Max number of bytes to be included in each TransferData request excluding the SID and the blockSequenceCounter.
	ErrorCode	If the operation Dcm_ProcessRequestReadDir returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful E_NOT_OK: Request was not successful DCM_E_PENDING: Request is not yet finished DCM_E_FORCE_RCRP: Application request the transmission of a response Response Pending (NRC 0x78)
Description	Callout function. DCM shall call this function to start a RequestFileTransfer process with mode OfOperation equal to 0x05 (ReadDir).	
Available via	Dcm_Externals.h	

]

8.5.10 Dcm_WriteFile

[SWS_Dcm_91083] Definition of configurable interface Dcm_WriteFile

Upstream requirements: [RS_Diag_04135](#)

[

Service Name	Dcm_WriteFile	
Syntax	<pre>Std_ReturnType Dcm_WriteFile (Dcm_OpStatusType OpStatus, uint64 DataLength, uint8* Data, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRP_OK: All In-parameters are set to 0x00.
	DataLength	Defines the length in bytes for the parameter Data. The value will not exceed, but might be less, compared to the value of max NumberOfBlockLength return in Dcm_ProcessRequestFile Transfer.
	Data	Pointer to the data to be written.
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Dcm_WriteFile returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.





Return value	Std_ReturnType	E_OK: Request was successful E_NOT_OK: Request was not successful DCM_E_PENDING: Request is not yet finished DCM_E_FORCE_RCRRP: Application request the transmission of a response Response Pending (NRC 0x78)
Description	Callout function. DCM shall call this function when data is received using UDS service Transfer Data if there's an ongoing RequestFileTransfer process started with 0x01 (AddFile) or 0x03 (ReplaceFile).	
Available via	Dcm_Externals.h	

]

8.5.11 Dcm_ReadFileOrDir

[SWS_Dcm_91085] Definition of configurable interface Dcm_ReadFileOrDir

Upstream requirements: [RS_Diag_04135](#)

[

Service Name	Dcm_ReadFileOrDir	
Syntax	<pre>Std_ReturnType Dcm_ReadFileOrDir (Dcm_OpStatusType OpStatus, uint64* DataLength, uint8* Data, Dcm_NegativeResponseType* ErrorCode)</pre>	
Service ID [hex]	0x78	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRRP_OK: All In-parameters are set to 0x00.
	Data	Pointer to the data to be written.
Parameters (inout)	DataLength	As in, the parameter defines the maximum block length to be used, i.e. the value of maxNumberOfBlockLength sent to the client in the response of RequestFileTransfer. As out, the parameter defines the actual length in bytes for the parameter Data. The value shall not exceed, but might be less, the value provided as in parameter.
Parameters (out)	ErrorCode	If the operation Dcm_ReadFileOrDir returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful E_NOT_OK: Request was not successful DCM_E_PENDING: Request is not yet finished DCM_E_FORCE_RCRRP: Application request the transmission of a response Response Pending (NRC 0x78)
Description	Callout function. DCM shall call this function when data shall be sent as a response to UDS service TransferData if there's an ongoing RequestFileTransfer process started with 0x04 (ReadFile) or 0x05 (ReadDir).	
Available via	Dcm_Externals.h	

]

8.5.12 UploadDownloadServices

From the point of view of the DCM, the operations has the following signatures:

8.5.12.1 ProcessRequestDownload

[SWS_Dcm_00754] Definition of configurable interface Dcm_ProcessRequestDownload

Service Name	Dcm_ProcessRequestDownload	
Syntax	<pre>Std_ReturnType Dcm_ProcessRequestDownload (Dcm_OpStatusType OpStatus, uint8 DataFormatIdentifier, uint8 MemoryIdentifier, uint32 MemoryAddress, uint32 MemorySize, uint32* BlockLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x30	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL DCM_PENDING DCM_CANCEL
	DataFormatIdentifier	Bit 7 - 4: Compression Method • 0x0: not compressed • 0x1..F: vehicle-manufacturer-specific Bit 3 - 0: Encrypting method • 0x0: not encrypted • 0x1..F: vehicle-manufacturer-specific
	MemoryIdentifier	Identifier of the Memory Block, if the parameter is not used it shall be set to 0.
	MemoryAddress	Starting address of server memory to which data is to be written
	MemorySize	Uncompressed memory size in bytes
Parameters (inout)	BlockLength	Max. Number of bytes for one Dcm_WriteMemory
Parameters (out)	ErrorCode	If the operation Dcm_ProcessRequestDownload returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful E_NOT_OK: Request was not successful DCM_E_PENDING: Request is not yet finished
Description	Callout function. DCM shall call this callout function to start a download process. This service is needed for the implementation of UDS service RequestDownload.	
Available via	Dcm_Externals.h	

]

8.5.12.2 ProcessRequestTransferExit

[SWS_Dcm_00755] Definition of configurable interface Dcm_ProcessRequestTransferExit

Service Name	Dcm_ProcessRequestTransferExit	
Syntax	<pre>Std_ReturnType Dcm_ProcessRequestTransferExit (Dcm_OpStatusType OpStatus, const uint8* transferRequestParameterRecord, uint32 transferRequestParameterRecordSize, uint8* transferResponseParameterRecord, uint32* transferResponseParameterRecordSize, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x32	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL DCM_PENDING DCM_CANCEL
	transferRequestParameterRecord	Pointer to vehicle manufacturer specific transferRequestParameterRecord
	transferRequestParameterRecordSize	Length of ParameterRecord in bytes
Parameters (inout)	transferResponseParameterRecordSize	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The function returns the actual number of written data bytes in transferResponseParameterRecord.
Parameters (out)	transferResponseParameterRecord	Pointer to vehicle manufacturer specific transferResponseParameterRecord
	ErrorCode	see below
Return value	Std_ReturnType	E_OK: Transfer was successful E_NOT_OK: Transfer was not successful or the response buffer is too small DCM_E_PENDING: Transfer is not yet finished
Description	Callout function. DCM shall call this callout function to terminate a download or upload process. This callout is needed for the implementation of UDS service RequestTransferExit.	
Available via	Dcm_Externals.h	

]

8.5.12.3 ProcessRequestUpload

[SWS_Dcm_00756] Definition of configurable interface Dcm_ProcessRequestUpload

Service Name	Dcm_ProcessRequestUpload	
Syntax	<pre>Std_ReturnType Dcm_ProcessRequestUpload (Dcm_OpStatusType OpStatus, uint8 DataFormatIdentifier, uint8 MemoryIdentifier, uint32 MemoryAddress, uint32 MemorySize, uint32* BlockLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	





Service ID [hex]	0x31	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL DCM_PENDING DCM_CANCEL DCM_FORCE_RCRRP_OK
	DataFormatIdentifier	Bit 7 - 4: Compression Method • 0x0: not compressed • 0x1..F: vehicle-manufacturer-specific Bit 3 - 0: Encrypting method • 0x0: not encrypted • 0x1..F: vehicle-manufacturer-specific
	MemoryIdentifier	Identifier of the Memory Block, if the parameter is not used it shall be set to 0.
	MemoryAddress	Starting address of server memory from which data are to be copied
	MemorySize	Uncompressed memory size in bytes
Parameters (inout)	BlockLength	Max. Number of bytes for one Dcm_ReadMemory
Parameters (out)	ErrorCode	If the operation Dcm_ProcessRequestUpload returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful E_NOT_OK: Request was not successful DCM_E_PENDING: Request is not yet finished
Description	Callout function. DCM shall call this callout function to start an upload process. This service is needed for the implementation of UDS service RequestUpload.	
Available via	Dcm_Externals.h	

]

8.5.12.4 ProcessTransferDataRead

[SWS_Dcm_91070] Definition of configurable interface Dcm_ProcessTransferDataRead [

Service Name	Dcm_ProcessTransferDataRead	
Syntax	<pre> Dcm_ReturnReadMemoryType Dcm_ProcessTransferDataRead (Dcm_OpStatusType OpStatus, uint8 MemoryIdentifier, uint32 MemoryAddress, uint32 MemorySize, Dcm_RequestDataArrayType MemoryData, Dcm_NegativeResponseCodeType* ErrorCode) </pre>	
Service ID [hex]	0x26	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL DCM_PENDING DCM_CANCEL DCM_FORCE_RCRRP_OK





	MemoryIdentifier	Identifier of the Memory Block (e.g. used if memory section distinguishing is needed) Note: If it's not used this parameter shall be set to 0.
	MemoryAddress	Starting address of server memory from which data is to be retrieved.
	MemorySize	Number of bytes in the MemoryData
Parameters (inout)	None	
Parameters (out)	MemoryData	Data read (Points to the diagnostic buffer in DCM)
	ErrorCode	If the operation Dcm_ReadMemory returns value DCM_READ_FAILED, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Dcm_ReturnReadMemoryType	DCM_READ_OK: read was successful DCM_READ_FAILED: read was not successful DCM_READ_PENDING: read is not yet finished DCM_READ_FORCE_RCRRP: reading is pending, the Response pending transmission starts immediately
Description	The ProcessTransferDataRead callout is used to request memory data identified by the parameter memoryAddress and memorySize from the UDS request message. This service is needed for the implementation of UDS services: • ReadMemoryByAddress • RequestUpload • ReadDataByIdentifier (in case of Dynamical DID defined by memory address) • TransferData	
Available via	Dcm_Externals.h	

└

8.5.12.5 ProcessTransferDataWrite

[SWS_Dcm_91071] Definition of configurable interface Dcm_ProcessTransferDataWrite

Service Name	Dcm_ProcessTransferDataWrite	
Syntax	<pre>Dcm_ReturnWriteMemoryType Dcm_ProcessTransferDataWrite (Dcm_OpStatusType OpStatus, uint8 MemoryIdentifier, uint32 MemoryAddress, uint32 MemorySize, const Dcm_RequestDataArrayType MemoryData, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x7b	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL DCM_PENDING DCM_CANCEL DCM_FORCE_RCRRP_OK
	MemoryIdentifier	Identifier of the Memory Block (e.g. used by WriteDataByIdentifier service). Note: If it's not used this parameter shall be set to 0.
	MemoryAddress	Starting address of server memory in which data is to be copied. Note: If it's not used (e.g. if the data is compressed) this parameter shall be set to 0.





	MemorySize	Number of bytes in MemoryData
	MemoryData	Data to write (Points to the diagnostic buffer in DCM)
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Dcm_WriteMemory returns value DCM_WRITE_FAILED, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Dcm_ReturnWriteMemoryType	DCM_WRITE_OK: write was successful DCM_WRITE_FAILED: write was not successful DCM_WRITE_PENDING: write is not yet finished DCM_WRITE_FORCE_RCRP: writing is pending, the Response pending transmission starts immediately
Description	The ProcessTransferDataWrite callout is used to write memory data identified by the parameter memoryAddress and memorySize. This service is needed for the implementation of UDS services : • WriteMemoryByAddress • RequestDownload • TransferData	
Available via	Dcm_Externals.h	

]

8.6 Scheduled functions

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

8.6.1 Dcm_MainFunction

[SWS_Dcm_00053] Definition of scheduled function Dcm_MainFunction

Upstream requirements: [SRS_BSW_00424](#), [SRS_BSW_00373](#)

[

Service Name	Dcm_MainFunction
Syntax	void Dcm_MainFunction (void)
Service ID [hex]	0x25
Description	This service is used for processing the tasks of the main loop.
Available via	SchM_Dcm.h

]

8.7 Expected interfaces

In this chapter all interfaces required from other modules are listed.

8.7.1 Mandatory interfaces

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

[SWS_Dcm_91001] Definition of mandatory interfaces required by module Dcm

[

API Function	Header File	Description
ComM_DCM_ActiveDiagnostic	ComM_Dcm.h	Indication of active diagnostic by the DCM.
ComM_DCM_InactiveDiagnostic	ComM_Dcm.h	Indication of inactive diagnostic by the DCM.
PduR_DcmTransmit	PduR_Dcm.h	Requests transmission of a PDU.

]

8.7.2 Optional interfaces

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

[SWS_Dcm_91002] Definition of optional interfaces requested by module Dcm

API Function	Header File	Description
BndM_GetBlockPtr_<Block Id.Shortname>	BndM_Externals.h	–
BndM_WriteBlock_<Block Id.Shortname>	BndM_Externals.h	Will persist the data in flash, that it can later directly accessed via BndM_GetBlockPtr API. The writing take a while and is finished after the successful callback xxx_BndMWriteBlockFinish
BndM_WriteCancel	BndM.h	Cancels the writing
BndM_WriteFinalize	BndM.h	Will trigger the finalization of writing phase. The finish of asynchronous processing will trigger the callback xxx_BndMWriteFinalizeFinish including the result of this operation.
BndM_WriteStart	BndM.h	Will trigger the start of writing phase. The finish of asynchronous processing will trigger the callback xxx_BndMWriteStartFinish including the result of this operation
BswM_Dcm_ApplicationUpdated	BswM_Dcm.h	This function is called by the DCM in order to report an updated application.
BswM_Dcm_Communication Mode_CurrentState	BswM_Dcm.h	Function called by DCM to inform the BswM about the current state of the communication mode.
Csm_RandomGenerate	Csm.h	Generate a random number and stores it in the memory location pointed by the result pointer.
Csm_SignatureGenerate	Csm.h	Uses the given data to perform the signature calculation and stores the signature in the memory location pointed by the result pointer.
Csm_SignatureVerify	Csm.h	Verifies the given signature by checking if it was generated with the given data.
Dem_ClearDTC	Dem.h	Clears single DTCs, as well as groups of DTCs.





API Function	Header File	Description
Dem_DcmGetAvailableOBDMIDs	Dem_Dcm.h	Reports the value of a requested "availability-OBDMID" to the DCM upon a Service \$06 request. Derived from that the tester displays the supported tests a mechanic can select from. API is needed in OBD-relevant ECUs only. API Availability: This API will be available only if $\{ \{ \text{ecuc}(\text{Dem}/\text{DemGeneral.DemOBDSupport}) \} \neq \text{DEM\_OBD\_NO\_OBD\_SUPPORT} \}$
Dem_DcmGetDTCOfOBDFreeze Frame	Dem_Dcm.h	Gets DTC by freeze frame record number. API is needed in OBD-relevant ECUs only. API Availability: This API will be available only if $\{ \{ \text{ecuc}(\text{Dem}/\text{DemGeneral.DemOBDSupport}) \} \neq \text{DEM\_OBD\_NO\_OBD\_SUPPORT} \}$
Dem_DcmGetDTRData	Dem_Dcm.h	Reports a DTR data along with TID-value, UaSID, test result with lower and upper limit. API is needed in OBD-relevant ECUs only. API Availability: This API will be available only if $\{ \{ \text{ecuc}(\text{Dem}/\text{DemGeneral.DemOBDSupport}) \} \neq \text{DEM\_OBD\_NO\_OBD\_SUPPORT} \}$
Dem_DcmGetNumTIDsOfOBDMID	Dem_Dcm.h	Gets the number of TIDs per (functional) OBDMID. This can be used by the DCM to iteratively request for OBD/TID result data within a loop from 0....numberOfTIDs-1 API is needed in OBD-relevant ECUs only. API Availability: This API will be available only if $\{ \{ \text{ecuc}(\text{Dem}/\text{DemGeneral.DemOBDSupport}) \} \neq \text{DEM\_OBD\_NO\_OBD\_SUPPORT} \}$
Dem_DcmReadDataOfOBDFreeze Frame	Dem_Dcm.h	Gets data element per PID and index of the most important freeze frame being selected for the output of service \$02. The function stores the data in the provided DestBuffer. API is needed in OBD-relevant ECUs only. API Availability: This API will be available only if $\{ \{ \text{ecuc}(\text{Dem}/\text{DemGeneral.DemOBDSupport}) \} \neq \text{DEM\_OBD\_NO\_OBD\_SUPPORT} \}$
Dem_DisableDTCRecordUpdate	Dem.h	Disables the event memory update of a specific DTC (only one at one time).
Dem_DisableDTCSetting	Dem.h	Disables the DTC setting for all DTCs assigned to the DemEventMemorySet of the addressed client.
Dem_EnableDTCRecordUpdate	Dem.h	Enables the event memory update of the DTC disabled by Dem_DisableDTCRecordUpdate() before. Note: As the call to Dem_EnableDTCRecordUpdate is the last action in the sequence of disabling/enabling a DTC, the caller is not interested in the return value. Therefore E_OK should be returned even if the DTC is enabled asynchronously.
Dem_EnableDTCSetting	Dem.h	(Re)-Enables the DTC setting for all DTCs assigned to the DemEventMemorySet of the addressed client.
Dem_GetDTCByOccurrenceTime	Dem.h	Gets the DTC by occurrence time. There is no explicit parameter for the DTC-origin as the origin always is DEM_DTC_ORIGIN_PRIMARY_MEMORY.
Dem_GetDTCSeverityAvailabilityMask	Dem.h	Gets the DTC Severity availability mask.
Dem_GetDTCStatusAvailabilityMask	Dem.h	Gets the DTC Status availability mask of the selected fault memory.
Dem_GetFunctionalUnitOfDTC	Dem.h	Gets the functional unit of the requested DTC.





API Function	Header File	Description
Dem_GetNextExtendedDataRecord	Dem.h	Gets extended data record for the DTC selected by Dem_SelectExtendedDataRecord. The function stores the data in the provided DestBuffer.
Dem_GetNextFilteredDTC	Dem.h	Gets the next filtered DTC matching the filter criteria. For UDS services, the interface has an asynchronous behavior, because a large number of DTCs has to be processed.
Dem_GetNextFilteredDTCAndFDC	Dem.h	Gets the next filtered DTC and its associated Fault Detection Counter (FDC) matching the filter criteria. The interface has an asynchronous behavior, because a large number of DTCs has to be processed and the FDC might be received asynchronously from a SW-C, too.
Dem_GetNextFilteredDTCAndSeverity	Dem.h	Gets the next filtered DTC and its associated Severity matching the filter criteria. The interface has an asynchronous behavior, because a large number of DTCs has to be processed.
Dem_GetNextFilteredRecord	Dem.h	Gets the next freeze frame record number and its associated DTC stored in the event memory. The interface has an asynchronous behavior, because NvRAM access might be required.
Dem_GetNextFreezeFrameData	Dem.h	Gets freeze frame data by the DTC selected by Dem_SelectFreezeFrameData. The function stores the data in the provided DestBuffer.
Dem_GetNumberOfFilteredDTC	Dem.h	Gets the number of a filtered DTC.
Dem_GetNumberOfFreezeFrame Records	Dem.h	This function returns the number of all freeze frame records currently stored in the primary event memory
Dem_GetSeverityOfDTC	Dem.h	Gets the severity of the requested DTC. For large configurations and DTC-calibration, the interface behavior can be asynchronous (splitting the DTC-search into segments).
Dem_GetSizeOfExtendedDataRecord Selection	Dem.h	Gets the size of Extended Data Record by DTC selected by the call of Dem_SelectExtendedData Record.
Dem_GetSizeOfFreezeFrameSelection	Dem.h	Gets the size of freeze frame data by DTC selected by the call of Dem_SelectFreezeFrameData.
Dem_GetStatusOfDTC	Dem.h	Gets the status of a DTC. For large configurations and DTC-calibration, the interface behavior can be asynchronous (splitting the DTC-search into segments). The DTCs of OBD Events Suppression shall be reported as Dem_WRONG_DTC.
Dem_GetTranslationType	Dem.h	Gets the supported DTC formats of the ECU. The supported formats are configured via DemTypeOf DTCSupported.





API Function	Header File	Description
Dem_SetDTCFilter	Dem.h	Sets the DTC Filter. The server shall perform a bit-wise logical AND-ing operation between the parameter DTCStatusMask and the current UDS status in the server. In addition to the DTCStatusAvailabilityMask, the server shall return all DTCs for which the result of the AND-ing operation is non-zero [i.e. (statusOfDTC & DTCStatusMask) != 0]. The server shall process only the DTC Status bits that it is supporting. OBD Events Suppression shall be ignored for this computation. If no DTCs within the server match the masking criteria specified in the client's request, no DTC or status information shall be provided following the DTCStatusAvailabilityMask byte in the positive response message (((statusOfDTC & DTCStatusMask) != 0) && ((severity & DTCSeverityMask) != 0)) == TRUE
Dem_SetFreezeFrameRecordFilter	Dem.h	Sets a freeze frame record filter.
Det_ReportError	Det.h	Service to report development errors.
IdsM_ReportSecurityEvent	IdsM.h	This API is the interface to report security events to the IdsM.
IoHwAb_Dcm_<EcuSignalName>	IoHwAb_Dcm.h	This function provides control access to a certain ECU Signal to the DCM module (<EcuSignalname> is the symbolic name of an ECU Signal). The ECU signal can be locked and unlocked by this function. Locking 'freezes' the ECU signal to the current value, the configured default value or a value given by the parameter 'signal'.
IoHwAb_Dcm_Read<EcuSignal Name>	IoHwAb_Dcm.h	This function provides read access to a certain ECU Signal to the DCM module (<EcuSignalname> is the symbolic name of an ECU Signal).
KeyM_CertElementGetFirst	KeyM.h	This function is used to initialize the interactive extraction of a certificate data element. It always retrieves the top element from the configured certificate element and initializes the structure Key M_CertElementIterator so that consecutive data from this element can be read with KeyM_CertElementGetNext().
KeyM_CertElementGetNext	KeyM.h	This function provides further data from a certificate element, e.g. if a set of data are located in one certificate element that shall be read one after another. This function can only be called if the function KeyM_CertElementGetFirst() has been called once before.
KeyM_GetCertificate	KeyM.h	This function provides the DER encoded certificate data
KeyM_SetCertificate	KeyM.h	This function provides the certificate data to the key management module to temporarily store the certificate.
KeyM_VerifyCertificate	KeyM.h	This function verifies a certificate that was previously provided with KeyM_SetCertificate() against already stored and provided certificates stored with other certificate IDs.
NvM_ReadBlock	NvM.h	Service to copy the data of the NV block to its corresponding RAM block.
NvM_SetRamBlockStatus	NvM.h	Service for setting the RAM block status of a permanent RAM block or the status of the explicit synchronization of a NVRAM block.





API Function	Header File	Description
NvM_WriteBlock	NvM.h	Service to copy the data of the RAM block to its corresponding NV block.
PduR_DcmCancelReceive	PduR_Dcm.h	Requests cancellation of an ongoing reception of a PDU in a lower layer transport protocol module.
PduR_DcmCancelTransmit	PduR_Dcm.h	Requests cancellation of an ongoing transmission of a PDU in a lower layer communication module.
SchM_ActMainFunction_Dcm	<none>	Invokes the SchM_ActMainFunction function to trigger the activation of a corresponding main processing function.

]

Dem_DcmReadDataOfOBDFreezeFrame is only required when OBD Service \$02 is configured (see configuration parameter [DcmDsdSidTabServiceId](#)).

8.7.3 Configurable interfaces

This section defines the interfaces where the [Dcm](#) configuration defines the actual functions that the [Dcm](#) will use. Depending on the configuration, an implementation of these functions could be provided by other BSW-modules (typically the DEM) or by software-components (through the RTE).

8.7.3.1 SecurityAccess

From the point of view of the DCM, the operation has the following signature:

8.7.3.1.1 GetSeed

If [DcmDspSecurityADRSIZE](#) is present:

[SWS_Dcm_01151] Definition of configurable interface Xxx_GetSeed [

Service Name	Xxx_GetSeed	
Syntax	<pre>Std_ReturnType Xxx_GetSeed (const uint8* SecurityAccessDataRecord, Dcm_OpStatusType OpStatus, uint8* Seed, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x8b	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SecurityAccessData Record	This data record contains additional data to calculate the seed value; the size of this parameter is DcmDspSecurityADRSIZE which is at least "1".





	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	Seed	Pointer for provided seed
	ErrorCode	If the operation Xxx_GetSeed returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	Request to application for asynchronous provision of seed value	
Available via	Dcm_Externals.h	

]

If `DcmDspSecurityADRSIZE` is not present:

[SWS_Dcm_91003] Definition of configurable interface Xxx_GetSeed [

Service Name	Xxx_GetSeed	
Syntax	<pre>Std_ReturnType Xxx_GetSeed (Dcm_OpStatusType OpStatus, uint8* Seed, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x8a	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	Seed	Pointer for provided seed
	ErrorCode	If the operation Xxx_GetSeed returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	Request to application for asynchronous provision of seed value	
Available via	Dcm_Externals.h	

]

8.7.3.1.2 CompareKey

[SWS_Dcm_91004] Definition of configurable interface Xxx_CompareKey [

Service Name	Xxx_CompareKey	
Syntax	<pre>Std_ReturnType Xxx_CompareKey (const uint8* Key, Dcm_OpStatusType OpStatus, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x47	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Key	Key, which needs to be compared
	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	ErrorCode	NRC to be sent if E_NOT_OK is returned
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish. DCM_E_COMPARE_KEY_FAILED: Key did not match.
Description	Request to application for asynchronous comparing key (DcmDspSecurityUsePort = USE_ASYNC_CLIENT_SERVER)	
Available via	Dcm_Externals.h	

]

8.7.3.1.3 GetSecurityAttemptCounter

[SWS_Dcm_01152] Definition of configurable interface Xxx_GetSecurityAttemptCounter [

Service Name	Xxx_GetSecurityAttemptCounter	
Syntax	<pre>Std_ReturnType Xxx_GetSecurityAttemptCounter (Dcm_OpStatusType OpStatus, uint8* AttemptCounter)</pre>	
Service ID [hex]	0x59	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL DCM_PENDING DCM_CANCEL
Parameters (inout)	None	
Parameters (out)	AttemptCounter	The attempt counter for this security level
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	Read the attempt counter for a specific security level from the application	
Available via	Dcm_Externals.h	

]

Note: In case the Security Access AttemptCounter needs to be shared between application and bootloader, the application needs to consider this in the API-call [Xxx_GetSecurityAttemptCounter](#) (see chapter 7.4.4 Interaction). Further this has also impact on the security delay timer which needs to be considered.

8.7.3.1.4 SetSecurityAttemptCounter

[SWS_Dcm_01153] Definition of configurable interface **Xxx_SetSecurityAttemptCounter** [

Service Name	Xxx_SetSecurityAttemptCounter	
Syntax	<pre>Std_ReturnType Xxx_SetSecurityAttemptCounter (Dcm_OpStatusType OpStatus, uint8 AttemptCounter)</pre>	
Service ID [hex]	0x5a	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL DCM_PENDING DCM_CANCEL
	AttemptCounter	The attempt counter for this security level
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	Set the attempt counter for a specific security level in the application	
Available via	Dcm_Externals.h	

]

8.7.3.2 DataServices

From the point of view of the DCM, the operations have the following signatures:

Note : The OpStatus parameter exist only for asynchronous operations (if [DcmDspDataUsePort](#) is set to [USE_DATA_ASYNC_CLIENT_SERVER](#) or [USE_DATA_ASYNC_CLIENT_SERVER_ERROR](#) or [USE_DATA_ASYNC_FNC](#) or [USE_DATA_ASYNC_FNC_ERROR](#) or [USE_DATA_ASYNC_FNC_PROXY](#)). In case of synchronous operations ([DcmDspDataUsePort](#) is set to [USE_DATA_SYNC_CLIENT_SERVER](#) or [USE_DATA_SYNC_FNC](#) or [USE_DATA_ASYNC_FNC_PROXY](#)), the OpStatus parameter does not exist.

8.7.3.2.1 ReadData

If [DcmDspDataUsePort](#) is set to [USE_DATA_SYNC_CLIENT_SERVER](#) or [USE_DATA_SYNC_FNC](#), the following definition is used:

[SWS_Dcm_00793] Definition of configurable interface Xxx_ReadData [

Service Name	Xxx_ReadData	
Syntax	<pre>Std_ReturnType Xxx_ReadData (uint8* Data)</pre>	
Service ID [hex]	0x34	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	Data	Buffer where the requested data shall be copied to
Return value	Std_ReturnType	E_OK: this value is always returned.
Description	This function requests to the application a data value of a DID/PID if DcmDspDataUsePort is set to USE_DATA_SYNCH_CLIENT_SERVER.	
Available via	Dcm_Externals.h	

]

If `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_CLIENT_SERVER` or `USE_DATA_ASYNC_FNC`, the following definition is used:

[SWS_Dcm_91006] Definition of configurable interface Xxx_ReadData [

Service Name	Xxx_ReadData	
Syntax	<pre>Std_ReturnType Xxx_ReadData (Dcm_OpStatusType OpStatus, uint8* Data)</pre>	
Service ID [hex]	0x3b	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	Data	Buffer where the requested data shall be copied to
Return value	Std_ReturnType	E_OK: Request was successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests to the application a data value of a DID/PID if DcmDspDataUsePort is set to USE_DATA_ASYNC_CLIENT_SERVER.	
Available via	Dcm_Externals.h	

]

If `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_CLIENT_SERVER_ERROR` or `USE_DATA_ASYNC_FNC_ERROR`, the following definition is used:

[SWS_Dcm_91005] Definition of configurable interface Xxx_ReadData [

Service Name	Xxx_ReadData	
Syntax	<pre>Std_ReturnType Xxx_ReadData (Dcm_OpStatusType OpStatus, uint8* Data, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x58	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	Data	Buffer where the requested data shall be copied to
	ErrorCode	If the operation Xxx_ReadData returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests to the application a data value of a DID/PID if DcmDspDataUsePort is set to USE_DATA_ASYNC_CLIENT_SERVER.	
Available via	Dcm_Externals.h	

]

If `DcmDspDataUsePort` is set to `USE_DATA_SYNC_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91090] Definition of configurable interface Xxx_ReadData

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

[

Service Name	Xxx_ReadData (draft)	
Syntax	<pre>Std_ReturnType Xxx_ReadData (uint8* Data, uint16 DataLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x68	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	DataLength	Length in byte of the data to be read
Parameters (inout)	None	
Parameters (out)	Data	Buffer where the requested data shall be copied to
	ErrorCode	If the operation Xxx_ReadData returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.





Description	This function requests to the application a data value of a DID/PID if DcmDspDataUsePort is set to USE_DATA_SYNC_FNC_PROXY. Tags: atp.Status=draft
Available via	Dcm_Externals.h

]

If `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91091] Definition of configurable interface Xxx_ReadData

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

[

Service Name	Xxx_ReadData (draft)	
Syntax	<pre>Std_ReturnType Xxx_ReadData (Dcm_OpStatusType OpStatus, uint8* Data, uint16 DataLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x7d	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	Status of the current operation
	DataLength	Length in byte of the data to be read
Parameters (inout)	None	
Parameters (out)	Data	Buffer where the requested data shall be copied to
	ErrorCode	If the operation Xxx_ReadData returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests to the application a data value of a DID/PID if DcmDspDataUsePort is set to USE_DATA_ASYNC_FNC_PROXY.. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

]

8.7.3.2.2 WriteData

If `DcmDspDataUsePort` is set to `USE_DATA_SYNC_CLIENT_SERVER` or `USE_DATA_SYNC_FNC`, the following definition is used:

If `DcmDspDataType` is NOT set to `UINT8_DYN`, the following definition is used:

[SWS_Dcm_00794] Definition of configurable interface Xxx_WriteData [

Service Name	Xxx_WriteData	
Syntax	<pre>Std_ReturnType Xxx_WriteData (const uint8* Data, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x51	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Data	Buffer containing the data to be written
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_WriteData returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests the application to write a data value of a DID.	
Available via	Dcm_Externals.h	

]

If `DcmDspDataType` is set to `UINT8_DYN`, the following definition is used:

[SWS_Dcm_91007] Definition of configurable interface Xxx_WriteData [

Service Name	Xxx_WriteData	
Syntax	<pre>Std_ReturnType Xxx_WriteData (const uint8* Data, uint16 DataLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x52	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Data	Buffer containing the data to be written
	DataLength	Length in byte of the data to be written
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_WriteData returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests the application to write a data value of a DID.	
Available via	Dcm_Externals.h	

]

If `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_CLIENT_SERVER` or `USE_DATA_ASYNC_FNC` or `USE_DATA_ASYNC_CLIENT_SERVER_ERROR` or `USE_DATA_ASYNC_FNC_ERROR`, the following definition is used:

If `DcmDspDataType` is NOT set to `UINT8_DYN`, the following definition is used:

[SWS_Dcm_91008] Definition of configurable interface Xxx_WriteData [

Service Name	Xxx_WriteData	
Syntax	<pre>Std_ReturnType Xxx_WriteData (const uint8* Data, Dcm_OpStatusType OpStatus, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x35	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Data	Buffer containing the data to be written
	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_WriteData returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests the application to write a data value of a DID.	
Available via	Dcm_Externals.h	

]

If [DcmDspDataType](#) is set to UINT8_DYN, the following definition is used:

[SWS_Dcm_91009] Definition of configurable interface Xxx_WriteData [

Service Name	Xxx_WriteData	
Syntax	<pre>Std_ReturnType Xxx_WriteData (const uint8* Data, uint16 DataLength, Dcm_OpStatusType OpStatus, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x3e	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Data	Buffer containing the data to be written
	DataLength	Length in byte of the data to be written
	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_WriteData returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests the application to write a data value of a DID.	
Available via	Dcm_Externals.h	

]

If `DcmDspDataType` is set to `USE_DATA_SYNCH_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91095] Definition of configurable interface `Xxx_WriteData`

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

Service Name	Xxx_WriteData (draft)	
Syntax	<pre>Std_ReturnType Xxx_WriteData (const uint8* Data, uint16 DataLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x85	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Data	Buffer containing the data to be written
	DataLength	Length in byte of the data to be written
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation <code>Xxx_WriteData</code> returns value <code>E_NOT_OK</code> , the DCM module shall send a negative response with NRC code equal to the parameter <code>ErrorCode</code> parameter value.
Return value	Std_ReturnType	<code>E_OK</code> : Request was successful. <code>E_NOT_OK</code> : Request was not successful.
Description	This function requests the application to write a data value of a DID if <code>DcmDspDataUsePort</code> is set to <code>USE_DATA_SYNCH_FNC_PROXY</code> . Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

If `DcmDspDataType` is set to `USE_DATA_ASYNCH_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91092] Definition of configurable interface `Xxx_WriteData`

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

Service Name	Xxx_WriteData (draft)	
Syntax	<pre>Std_ReturnType Xxx_WriteData (const uint8* Data, uint16 DataLength, Dcm_OpStatusType OpStatus, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x91	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Data	Buffer containing the data to be written





	DataLength	Length in byte of the data to be written
	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_WriteData returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests the application to write a data value of a DID if DcmDspDataUsePort is set to USE_DATA_ASYNC_FNC_PROXY. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

]

8.7.3.2.3 ReadDataLength

If `DcmDspDataUsePort` is set to `USE_DATA_SYNC_CLIENT_SERVER` or `USE_DATA_SYNC_FNC` or `USE_DATA_SYNC_FNC_PROXY`, the following definition is used:

[SWS_Dcm_00796] Definition of configurable interface Xxx_ReadDataLength [

Service Name	Xxx_ReadDataLength	
Syntax	Std_ReturnType Xxx_ReadDataLength (uint16* DataLength)	
Service ID [hex]	0x86	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	DataLength	Length in byte of the data to be read
Return value	Std_ReturnType	E_OK: this value is always returned.
Description	This function requests the application to return the data length in byte of a Data.	
Available via	Dcm_Externals.h	

]

Please note that the function definition according [SWS_Dcm_00796] will become obsolete and gets replaced by [SWS_Dcm_91096]

[SWS_Dcm_91096] Definition of configurable interface Xxx_ReadDataLength

Status: DRAFT

Service Name	Xxx_ReadDataLength (draft)	
Syntax	Std_ReturnType Xxx_ReadDataLength (uint16* DataLength)	
Service ID [hex]	0x36	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	DataLength	Length in byte of the data to be read
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests the application to return the data length in byte of a Data. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

If [DcmDspDataUsePort](#) is set to [USE_DATA_ASYNC_CLIENT_SERVER](#) or [USE_DATA_ASYNC_CLIENT_SERVER_ERROR](#) or [USE_DATA_ASYNC_FNC](#) or [USE_DATA_ASYNC_FNC_ERROR](#) or [USE_DATA_ASYNC_FNC_PROXY](#), the following definition is used:

[SWS_Dcm_91010] Definition of configurable interface Xxx_ReadDataLength

Service Name	Xxx_ReadDataLength	
Syntax	Std_ReturnType Xxx_ReadDataLength (Dcm_OpStatusType OpStatus, uint16* DataLength)	
Service ID [hex]	0x4c	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	DataLength	Length in byte of the data to be read
Return value	Std_ReturnType	E_OK: this value is always returned. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests the application to return the data length in byte of a Data.	
Available via	Dcm_Externals.h	

Please note that the function definition according [\[SWS_Dcm_91010\]](#) will become obsolete and gets replaced by [\[SWS_Dcm_91097\]](#)

[SWS_Dcm_91097] Definition of configurable interface Xxx_ReadDataLength

Status: DRAFT

Service Name	Xxx_ReadDataLength (draft)	
Syntax	<pre>Std_ReturnType Xxx_ReadDataLength (Dcm_OpStatusType OpStatus, uint16* DataLength)</pre>	
Service ID [hex]	0x8c	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	DataLength	Length in byte of the data to be read
Return value	Std_ReturnType	E_OK: this value is always returned. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests the application to return the data length in byte of a Data. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

8.7.3.2.4 ConditionCheckRead

If `DcmDspDataUsePort` is set to `USE_DATA_SYNCH_CLIENT_SERVER` or `USE_DATA_SYNCH_FNC` or `USE_DATA_SYNCH_FNC_PROXY`, the following definition is used:

[SWS_Dcm_00797] Definition of configurable interface Xxx_ConditionCheckRead

Service Name	Xxx_ConditionCheckRead	
Syntax	<pre>Std_ReturnType Xxx_ConditionCheckRead (Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x49	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation <code>Xxx_ConditionCheckRead</code> returns value <code>E_NOT_OK</code> , the DCM module shall send a negative response with NRC code equal to the parameter <code>ErrorCode</code> parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests to the application if the conditions to read the Data are correct.	
Available via	Dcm_Externals.h	

If `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_CLIENT_SERVER` or `USE_DATA_ASYNC_FNC` or `USE_DATA_ASYNC_CLIENT_SERVER_ERROR` or `USE_DATA_ASYNC_FNC_ERROR` or `USE_DATA_ASYNC_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91011] Definition of configurable interface Xxx_ConditionCheckRead

Service Name	Xxx_ConditionCheckRead	
Syntax	<pre>Std_ReturnType Xxx_ConditionCheckRead (Dcm_OpStatusType OpStatus, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x37	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation <code>Xxx_ConditionCheckRead</code> returns value <code>E_NOT_OK</code> , the DCM module shall send a negative response with NRC code equal to the parameter <code>ErrorCode</code> parameter value.
Return value	Std_ReturnType	<code>E_OK</code> : Request was successful. <code>E_NOT_OK</code> : Request was not successful. <code>DCM_E_PENDING</code> : Request is not yet finished. Further call(s) required to finish.
Description	This function requests to the application if the conditions to read the Data are correct.	
Available via	Dcm_Externals.h	

8.7.3.2.5 GetScalingInformation

This function requests to the application for the scaling information of a Data (scaling-Byte and scalingByteExtension).

If `DcmDspDataUsePort` is set to `USE_DATA_SYNC_CLIENT_SERVER` or `USE_DATA_SYNC_FNC`, the following definition is used:

[SWS_Dcm_00798] Definition of configurable interface Xxx_GetScalingInformation

Service Name	Xxx_GetScalingInformation	
Syntax	<pre>Std_ReturnType Xxx_GetScalingInformation (uint8* ScalingInfo, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x4b	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	





Parameters (inout)	None	
Parameters (out)	ScalingInfo	Scaling information (scalingByte and scalingByteExtension)
	ErrorCode	If the operation Xxx_GetScalingInformation returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests to the application for the scaling information of a Data.	
Available via	Dcm_Externals.h	

]

If `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_CLIENT_SERVER` or `USE_DATA_ASYNC_FNC` or `USE_DATA_ASYNC_CLIENT_SERVER_ERROR` or `USE_DATA_ASYNC_FNC_ERROR`, the following definition is used:

[SWS_Dcm_91012] Definition of configurable interface Xxx_GetScalingInformation

Service Name	Xxx_GetScalingInformation	
Syntax	<pre>Std_ReturnType Xxx_GetScalingInformation (Dcm_OpStatusType OpStatus, uint8* ScalingInfo, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x38	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	ScalingInfo	Scaling information (scalingByte and scalingByteExtension)
	ErrorCode	If the operation Xxx_GetScalingInformation returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests to the application for the scaling information of a Data.	
Available via	Dcm_Externals.h	

]

If `DcmDspDataUsePort` is set to `USE_DATA_SYNC_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91093] Definition of configurable interface Xxx_GetScalingInformation*Status:* DRAFT*Upstream requirements:* [RS_Diag_04254](#)

[

Service Name	Xxx_GetScalingInformation (draft)	
Syntax	<pre>Std_ReturnType Xxx_GetScalingInformation (uint8* ScalingInfo, uint8 ScalingInfoLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x90	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == (USE_DATA_SYNCH_FNC_PROXY) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataInfoRef->DcmDspDataScalingInfoSize)) != NULL))	
Parameters (in)	ScalingInfoLength	Length in byte of the scaling information to be read
Parameters (inout)	None	
Parameters (out)	ScalingInfo	Scaling information (scalingByte and scalingByteExtension)
	ErrorCode	If the operation Xxx_GetScalingInformation returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests to the application for the scaling information of a Data. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

]

If [DcmDspDataUsePort](#) is set to [USE_DATA_ASYNC_FNC_PROXY](#), the following definition is used:

[SWS_Dcm_91094] Definition of configurable interface Xxx_GetScalingInformation*Status:* DRAFT*Upstream requirements:* [RS_Diag_04254](#)

[

Service Name	Xxx_GetScalingInformation (draft)	
Syntax	<pre>Std_ReturnType Xxx_GetScalingInformation (Dcm_OpStatusType OpStatus, uint8* ScalingInfo, uint8 ScalingInfoLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x8f	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	





Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)}== (USE_DATA_ASYNC_FNC_PROXY) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataInfoRef->DcmDspDataScalingInfoSize)} != NULL))	
Parameters (in)	OpStatus	Status of the current operation
	ScalingInfoLength	Length in byte of the scaling information to be read
Parameters (inout)	None	
Parameters (out)	ScalingInfo	Scaling information (scalingByte and scalingByteExtension)
	ErrorCode	If the operation Xxx_GetScalingInformation returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests to the application for the scaling information of a Data. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

]

8.7.3.2.6 ReturnControlToECU

If `DcmDspDataUsePort` is set to `USE_DATA_SYNC_CLIENT_SERVER` or `USE_DATA_SYNC_FNC` or `USE_DATA_ASYNC_CLIENT_SERVER` or `USE_DATA_ASYNC_FNC` or `USE_DATA_ASYNC_CLIENT_SERVER_ERROR` or `USE_DATA_ASYNC_FNC_ERROR`, the following definition is used:

[SWS_Dcm_01285] Definition of configurable interface Xxx_ReturnControlToECU

Service Name	Xxx_ReturnControlToECU	
Syntax	<pre>Std_ReturnType Xxx_ReturnControlToECU ([Dcm_ControlMask_{DID}Type controlMask], [uint8* controlMask], Dcm_NegativeResponseType* ErrorCode)</pre>	
Service ID [hex]	0x4f	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)}== (USE_DATA_SYNC_FNC USE_DATA_ASYNC_FNC USE_DATA_ASYNC_FNC_ERROR USE_DATA_SYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreeze CurrentState)} == TRUE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)} == TRUE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE))	
Parameters (in)	controlMask	–
	Type	<code>Dcm_ControlMask_{DID}Type</code>





	Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)) == DCM_CONTROLMASK_EXTERNAL) && {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)) <= 0x04)}
	controlMask	–
	Type	uint8*
	Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == (USE_DATA_SYNCH_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)) == DCM_CONTROLMASK_EXTERNAL) && {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)) => 0x05)}
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_ReturnControlToECU returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests to the application to return control to ECU of an IOControl.	
Available via	Dcm_Externals.h	

]

Note: Square brackets [] indicate that an argument is optional.

If `DcmDspDataUsePort` is set to `USE_DATA_SYNCH_FNC_PROXY` or `USE_DATA_ASYNC_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91098] Definition of configurable interface Xxx_ReturnControlToECU

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

[

Service Name	Xxx_ReturnControlToECU (draft)	
Syntax	Std_ReturnType Xxx_ReturnControlToECU (uint8* controlMask, uint8 controlMaskLength, Dcm_NegativeResponseType ErrorCode)	
Service ID [hex]	0x8e	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == (USE_DATA_SYNCH_FNC_PROXY USE_DATA_ASYNC_FNC_PROXY)) && {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl)) != NULL)}	
Parameters (in)	controlMask	Control enable mask





	controlMaskLength	Control enable mask length in byte
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_ReturnControlToECU returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests to the application to return control to ECU of an IOControl. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

└

8.7.3.2.7 ResetToDefault

8.7.3.2.7.1 Synchronous interface

If `DcmDspDataUsePort` is set to `USE_DATA_SYNCH_CLIENT_SERVER` or `USE_DATA_SYNCH_FNC`, the following definition is used:

[SWS_Dcm_01286] Definition of configurable interface Xxx_ResetToDefault [

Service Name	Xxx_ResetToDefault	
Syntax	<pre>Std_ReturnType Xxx_ResetToDefault ([Dcm_ControlMask_{DID}Type controlMask], [uint8* controlMask], Dcm_NegativeResponseType* ErrorCode)</pre>	
Service ID [hex]	0x4d	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_SYNCH_FNC USE_DATA_SYNCH_CLIENT_SERVER)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)} == TRUE)	
Parameters (in)	controlMask	–
	Type	Dcm_ControlMask_{DID}Type
	Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL)&& ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04)
	controlMask	–
	Type	uint8*





	Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == USE_DATA_SYNCH_CLIENT_SERVER) && {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef -> DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)) == DCM_CONTROLMASK_EXTERNAL)&& {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)) == 0x05)}
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_ResetToDefault returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests to the application to reset an IOControl to default value.	
Available via	Dcm_Externals.h	

]

Note: Square brackets [] indicate that an argument is optional.

If `DcmDspDataUsePort` is set to `USE_DATA_SYNCH_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91099] Definition of configurable interface Xxx_ResetToDefault

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

[

Service Name	Xxx_ResetToDefault (draft)	
Syntax	<pre>Std_ReturnType Xxx_ResetToDefault (uint8* controlMask, uint8 controlMaskLength, Dcm_NegativeResponseType* ErrorCode)</pre>	
Service ID [hex]	0x8d	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == USE_DATA_SYNCH_FNC_PROXY) && {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)) == TRUE}	
Parameters (in)	controlMask	Control enable mask
	controlMaskLength	Control enable mask length in byte
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_ResetToDefault returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests to the application to reset an IOControl to default value. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

]

8.7.3.2.7.2 Asynchronous interface

If `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_CLIENT_SERVER` or `USE_DATA_ASYNC_FNC` or `USE_DATA_ASYNC_CLIENT_SERVER_ERROR` or `USE_DATA_ASYNC_FNC_ERROR`, the following definition is used:

[SWS_Dcm_01314] Definition of configurable interface `Xxx_ResetToDefault` [

Service Name	<code>Xxx_ResetToDefault</code>	
Syntax	<pre>Std_ReturnType Xxx_ResetToDefault (Dcm_OpStatusType OpStatus, [Dcm_ControlMask_{DID}Type controlMask], [uint8* controlMask], Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x3c	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Variation	$((\text{ecuc}(\text{Dcm}/\text{DcmConfigSet}/\text{DcmDsp}/\text{DcmDspData}.\text{DcmDspDataUsePort})) == (\text{USE\_DATA\_ASYNC\_FNC} \text{USE\_DATA\_ASYNC\_FNC\_ERROR} \text{USE\_DATA\_ASYNC\_CLIENT\_SERVER} \text{USE\_DATA\_ASYNC\_CLIENT\_SERVER\_ERROR})) \&\& ((\text{ecuc}(\text{Dcm}/\text{DcmConfigSet}/\text{DcmDsp}/\text{DcmDspDidInfo}/\text{DcmDspDidControl}/\text{DcmDspDidResetToDefault})) == \text{TRUE})$	
Parameters (in)	<code>OpStatus</code>	Status of the current operation
	<code>controlMask</code>	–
	Type	<code>Dcm_ControlMask_{DID}Type</code>
	Variation	$((\text{ecuc}(\text{Dcm}/\text{DcmConfigSet}/\text{DcmDsp}/\text{DcmDspDid}/\text{DcmDspDidInfoRef} \rightarrow \text{DcmDspDidInfo}/\text{DcmDspDidControl}/\text{DcmDspDidControlMask})) == \text{DCM\_CONTROLMASK\_EXTERNAL}) \&\& ((\text{ecuc}(\text{Dcm}/\text{DcmConfigSet}/\text{DcmDsp}/\text{DcmDspDid}/\text{DcmDspDidInfoRef} \rightarrow \text{DcmDspDidInfo}/\text{DcmDspDidControl}/\text{DcmDspDidControlMaskSize})) \leq 0x04)$
	<code>controlMask</code>	–
	Type	<code>uint8*</code>
	Variation	$((\text{ecuc}(\text{Dcm}/\text{DcmConfigSet}/\text{DcmDsp}/\text{DcmDspData}.\text{DcmDspDataUsePort})) == (\text{USE\_DATA\_ASYNC\_CLIENT\_SERVER} \text{USE\_DATA\_ASYNC\_CLIENT\_SERVER\_ERROR})) \&\& ((\text{ecuc}(\text{Dcm}/\text{DcmConfigSet}/\text{DcmDsp}/\text{DcmDspDid}/\text{DcmDspDidInfoRef} \rightarrow \text{DcmDspDidInfo}/\text{DcmDspDidControl}/\text{DcmDspDidControlMask})) == \text{DCM\_CONTROLMASK\_EXTERNAL}) \&\& ((\text{ecuc}(\text{Dcm}/\text{DcmConfigSet}/\text{DcmDsp}/\text{DcmDspDid}/\text{DcmDspDidInfoRef} \rightarrow \text{DcmDspDidInfo}/\text{DcmDspDidControl}/\text{DcmDspDidControlMaskSize})) \Rightarrow 0x05)$
Parameters (inout)	None	
Parameters (out)	<code>ErrorCode</code>	If the operation <code>Xxx_ResetToDefault</code> returns value <code>E_NOT_OK</code> , the DCM module shall send a negative response with NRC code equal to the parameter <code>ErrorCode</code> parameter value.
Return value	<code>Std_ReturnType</code>	<code>E_OK</code> : Request was successful. <code>E_NOT_OK</code> : Request was not successful. <code>DCM_E_PENDING</code> : Request is not yet finished. Further call(s) required to finish.
Description	This function requests to the application to reset an IOControl to default value.	
Available via	<code>Dcm_Externals.h</code>	

]

Note: Square brackets [] indicate that an argument is optional.

If `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91100] Definition of configurable interface Xxx_ResetToDefault

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

Service Name	Xxx_ResetToDefault (draft)	
Syntax	<pre>Std_ReturnType Xxx_ResetToDefault (Dcm_OpStatusType OpStatus, uint8* controlMask, uint8 controlMaskLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x7f	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)}== USE_DATA_ ASYNCH_FNC_PROXY) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)} == TRUE)	
Parameters (in)	OpStatus	Status of the current operation
	controlMask	Control enable mask
	controlMaskLength	Control enable mask length in byte
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_ResetToDefault returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests to the application to reset an IOControl to default value. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

8.7.3.2.8 FreezeCurrentState

8.7.3.2.8.1 Synchronous interface

If [DcmDspDataUsePort](#) is set to [USE_DATA_SYNCH_CLIENT_SERVER](#) or [USE_DATA_SYNCH_FNC](#), the following definition is used:

[SWS_Dcm_01290] Definition of configurable interface Xxx_FreezeCurrentState

Service Name	Xxx_FreezeCurrentState	
Syntax	<pre>Std_ReturnType Xxx_FreezeCurrentState ([Dcm_ControlMask_{DID}Type controlMask], [uint8* controlMask], Dcm_NegativeResponseCodeType* ErrorCode)</pre>	





Service ID [hex]	0x4a	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_FNC USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreezeCurrentState)} == TRUE)	
Parameters (in)	controlMask	–
	Type	Dcm_ControlMask_{DID}Type
	Variation	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef -> DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef -> DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04))
	controlMask	–
	Type	uint8*
	Variation	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef -> DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef -> DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} => 0x05))
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_FreezeCurrentState returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests to the application to freeze the current state of an IOControl.	
Available via	Dcm_Externals.h	

]

Note: Square brackets [] indicate that an argument is optional.

If [DcmDspDataUsePort](#) is set to [USE_DATA_SYNCH_FNC_PROXY](#), the following definition is used:

[SWS_Dcm_91101] Definition of configurable interface Xxx_FreezeCurrentState

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

[

Service Name	Xxx_FreezeCurrentState (draft)
Syntax	Std_ReturnType Xxx_FreezeCurrentState (uint8* controlMask, uint8 controlMaskLength, Dcm_NegativeResponseCodeType * ErrorCode)
Service ID [hex]	0x77
Sync/Async	Synchronous





Reentrancy	Non Reentrant	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)}== USE_DATA_SYNC_FNC_PROXY) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreezeCurrentState)} == TRUE)	
Parameters (in)	controlMask	Control enable mask
	controlMaskLength	Control enable mask length in byte
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_FreezeCurrentState returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests to the application to freeze the current state of an IOControl. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

]

8.7.3.2.8.2 Asynchronous interface

If `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_CLIENT_SERVER` or `USE_DATA_ASYNC_FNC` or `USE_DATA_ASYNC_CLIENT_SERVER_ERROR` or `USE_DATA_ASYNC_FNC_ERROR`, the following definition is used:

[SWS_Dcm_01315] Definition of configurable interface Xxx_FreezeCurrentState

[

Service Name	Xxx_FreezeCurrentState	
Syntax	<pre>Std_ReturnType Xxx_FreezeCurrentState (Dcm_OpStatusType OpStatus, [Dcm_ControlMask_{DID}Type controlMask], [uint8* controlMask], Dcm_NegativeResponseType* ErrorCode)</pre>	
Service ID [hex]	0x84	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ASYNC_FNC USE_DATA_ASYNC_FNC_ERROR USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreezeCurrentState)} == TRUE)	
Parameters (in)	OpStatus	Status of the current operation
	controlMask	–
	Type	<code>Dcm_ControlMask_{DID}Type</code>
	Variation	<pre>{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef -> DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef -> DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04)</pre>
	controlMask	–





	Type	uint8*
	Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == (USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef -> DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)) == DCM_CONTROLMASK_EXTERNAL}&& {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)) => 0x05)}
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_FreezeCurrentState returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests to the application to freeze the current state of an IOControl.	
Available via	Dcm_Externals.h	

]

If [DcmDspDataUsePort](#) is set to [USE_DATA_ASYNC_FNC_PROXY](#), the following definition is used:

[SWS_Dcm_91102] Definition of configurable interface Xxx_FreezeCurrentState

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

[

Service Name	Xxx_FreezeCurrentState (draft)	
Syntax	<pre>Std_ReturnType Xxx_FreezeCurrentState (Dcm_OpStatusType OpStatus, uint8* controlMask, uint8 controlMaskLength, Dcm_NegativeResponseType* ErrorCode)</pre>	
Service ID [hex]	0x88	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == USE_DATA_ASYNC_FNC_PROXY) && {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreezeCurrentState)) == TRUE)}	
Parameters (in)	OpStatus	Status of the current operation
	controlMask	Control enable mask
	controlMaskLength	Control enable mask length in byte
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_FreezeCurrentState returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.





Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests to the application to freeze the current state of an IOControl. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

└

8.7.3.2.9 ShortTermAdjustment

8.7.3.2.9.1 Synchronous interface

If `DcmDspDataUsePort` is set to `USE_DATA_SYNCH_CLIENT_SERVER` or `USE_DATA_SYNCH_FNC`, the following definition is used:

[SWS_Dcm_00802] Definition of configurable interface Xxx_ShortTermAdjustment ┌

Service Name	Xxx_ShortTermAdjustment	
Syntax	<pre>Std_ReturnType Xxx_ShortTermAdjustment (const uint8* ControlStateInfo, [uint16 DataLength], [Dcm_ControlMask_{DID}Type controlMask], [uint8* controlMask], Dcm_NegativeResponseType* ErrorCode)</pre>	
Service ID [hex]	0x50	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_FNC USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE)	
Parameters (in)	ControlStateInfo	ControlState information contained in the ControlOptionRecord parameter of the InputOutputControlByIdentifier diagnostic request
	DataLength	Length in byte of the data to be written
	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspData Type) == UINT8_DYN}
	controlMask	–
	Type	Dcm_ControlMask_{DID}Type
	Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef -> DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef -> DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04)
	controlMask	–
	Type	uint8*





	Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == USE_DATA_SYNCH_CLIENT_SERVER) && {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef -> DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)) == DCM_CONTROLMASK_EXTERNAL)&& {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)) >= 0x05)}
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_ShortTermAdjustment returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests to the application to adjust the IO signal.	
Available via	Dcm_Externals.h	

]

Note: Square brackets [] indicate that an argument is optional.

If `DcmDspDataUsePort` is set to `USE_DATA_SYNCH_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91103] Definition of configurable interface Xxx_ShortTermAdjustment

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

[

Service Name	Xxx_ShortTermAdjustment (draft)	
Syntax	<pre>Std_ReturnType Xxx_ShortTermAdjustment (const uint8* ControlStateInfo, uint16 DataLength, uint8* controlMask, uint8 controlMaskLength, Dcm_NegativeResponseType* ErrorCode)</pre>	
Service ID [hex]	0x87	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == USE_DATA_SYNCH_FNC_PROXY) && {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)) == TRUE}	
Parameters (in)	ControlStateInfo	ControlState information contained in the ControlOptionRecord parameter of the InputOutputControlByIdentifier diagnostic request
	DataLength	ControlState information length in byte
	controlMask	Control enable mask
	controlMaskLength	Control enable mask length in byte
Parameters (inout)	None	





Parameters (out)	ErrorCode	If the operation Xxx_ShortTermAdjustment returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function requests to the application to adjust the IO signal. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

└

8.7.3.2.9.2 Asynchronous interface

If `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_CLIENT_SERVER` or `USE_DATA_ASYNC_FNC` or `USE_DATA_ASYNC_CLIENT_SERVER_ERROR` or `USE_DATA_ASYNC_FNC_ERROR`, the following definition is used:

[SWS_Dcm_01316] Definition of configurable interface Xxx_ShortTermAdjustment

Service Name	Xxx_ShortTermAdjustment	
Syntax	<pre>Std_ReturnType Xxx_ShortTermAdjustment (const uint8* ControlStateInfo, [uint16 DataLength], Dcm_OpStatusType OpStatus, [Dcm_ControlMask_{DID}Type controlMask], [uint8* controlMask], Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x55	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Variation	$(\{ \text{ecuc}(\text{Dcm}/\text{DcmConfigSet}/\text{DcmDsp}/\text{DcmDspData}.\text{DcmDspDataUsePort}) == (\text{USE\_DATA\_ASYNC\_FNC} \parallel \text{USE\_DATA\_ASYNC\_FNC\_ERROR} \parallel \text{USE\_DATA\_ASYNC\_CLIENT\_SERVER} \parallel \text{USE\_DATA\_ASYNC\_CLIENT\_SERVER\_ERROR}) \} \& \& (\{ \text{ecuc}(\text{Dcm}/\text{DcmConfigSet}/\text{DcmDsp}/\text{DcmDspDidInfo}/\text{DcmDspDidControl}/\text{DcmDspDidShortTermAdjustment}) == \text{TRUE} \})$	
Parameters (in)	ControlStateInfo	ControlState information contained in the ControlOptionRecord parameter of the InputOutputControlByIdentifier diagnostic request
	DataLength	Length in byte of the data to be written
	Variation	$\{ \text{ecuc}(\text{Dcm}/\text{DcmConfigSet}/\text{DcmDsp}/\text{DcmDspData}.\text{DcmDspDataType}) == \text{UINT8\_DYN} \}$
	OpStatus	Status of the current operation
	controlMask	–
	Type	<code>Dcm_ControlMask_{DID}Type</code>
	Variation	$(\{ \text{ecuc}(\text{Dcm}/\text{DcmConfigSet}/\text{DcmDsp}/\text{DcmDspDid}/\text{DcmDspDidInfoRef} \rightarrow \text{DcmDspDidInfo}/\text{DcmDspDidControl}/\text{DcmDspDidControlMask}) == \text{DCM\_CONTROLMASK\_EXTERNAL} \} \& \& (\{ \text{ecuc}(\text{Dcm}/\text{DcmConfigSet}/\text{DcmDsp}/\text{DcmDspDid}/\text{DcmDspDidInfoRef} \rightarrow \text{DcmDspDidInfo}/\text{DcmDspDidControl}/\text{DcmDspDidControlMaskSize}) \leq 0x04 \})$





	controlMask	–
	Type	uint8*
	Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == (USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef -> DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)) == DCM_CONTROLMASK_EXTERNAL)&& ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)) >= 0x05)}
Parameters (inout)	None	
Parameters (out)	ErrorCode	NRC to be sent in the negative response in case of failure (E_NOT_OK)
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests to the application to adjust the IO signal.	
Available via	Dcm_Externals.h	

]

If `DcmDspDataUsePort` is set to `USE_DATA_ASYNC_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91104] Definition of configurable interface `Xxx_ShortTermAdjustment`

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

[

Service Name	Xxx_ShortTermAdjustment (draft)	
Syntax	<pre>Std_ReturnType Xxx_ShortTermAdjustment (const uint8* ControlStateInfo, uint16 DataLength, Dcm_OpStatusType OpStatus, uint8* controlMask, uint8 controlMaskLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x83	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == USE_DATA_ASYNC_FNC_PROXY) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)) == TRUE)}	
Parameters (in)	ControlStateInfo	ControlState information contained in the ControlOptionRecord parameter of the InputOutputControlByIdentifier diagnostic request
	DataLength	ControlState information length in byte
	OpStatus	Status of the current operation
	controlMask	Control enable mask
	controlMaskLength	Control enable mask length in byte





Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_ShortTermAdjustment returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests to the application to adjust the IO signal. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

]

8.7.3.3 DataServices_DIDRange

From the point of view of the DCM, the operations have the following signatures:

Note : The OpStatus parameter should only be used for asynchronous operations (if DcmDspDataUsePort is set to USE_DATA_ASYNC_CLIENT_SERVER or USE_DATA_ASYNC_FNC or USE_DATA_ASYNC_CLIENT_SERVER_ERROR or USE_DATA_ASYNC_FNC_ERROR). In case of synchronous operations (DcmDspDataUsePort is set to USE_DATA_SYNC_CLIENT_SERVER or USE_DATA_SYNC_FNC), the OpStatus parameter should not be used.

8.7.3.3.1 IsDidAvailable

[SWS_Dcm_00803] Definition of configurable interface Xxx_IsDidAvailable [

Service Name	Xxx_IsDidAvailable	
Syntax	Std_ReturnType Xxx_IsDidAvailable (uint16 DID, Dcm_OpStatusType OpStatus, Dcm_DidSupportedType* supported)	
Service ID [hex]	0x53	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	DID	DID value
	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	supported	Indicate if the DID is available within the range. Returning DCM_DID_SUPPORTED means it is supported within the range, Returning DCM_DID_NOT_SUPPORTED means it is not supported within the range
Return value	Std_ReturnType	E_OK: This value is returned when the Did is finally available. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.





Description	This function requests if a specific DID is available within the range or not.
Available via	Dcm_Externals.h

8.7.3.3.2 ReadDidData

[SWS_Dcm_00804] Definition of configurable interface Xxx_ReadDidData [

Service Name	Xxx_ReadDidData	
Syntax	<pre>Std_ReturnType Xxx_ReadDidData (uint16 DID, uint8* Data, Dcm_OpStatusType OpStatus, uint16 DataLength, Dcm_NegativeResponseCodeType ErrorCode)</pre>	
Service ID [hex]	0x89	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	DID	Data ID value
	OpStatus	Status of the current operation
Parameters (inout)	None	
Parameters (out)	Data	Buffer where the requested data shall be copied to
	DataLength	Length of the data to be read
	ErrorCode	If the operation Xxx_ReadDidData returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests to the application a data value of a DID	
Available via	Dcm_Externals.h	

8.7.3.3.3 WriteDidData

[SWS_Dcm_00805] Definition of configurable interface Xxx_WriteDidData [

Service Name	Xxx_WriteDidData	
Syntax	<pre>Std_ReturnType Xxx_WriteDidData (uint16 DID, const uint8* Data, Dcm_OpStatusType OpStatus, uint16 DataLength, Dcm_NegativeResponseType ErrorCode)</pre>	
Service ID [hex]	0x41	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	DID	Data ID value
	Data	Buffer containing the data to be written
	OpStatus	Status of the current operation
	DataLength	Length of the data to be written
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_WriteDidData returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests the application to write a data value of a DID.	
Available via	Dcm_Externals.h	

]

8.7.3.3.4 ReadDidRangeDataLength

ReadDidRangeDataLength requests the application to return the data length of a [DID](#) range. This interface is used for [UDS Service ReadDataByIdentifier](#).

[SWS_Dcm_01271] Definition of configurable interface Xxx_ReadDidRangeDataLength [

Service Name	Xxx_ReadDidRangeDataLength	
Syntax	<pre>Std_ReturnType Xxx_ReadDidRangeDataLength (uint16 DID, Dcm_OpStatusType OpStatus, uint16* DataLength)</pre>	
Service ID [hex]	0x5e	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	DID	Data ID value
	OpStatus	Status of the current operation





Parameters (inout)	None	
Parameters (out)	DataLength	Length of the data to be written/read
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	This function requests the application to return the data length of a DID Range.	
Available via	Dcm_Externals.h	

└

8.7.3.4 InfoTypesServices

8.7.3.4.1 GetInfotypeValueData

If DcmDspVehInfoDataSize is set to a value less than 256, the following definition based on [SWS_Dcm_91014] is used.

[SWS_Dcm_91014] Definition of configurable interface Xxx_GetInfotypeValueData for one byte data length ┌

Service Name	Xxx_GetInfotypeValueData	
Syntax	Std_ReturnType Xxx_GetInfotypeValueData (Dcm_OpStatusType OpStatus, uint8* DataValueBuffer, uint8* DataValueBufferSize)	
Service ID [hex]	0x60	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	Status of the current operation
Parameters (inout)	DataValueBufferSize	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The callee fills in the number of written data bytes in DataValueBuffer.
Parameters (out)	DataValueBuffer	Buffer containing the Infotype information
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	The function provides the data related to the requested Infotype.	
Available via	Dcm_Externals.h	

└

If DcmDspVehInfoDataSize is set to a value greater or equal than 256, the following definition based on SWS_Dcm_91126 is used:

**[SWS_Dcm_91126] Definition of configurable interface Xxx_GetInfotypeValue
Data for two byte data length [**

Service Name	Xxx_GetInfotypeValueData	
Syntax	<pre>Std_ReturnType Xxx_GetInfotypeValueData (Dcm_OpStatusType OpStatus, uint8* DataValueBuffer, uint16* DataValueBufferSize)</pre>	
Service ID [hex]	0x92	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	Status of the current operation
Parameters (inout)	DataValueBufferSize	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The callee fills in the number of written data bytes in DataValueBuffer.
Parameters (out)	DataValueBuffer	Buffer containing the Infotype information
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish.
Description	The function provides the data related to the requested Infotype.	
Available via	Dcm_Externals.h	

]

8.7.3.5 RoutineServices

The operations mentioned in the following sub-chapters are only general examples, because the number of In and OUT parameters can be variable from 0 to an arbitrary number. It is therefore not possible to list all variations of operation prototypes.

8.7.3.5.1 Xxx_Start Operation

If `DcmDspRoutineFncSignature` is set to `ROUTINE_FNC_NORMAL`, the following definition is used:

[SWS_Dcm_01203] Definition of configurable interface Xxx_Start [

Service Name	Xxx_Start	
Syntax	<pre>Std_ReturnType Xxx_Start ([DcmDspRoutineSignalType dataIn_1, ... DcmDspRoutineSignalType dataIn_n], [const uint8* dataInVar], Dcm_OpStatusType OpStatus, [DcmDspRoutineSignalType dataOut_1, ... DcmDspRoutineSignalType dataOut_n], [uint8* dataOutVar], [uint16* currentDataLength], Dcm_NegativeResponseCodeType ErrorCode)</pre>	





Service ID [hex]	0x5b	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	dataIn_1	Fixed-length input data provided in the routine control request
	...	...
	dataIn_n	Fixed-length input data provided in the routine control request
	dataInVar	Variable-length input data provided in the routine control request
	OpStatus	Status of the current operation
Parameters (inout)	currentDataLength	If variable length routine input data is used, this parameter contains the length in bytes of the dataInVar array. If variable length routine output data is used, this parameter contains the length in bytes of the dataOutVar parameter.
Parameters (out)	dataOut_1	Fixed-length output data to provide in the routine control response
	...	...
	dataOut_n	Fixed-length output data to provide in the routine control response
	dataOutVar	Variable-length output data to provide in the routine control response
	ErrorCode	If the operation Xxx_Start returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish. DCM_E_FORCE_RCRRP: application requests the transmission of a response Pending (NRC 0x78)
Description	This function requests to the application to start the execution of a routine.	
Available via	Dcm_Externals.h	

]

Note: Square brackets [] indicate that an argument is optional.

If `DcmDspRoutineFncSignature` is set to `ROUTINE_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91105] Definition of configurable interface Xxx_Start

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

[

Service Name	Xxx_Start (draft)
Syntax	<pre>Std_ReturnType Xxx_Start (Dcm_OpStatusType OpStatus, uint8* dataInOut, uint16* currentDataLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>
Service ID [hex]	0x80
Sync/Async	Asynchronous
Reentrancy	Non Reentrant





Parameters (in)	OpStatus	Status of the current operation
Parameters (inout)	dataInOut	Input and output data in the routine control request / response
	currentDataLength	This parameter contains the length in bytes of the dataInOut array. It include fixed length and variable length data.
Parameters (out)	ErrorCode	If the operation Xxx_Start returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish. DCM_E_FORCE_RCRRP: application requests the transmission of a response Pending (NRC 0x78)
Description	This function requests to the application to start the execution of a routine. Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

]

8.7.3.5.2 Xxx_StartConfirmation Operation

If `DcmDspRoutineFncSignature` is set to `ROUTINE_FNC_NORMAL` or `ROUTINE_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91016] Definition of configurable interface Xxx_StartConfirmation [

Service Name	Xxx_StartConfirmation	
Syntax	Std_ReturnType Xxx_StartConfirmation (Dcm_ConfirmationStatusType ConfirmationStatus)	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	ConfirmationStatus	Confirmation status of a StartRoutine request
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function indicates the transmission of a response to a StartRoutine request	
Available via	Dcm_Externals.h	

]

Note: Square brackets [] indicate that an argument is optional.

8.7.3.5.3 Xxx_Stop Operation

If `DcmDspRoutineFncSignature` is set to `ROUTINE_FNC_NORMAL`, the following definition is used:

[SWS_Dcm_01204] Definition of configurable interface Xxx_Stop [

Service Name	Xxx_Stop	
Syntax	<pre>Std_ReturnType Xxx_Stop ([DcmDspRoutineSignalType dataIn_1, ... DcmDspRoutineSignalType dataIn_n], [const uint8* dataInVar], Dcm_OpStatusType OpStatus, [DcmDspRoutineSignalType dataOut_1, ... DcmDspRoutineSignalType dataOut_n], [uint8* dataOutVar], [uint16* currentDataLength], Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x5c	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	dataIn_1	Fixed-length input data provided in the routine control request
	...	...
	dataIn_n	Fixed-length input data provided in the routine control request
	dataInVar	Variable-length input data provided in the routine control request
	OpStatus	Status of the current operation
Parameters (inout)	currentDataLength	If variable length routine input data is used, this parameter contains the length in bytes of the dataInVar array. If variable length routine output data is used, this parameter contains the length in bytes of the dataOutVar parameter.
Parameters (out)	dataOut_1	Fixed-length output data to provide in the routine control response
	...	...
	dataOut_n	Fixed-length output data to provide in the routine control response
	dataOutVar	Variable-length output data to provide in the routine control response
	ErrorCode	If the operation Xxx_Stop returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish DCM_E_FORCE_RCRRP: application requests the transmission of a response Pending (NRC 0x78)
Description	This function requests to the application to stop the execution of a routine	
Available via	Dcm_Externals.h	

]

Note: Square brackets [] indicate that an argument is optional.

If `DcmDspRoutineFncSignature` is set to `ROUTINE_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91106] Definition of configurable interface Xxx_Stop*Status:* DRAFT*Upstream requirements:* [RS_Diag_04254](#)

Service Name	Xxx_Stop (draft)	
Syntax	<pre>Std_ReturnType Xxx_Stop (Dcm_OpStatusType OpStatus, uint8* dataInOut, uint16* currentDataLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x81	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	Status of the current operation
Parameters (inout)	dataInOut	Input and output data in the routine control request / response
	currentDataLength	This parameter contains the length in bytes of the dataInOut array. It include fixed length and variable length data.
Parameters (out)	ErrorCode	If the operation Xxx_Stop returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish DCM_E_FORCE_RCRRP: application requests the transmission of a response Pending (NRC 0x78)
Description	This function requests to the application to stop the execution of a routine Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

8.7.3.5.4 Xxx_StopConfirmation Operation

If [DcmDspRoutineFncSignature](#) is set to [ROUTINE_FNC_NORMAL](#) or [ROUTINE_FNC_PROXY](#), the following definition is used:

[SWS_Dcm_91017] Definition of configurable interface Xxx_StopConfirmation

Service Name	Xxx_StopConfirmation	
Syntax	<pre>Std_ReturnType Xxx_StopConfirmation (Dcm_ConfirmationStatusType ConfirmationStatus)</pre>	
Service ID [hex]	0x69	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	ConfirmationStatus	Dcm_ConfirmationStatus Confirmation status of a StopRoutine request
Parameters (inout)	None	





Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function indicates the transmission of a response to a StopRoutine request	
Available via	Dcm_Externals.h	

]

Note: Square brackets [] indicate that an argument is optional.

8.7.3.5.5 Xxx_RequestResults Operation

If `DcmDspRoutineFncSignature` is set to `ROUTINE_FNC_NORMAL`, the following definition is used:

[SWS_Dcm_91013] Definition of configurable interface Xxx_RequestResults [

Service Name	Xxx_RequestResults	
Syntax	<pre>Std_ReturnType Xxx_RequestResults ([DcmDspRoutineSignalType* dataIn_1, ... DcmDspRoutineSignalType* dataIn_n], [const uint8* dataInVar], Dcm_OpStatusType OpStatus, [DcmDspRoutineSignalType* dataOut_1, ... DcmDspRoutineSignalType* dataOut_n], [uint8* dataOutVar], [uint16* currentDataLength], Dcm_NegativeResponseType* ErrorCode)</pre>	
Service ID [hex]	0x71	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	dataIn_1	Fixed-length input data provided in the routine control request
	...	...
	dataIn_n	Fixed-length input data provided in the routine control request
	dataInVar	Variable-length input data provided in the routine control request
	OpStatus	Move to be mentioned after dataInVar.
Parameters (inout)	currentDataLength	If variable length routine input data is used, this parameter contains the length in bytes of the dataInVar array. If variable length routine output data is used, this parameter contains the length in bytes of the dataOutVar parameter.
Parameters (out)	dataOut_1	Fixed-length Output data to provide in the routine control response
	...	...
	dataOut_n	Fixed-length Output data to provide in the routine control response
	dataOutVar	Variable-length Output data to provide in the routine control response





	ErrorCode	If the operation Xxx_RequestResults returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish DCM_E_FORCE_RCRP: application requests the transmission of a response Pending (NRC 0x78)
Description	This function requests to the application the result of a routine execution	
Available via	Dcm_Externals.h	

]

Note: Square brackets [] indicate that an argument is optional.

If `DcmDspRoutineFncSignature` is set to `ROUTINE_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91107] Definition of configurable interface Xxx_RequestResults

Status: DRAFT

Upstream requirements: [RS_Diag_04254](#)

[

Service Name	Xxx_RequestResults (draft)	
Syntax	<pre>Std_ReturnType Xxx_RequestResults (Dcm_OpStatusType OpStatus, uint8* dataInOut, uint16* currentDataLength, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x82	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	OpStatus	Status of the current operation
Parameters (inout)	dataInOut	Input and output data in the routine control request / response If variable length routine input data is used, this parameter contains the length in bytes of the dataInVar array. If variable length routine output data is used, this parameter contains the length in bytes of the dataOutVar parameter.
	currentDataLength	This parameter contains the length in bytes of the dataInOut array. It include fixed length and variable length data.
Parameters (out)	ErrorCode	If the operation Xxx_RequestResults returns value E_NOT_OK, the Dcm module shall send a negative response with N
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. DCM_E_PENDING: Request is not yet finished. Further call(s) required to finish DCM_E_FORCE_RCRP: application requests the transmission of a response Pending (NRC 0x78)
Description	This function requests to the application the result of a routine execution Tags: atp.Status=draft	
Available via	Dcm_Externals.h	

]

8.7.3.5.6 Xxx_RequestResultsConfirmation Operation

If `DcmDspRoutineFncSignature` is set to `ROUTINE_FNC_NORMAL` or `ROUTINE_FNC_PROXY`, the following definition is used:

[SWS_Dcm_91018] Definition of configurable interface Xxx_RequestResultsConfirmation [

Service Name	Xxx_RequestResultsConfirmation	
Syntax	<pre>Std_ReturnType Xxx_RequestResultsConfirmation (Dcm_ConfirmationStatusType ConfirmationStatus)</pre>	
Service ID [hex]	0x70	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	ConfirmationStatus	Confirmation status of a RequestRoutineResults request
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function indicates the transmission of a response to a RequestRoutineResults request	
Available via	Dcm_Externals.h	

]

Note: Square brackets [] indicate that an argument is optional.

8.7.3.6 RequestControlServices

From the point of view of the DCM, the operation has the following signature:

8.7.3.6.1 RequestControl callout

[SWS_Dcm_01338] Definition of configurable interface Xxx_RequestControl [

Service Name	Xxx_RequestControl	
Syntax	<pre>Std_ReturnType Xxx_RequestControl (uint8* OutBuffer, const uint8* InBuffer)</pre>	
Service ID [hex]	0x63	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	OutBuffer	Output buffer in which the RequestControl function can store its result
	InBuffer	Input buffer containing the data of the OBD Service 0x08 request
Parameters (out)	None	





Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	Invokes a TID-specific function taking a configured number of bytes as input and returning a fixed number of bytes as output. This is typically used to implement OBD Service \$08	
Available via	Dcm_Externals.h	

]

8.7.3.7 ServiceRequestNotification

From the point of view of the DCM, the operations has the following signatures:

8.7.3.7.1 Indication

[SWS_Dcm_01341] Definition of configurable interface Xxx_Indication [

Service Name	Xxx_Indication	
Syntax	<pre>Std_ReturnType Xxx_Indication (uint8 SID, const uint8* RequestData, uint32 DataSize, uint8 ReqType, uint16 ConnectionId, Dcm_NegativeResponseCodeType* ErrorCode, Dcm_ProtocolType ProtocolType, uint16 TesterSourceAddress)</pre>	
Service ID [hex]	0x65	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SID	–
	RequestData	Complete request data (diagnostic buffer), except the service ID
	DataSize	Number of valid bytes in the RequestData parameter
	ReqType	Addressing type of the request(0=physical request,1=functional request)
	ConnectionId	Unique connection identifier
	ProtocolType	Type of the protocol to be indicated
	TesterSourceAddress	source address of the tester
Parameters (inout)	None	
Parameters (out)	ErrorCode	If the operation Xxx_Indication returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful. E_REQUEST_NOT_ACCEPTED : Request not accepted
Description	This function indicates to the application that a service is about to be executed and allows the application to reject the execution of the service request	
Available via	Dcm_Externals.h	

]

8.7.3.7.2 Confirmation

[SWS_Dcm_01342] Definition of configurable interface Xxx_Confirmation [

Service Name	Xxx_Confirmation	
Syntax	<pre>Std_ReturnType Xxx_Confirmation (uint8 SID, uint8 ReqType, uint16 ConnectionId, Dcm_ConfirmationStatusType ConfirmationStatus, Dcm_ProtocolType ProtocolType, uint16 TesterSourceAddress)</pre>	
Service ID [hex]	0x66	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SID	Value of service identifier
	ReqType	Addressing type of the request(0=physical request,1=functional request)
	ConnectionId	Unique connection identifier
	ConfirmationStatus	Confirmation of a successful transmission or a transmission error of a diagnostic service.
	ProtocolType	Type of Dcm Protocol
	TesterSourceAddress	source address of the tester
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request was successful. E_NOT_OK: Request was not successful.
Description	This function confirms to the application the successful transmission or a transmission error of a diagnostic service.	
Available via	Dcm_Externals.h	

]

8.7.3.8 ClearDTCCheckFnc

From the point of view of the Dcm, the operation has the following signature:

[SWS_Dcm_01270] Definition of configurable interface Xxx_ClearDTCCheckFnc [

Service Name	Xxx_ClearDTCCheckFnc	
Syntax	<pre>Std_ReturnType Xxx_ClearDTCCheckFnc (uint32 GoDTC, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x5f	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	GoDTC	requested groupOfDTC
Parameters (inout)	None	





Parameters (out)	ErrorCode	If the operation Xxx_ClearDTCCheckFnc returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: application allows to clear the requested groupOfDTC E_NOT_OK: application does not allow to clear the requested groupOfDTC. Dcm shall send a negative response with the NRC returned in the ErrorCode
Description	Callout function for condition check, manufacturer / supplier specific checks on the groupOfDTC, which is requested to clear.	
Available via	Dcm_Externals.h	

]

8.8 Dcm as Service-Component

8.8.1 Implementation Data Types

8.8.1.1 Dcm_OpStatusType

[SWS_Dcm_00984] Definition of ImplementationDataType Dcm_OpStatusType [

Name	Dcm_OpStatusType		
Kind	Type		
Derived from	uint8		
Range	DCM_INITIAL	0x00	Indicates the initial call to the operation
	DCM_PENDING	0x01	Indicates that a pending return has been done on the previous call of the operation
	DCM_CANCEL	0x02	Indicates that the DCM requests to cancel the pending operation
	DCM_FORCE_RCRRP_OK	0x03	Confirm a response pending transmission
Description	–		
Variation	–		
Available via	Rte_Dcm_Type.h		

]

8.8.1.2 Dcm_ConfirmationStatusType

[SWS_Dcm_00983] Definition of ImplementationDataType Dcm_ConfirmationStatusType [

Name	Dcm_ConfirmationStatusType
Kind	Type
Derived from	uint8





Range	DCM_RES_POS_OK	0x00	–
	DCM_RES_POS_NOT_OK	0x01	–
	DCM_RES_NEG_OK	0x02	–
	DCM_RES_NEG_NOT_OK	0x03	–
Description	–		
Variation	–		
Available via	Rte_Dcm_Type.h		

]

8.8.1.3 Dcm_SecLevelType

[SWS_Dcm_00977] Definition of ImplementationDataType Dcm_SecLevelType [

Name	Dcm_SecLevelType		
Kind	Type		
Derived from	uint8		
Range	DCM_SEC_LEV_LOCKED	0x00	–
	configuration dependent	0x01...0x3F	–
	Reserved by Document	0x40...0xFF	–
Description	Security Level type definition		
Variation	–		
Available via	Rte_Dcm_Type.h		

]

8.8.1.4 Dcm_SesCtrlType

[SWS_Dcm_00978] Definition of ImplementationDataType Dcm_SesCtrlType [

Name	Dcm_SesCtrlType		
Kind	Type		
Derived from	uint8		
Range	DCM_DEFAULT_SESSION	0x01	–
	DCM_PROGRAMMING_SESSION	0x02	–
	DCM_EXTENDED_DIAGNOSTIC_SESSION	0x03	–
	DCM_SAFETY_SYSTEM_DIAGNOSTIC_SESSION	0x04	–
	configuration dependent	0x40...0x7E	(according to "diagnosticSessionType" parameter of Diagnostic SessionControl request)
Description	Session type definition. 0, 127 and all values above 127 are reserved by ISO.		
Variation	–		





Available via	Rte_Dcm_Type.h

]

8.8.1.5 Dcm_ProtocolType

[SWS_Dcm_00979] Definition of ImplementationDataType Dcm_ProtocolType [

Name	Dcm_ProtocolType		
Kind	Type		
Derived from	uint8		
Range	DCM_OBD_ON_CAN	0x00	OBD on CAN (ISO15765-4; ISO15031-5)
	DCM_OBD_ON_FLEXRAY	0x01	(OBD on Flexray (Manufacturer specific; ISO15031-5))
	DCM_OBD_ON_IP	0x02	(OBD on Internet Protocol (Manufacturer specific; ISO15031-5))
	DCM_UDS_ON_CAN	0x03	UDS on CAN (ISO15765-3; ISO14229-1)
	DCM_UDS_ON_FLEXRAY	0x04	UDS on FlexRay (Manufacturer specific; ISO14229-1)
	DCM_UDS_ON_IP	0x05	(UDS on Internet Protocol (Manufacturer specific; ISO14229-1))
	DCM_ROE_ON_CAN	0x06	Response On Event on CAN
	DCM_ROE_ON_FLEXRAY	0x07	Response On Event on FlexRay
	DCM_ROE_ON_IP	0x08	(Response on Event on Internet Protocol)
	DCM_PERIODICTRANS_ON_CAN	0x09	Periodic Transmission on CAN
	DCM_PERIODICTRANS_ON_FLEXRAY	0x0A	Periodic Transmission on FlexRay
	DCM_PERIODICTRANS_ON_IP	0x0B	(Periodic Transmission on Internet Protocol)
	DCM_NO_ACTIVE_PROTOCOL	0x0C	No protocol has been started
	DCM_UDS_ON_LIN	0x0D	UDS on LIN (ISO14229-1; ISO14229-7)
	Reserved for further AUTOSAR implementation	0x0E..0xEF	–
	DCM_SUPPLIER_1	0xF0	Reserved for SW supplier specific.
	DCM_SUPPLIER_2	0xF1	Reserved for SW supplier specific.
	DCM_SUPPLIER_3	0xF2	Reserved for SW supplier specific.
	DCM_SUPPLIER_4	0xF3	Reserved for SW supplier specific.





	DCM_SUPPLIER_5	0xF4	Reserved for SW supplier specific.
	DCM_SUPPLIER_6	0xF5	Reserved for SW supplier specific.
	DCM_SUPPLIER_7	0xF6	Reserved for SW supplier specific.
	DCM_SUPPLIER_8	0xF7	Reserved for SW supplier specific.
	DCM_SUPPLIER_9	0xF8	Reserved for SW supplier specific.
	DCM_SUPPLIER_10	0xF9	Reserved for SW supplier specific.
	DCM_SUPPLIER_11	0xFA	Reserved for SW supplier specific.
	DCM_SUPPLIER_12	0xFB	Reserved for SW supplier specific.
	DCM_SUPPLIER_13	0xFC	Reserved for SW supplier specific.
	DCM_SUPPLIER_14	0xFD	Reserved for SW supplier specific.
	DCM_SUPPLIER_15	0xFE	Reserved for SW supplier specific.
Description	Protocol type definition		
Variation	–		
Available via	Rte_Dcm_Type.h		

]

8.8.1.6 Dcm_NegativeResponseCodeType

[SWS_Dcm_00980] Definition of ImplementationDataType Dcm_NegativeResponseCodeType [

Name	Dcm_NegativeResponseCodeType		
Kind	Type		
Derived from	uint8		
Range	DCM_POS_RESP	0x00	PR
	range of values 0x01..0x0F reserved by ISO 14229	0x01..0x0F	ISOSAERESRVD
	DCM_E_GENERALREJECT	0x10	GR
	DCM_E_SERVICENOTSUPPORTED	0x11	SNS
	DCM_E_SUBFUNCTION-NOTSUPPORTED	0x12	SFNS
	DCM_E_INCORRECTMESSAGELENGTHORINVALID-FORMAT	0x13	IMLOIF
	DCM_E_RESPONSETOOLONG	0x14	RTL





	range of values 0x15..0x20 reserved by ISO 14229	0x15..0x20	ISOSAERESRVD
	DCM_E_BUSYREPEATREQUEST	0x21	BRR
	DCM_E_CONDITIONSNOT-CORRECT	0x22	CNC
	value 0x23 reserved by ISO 14229	0x23	ISOSAERESRVD
	DCM_E_REQUESTSEQUENCEERROR	0x24	RSE
	DCM_E_NORESPONSEFROMSUB-NETCOMPONENT	0x25	NRFSC
	DCM_E_FAILUREPREVENTSEXECUTIONOFREQUESTEDACTION	0x26	FPEORA
	range of values 0x27..0x30 reserved by ISO 14229	0x27..0x30	ISOSAERESRVD
	DCM_E_REQUESTOUTOFRANGE	0x31	ROOR
	value 0x32 reserved by ISO 14229	0x32	ISOSAERESRVD
	DCM_E_SECURITYACCESSDENIED	0x33	SAD
	DCM_E_AUTHENTICATIONREQUIRED	0x34	AR
	DCM_E_INVALIDKEY	0x35	IK
	DCM_E_EXCEEDNUMBEROFATTEMPTS	0x36	ENOA
	DCM_E_REQUIRED-TIMEDELAYNOTEXPIRED	0x37	RTDNE
	DCM_E_SECUREDATATRANSMISSIONREQUIRED	0x38	SDTR
	DCM_E_SECUREDATATRANSMISSIONNOTALLOWED	0x39	SDTNA
	DCM_E_SECURE-DATAVERIFICATIONFAILED	0x3A	SDTF
	range of values 0x3B..0x4F reserved by ISO 15764	0x3B..0x4F	RBEDLSD
	DCM_E_CERTIFICATEVERIFICATIONFAILEDIN-VALIDTIMEPERIOD	0x50	CVFITP
	DCM_E_CERTIFICATEVERIFICATIONFAILEDIN-VALIDSIGNATURE	0x51	CVFIS
	DCM_E_CERTIFICATEVERIFICATIONFAILEDIN-VALIDCHAINOFTRUST	0x52	CVFICOT
	DCM_E_CERTIFICATEVERIFICATIONFAILEDININVALIDTYPE	0x53	CVFIT





DCM_E_CERTIFICATEVERIFICATIONFAILEDIN- VALIDFORMAT	0x54	CVFIF
DCM_E_CERTIFICATEVERIFICATIONFAILEDIN- VALIDCONTENT	0x55	CVFIC
DCM_E_CERTIFICATEVERIFICATIONFAILEDIN- VALIDSCOPE	0x56	CVFIS
DCM_E_CERTIFICATEVERIFICATIONFAILEDIN- VALIDDCERTIFICATE REVOKED	0x57	CVFIC
DCM_E_OWNERSHIPVERIFICATIONFAILED	0x58	OVF
DCM_E_CHALLENGECALCULATIONFAILED	0x59	CCF
DCM_E_SETTINGACCESSRIGHTSFAILED	0x5A	SARF
DCM_E_SESSIONKEYCREATION- DERIVATIONFAILED	0x5B	SKCDF
DCM_E_CONFIGURATIONDATAUSAGEFAILED	0x5C	CDUF
DCM_E_DEAUTHENTICATIONFAILED	0x5D	DAF
range of values 0x5E..0x6F reserved by ISO 14229	0x5E..0x6F	ISOSAERESRVD
DCM_E_UPLOADDOWN- LOADNOTACCEPTED	0x70	UDNA
DCM_E_TRANSFERDATA- SUSPENDED	0x71	TDS
DCM_E_GENERALPROGRAMMINGFAILURE	0x72	GPF
DCM_E_WRONGBLOCK- SEQUENCECOUNTER	0x73	WBSC
range of values 0x74..0x77 reserved by ISO 14229	0x74..0x77	ISOSAERESRVD
range of values 0x79..0x7D reserved by ISO 14229	0x79..0x7D	ISOSAERESRVD
DCM_E_SUBFUNCTION- NOTSUPPORTEDINAC- TIVESESSION	0x7E	SFNSIAS
DCM_E_SERVICENOTSUPPORTEDIN- ACTIVESESSION	0x7F	SNSIAS
value 0x80 reserved by ISO 14229	0x80	ISOSAERESRVD
DCM_E_RPMTOOHIGH	0x81	RPMTH
DCM_E_RPMTOOLOW	0x82	RPMTL
DCM_E_ENGINEISRUNNING	0x83	EIR
DCM_E_ENGINEISNOTRUNNING	0x84	EINR





	DCM_E_ENGINE RUN-TIME TOO LOW	0x85	ERTTL
	DCM_E_TEMPERATURE TOO HIGH	0x86	TEMPTH
	DCM_E_TEMPERATURE TOO LOW	0x87	TEMPTL
	DCM_E_VEHICLE SPEED TOO HIGH	0x88	VSTH
	DCM_E_VEHICLE SPEED TOO LOW	0x89	VSTL
	DCM_E_THROTTLE_PEDAL TOO HIGH	0x8A	TPTH
	DCM_E_THROTTLE_PEDAL TOO LOW	0x8B	TPTL
	DCM_E_TRANSMISSION-RANGE NOT IN NEUTRAL	0x8C	TRNIN
	DCM_E_TRANSMISSION-RANGE NOT IN GEAR	0x8D	TRNIG
	value 0x8E reserved by ISO 14229	0x8E	ISOSAERESRVD
	DCM_E_BRAKE SWITCH NOT CLOSED	0x8F	BSNC
	DCM_E_SHIFTER LEVER NOT IN PARK	0x90	SLNIP
	DCM_E_TORQUE CONVERTER CLUTCH LOCKED	0x91	TCCL
	DCM_E_VOLTAGE TOO HIGH	0x92	VTH
	DCM_E_VOLTAGE TOO LOW	0x93	VTL
	DCM_E_RESOURCE TEMPORARILY NOT AVAILABLE	0x94	RTNA
	range of values 0x95..0xEF reserved by ISO 14229	0x95..0xEF	RFSCNC
	DCM_E_VMSCNC_0	0xF0	VMSCNC
	DCM_E_VMSCNC_1	0xF1	VMSCNC1
	DCM_E_VMSCNC_2	0xF2	VMSCNC2
	DCM_E_VMSCNC_3	0xF3	VMSCNC3
	DCM_E_VMSCNC_4	0xF4	VMSCNC4
	DCM_E_VMSCNC_5	0xF5	VMSCNC5
	DCM_E_VMSCNC_6	0xF6	VMSCNC6
	DCM_E_VMSCNC_7	0xF7	VMSCNC7
	DCM_E_VMSCNC_8	0xF8	VMSCNC8
	DCM_E_VMSCNC_9	0xF9	VMSCNC9
	DCM_E_VMSCNC_A	0xFA	VMSCNCA
	DCM_E_VMSCNC_B	0xFB	VMSCNCB
	DCM_E_VMSCNC_C	0xFC	VMSCNCC
	DCM_E_VMSCNC_D	0xFD	VMSCNCD
	DCM_E_VMSCNC_E	0xFE	VMSCNCE





	value 0xFF reserved by ISO 14229	0xFF	ISOSAERESRVD
Description	This Table of available Negative Response Codes represents the allowed Response Codes an AUTOSAR SW Component shall return after a function call. For the allowed NRC of the executed Service ID please refer to the specification of the service in ISO14229-1 (UDS) and ISO15031-5 (OBD/CARB) (Response code parameter definition Table 18 of ISO 14229-1:2020).		
Variation	–		
Available via	Rte_Dcm_Type.h		

8.8.1.7 Dcm_DataElementType_{Data}Type

[SWS_Dcm_91051] Definition of ImplementationData Type Dcm_DataElement_{Data}Type [

Name	Dcm_DataElement_{Data}Type	
Kind	Type	
Derived from	Basetype	Variation
	Dcm_DataElement_{Data}_ArrayType	(({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)} == ([S U]INT[8 16 32 64]FLOAT DOUBLE)_N) ({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)} == ([S U]INT[8 16 32 64]FLOAT DOUBLE)_N))
	Dcm_DataElement_{Data}_PrimitiveType	(({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)} == (BOOLEAN [S U]INT[8 16 32 64]FLOAT DOUBLE)) ({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)} == (BOOLEAN [S U]INT[8 16 32 64]FLOAT DOUBLE)))
	Dcm_DataElement_{Data}_StructuredType	DcmDspDidSignal with at least one DcmDspDidSignalComposite Ref.
Description	Common description for S/R and C/S data elements.	
Variation	(((Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort == USE_DATA_ELEMENT_SPECIFIC_INTERFACES)) && (({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataUsePort)} == USE_DATA_SENDER_RECEIVER) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01/DcmDspPidDataUsePort)} == USE_DATA_SENDER_RECEIVER) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01/DcmDspPidDataUsePort)} == USE_DATA_SENDER_RECEIVER_AS_SERVICE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataUsePort)} == USE_DATA_SENDER_RECEIVER_AS_SERVICE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_ASYNC_CLIENT_SERVER) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_ASYNC_CLIENT_SERVER_ERROR) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER))) Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME)})	
Available via	Rte_Dcm_Type.h	

8.8.1.8 Dcm_DataElement_{Data}StructuredType

[SWS_Dcm_91118] Definition of ImplementationDataType Dcm_DataElement_{Data}_StructuredType [

Name	Dcm_DataElement_{Data}_StructuredType	
Kind	Structure	
Elements	structureMember	
	Type	Dcm_DataElement_{Data}Type
	Comment	–
Description	–	
Variation	DcmDspDidSignal.DcmDspDidSignalCompositeRef.DcmDspDidSignalCompositePool	
Available via	Rte_Dcm_Type.h	

]

8.8.1.9 Dcm_DataElementType_{Data}ArrayType

[SWS_Dcm_91063] Definition of ImplementationDataType Dcm_DataElement_{Data}_ArrayType [

Name	Dcm_DataElement_{Data}_ArrayType	
Kind	Array	
Element type	Type	Variation
	float32	(({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)} == FLOAT_N) ({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)} == FLOAT_N))
	float64	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == DOUBLE_N) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataType)} == DOUBLE_N))
	sint16	(({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)} == SINT16_N) ({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)} == SINT16_N))
	sint32	(({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)} == SINT32_N) ({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)} == SINT32_N))
	sint64	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == SINT64_N) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataType)} == SINT64_N))
	sint8	(({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)} == SINT8_N) ({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)} == SINT8_N))
	uint16	(({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)} == UINT16_N) ({ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)} == UINT16_N))





	uint32	((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)) == UINT32_N) (ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)) == UINT32_N))
	uint64	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)) == UINT64_N) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataType)) == UINT64_N))
	uint8	((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)) == UINT8_N) (ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)) == UINT8_N))
Size	–	
Description	–	
Variation	((({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == ({S U INT[8 16 32 64] FLOAT DOUBLE)_N) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataType)} == ({S U INT[8 16 32 64] FLOAT DOUBLE)_N)) && ({Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort == USE_DATA_ELEMENT_SPECIFIC_INTERFACES}) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataUsePort)} == USE_DATA_SENDER_RECEIVER) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01/DcmDspPidDataUsePort)} == USE_DATA_SENDER_RECEIVER) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01/DcmDspPidDataUsePort)} == USE_DATA_SENDER_RECEIVER) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataUsePort)} == USE_DATA_SENDER_RECEIVER) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_ASYNC_CLIENT_SERVER) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_ASYNC_CLIENT_SERVER_ERROR) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER))) Data = ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME))) ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME))) Data = ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME))) ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME)))	
Available via	Rte_Dcm_Type.h	

]

8.8.1.10 Dcm_DataElementType_{Data}PrimitiveType

[SWS_Dcm_91062] Definition of ImplementationDataType Dcm_DataElement_{Data}_PrimitiveType [

Name	Dcm_DataElement_{Data}_PrimitiveType	
Kind	Type	
Derived from	Basetype	Variation
	boolean	((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)) == BOOLEAN) (ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)) == BOOLEAN))
	float32	((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)) == FLOAT) (ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)) == FLOAT))





	float64	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)) == DOUBLE) ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataType)) == DOUBLE))
	sint16	((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)) == SINT16) ((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)) == SINT16))
	sint32	((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)) == SINT32) ((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)) == SINT32))
	sint64	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)) == SINT64) ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataType)) == SINT64))
	sint8	((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)) == SINT8) ((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)) == SINT8))
	uint16	((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)) == UINT16) ((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)) == UINT16))
	uint32	((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)) == UINT32) ((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)) == UINT32))
	uint64	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)) == UINT64) ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataType)) == UINT64))
	uint8	((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspData.?DcmDspDataType)) == UINT8) ((ecuc(Dcm/?DcmConfigSet/?DcmDsp/?DcmDspPid/?DcmDspPidData/?DcmDspPidService01.?DcmDspPidDataType)) == UINT8))
Description	–	
Variation	((({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == (BOOLEAN [S U]INT[8 16 32 64] FLOAT DOUBLE)) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataType)} == (BOOLEAN [S U]INT[8 16 32 64] FLOAT DOUBLE))) && ({Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort == USE_DATA_ELEMENT_SPECIFIC_INTERFACES}) && (({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataUsePort)} == USE_DATA_SENDER_RECEIVER) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01/DcmDspPidDataUsePort)} == USE_DATA_SENDER_RECEIVER) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01/DcmDspPidDataUsePort)} == USE_DATA_SENDER_RECEIVER_AS_SERVICE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataUsePort)} == USE_DATA_SENDER_RECEIVER_AS_SERVICE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_ASYNC_CLIENT_SERVER) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_ASYNC_CLIENT_SERVER_ERROR) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER))) Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME)}) Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME)})	





Available via	Rte_Dcm_Type.h

┌

8.8.1.11 Dcm_DataArrayTypeUint8_{Data}Type

[SWS_Dcm_01121] Definition of ImplementationDataType Dcm_DataArrayType Uint8_{Data}Type ┌

Name	Dcm_DataArrayTypeUint8_{Data}Type		
Kind	Array	Element type	uint8
Size	(((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataByteSize))) ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.DcmDspPidDataByteSize)))) Elements		
Description	–		
Variation	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == USE_DATA_ELEMENT_SPECIFIC_INTERFACES) && (({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == UINT8_N) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == UINT8_DYN) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataType)} == UINT8_N)) Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME)})		
Available via	Rte_Dcm_Type.h		

┌

8.8.1.12 {DID}_Struct_DataType

This data type has a different modeling as other data types. The {DID}_Struct_DataType datatype is modeled as prosa text only. At the time this specification was created there are no means to visualize this datatype with existing AUTOSAR tooling as table as all the other data types. Still AUTOSAR allows modeling such data types. Simply that they cannot be shown here as table.

[SWS_Dcm_91056] Definition of ImplementationDataType {DID}_Struct_DataType ┌

Name	{DID}_Struct_DataType
Kind	Structure
Description	The elements of this structure data type is a composition of all DcmDspDataElement of the DcmDspDid. Example: A DID with the 3 data elements uint32 data1, sint8 data2 and sint16 data 3, has a structure definition of struct { uint32 data1, sint8 data2, sint16 data 3}.
Variation	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == ((USE_ATOMIC_SENDER_RECEIVER_INTERFACE USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE) USE_ATOMIC_NV_DATA_INTERFACE)) DID = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)})
Available via	Rte_Dcm_Type.h

┌

8.8.1.13 Dcm_RangeArray_{Range}Type

[SWS_Dcm_01012] Definition of ImplementationDataType Dcm_RangeArray_{Range}Type [

Name	Dcm_RangeArray_{Range}Type		
Kind	Array	Element type	uint8
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRangeMaxDataLength)) Elements		
Description	–		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRange.DcmDspDidRangeUsePort)) == TRUE Range = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRange.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

]

8.8.1.14 Dcm_InfoTypeServicesArray_{VehInfoData}Type

[SWS_Dcm_01013] Definition of ImplementationDataType Dcm_InfoTypeServicesArray_{VehInfoData}Type [

Name	Dcm_InfoTypeServicesArray_{VehInfoData}Type		
Kind	Array	Element type	uint8
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspVehInfo/DcmDspVehInfoData.DcmDspVehInfoDataSize)) Elements		
Description	–		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspVehInfo/DcmDspVehInfoData/DcmDspVehInfoDataUsePort)) == TRUE VehInfoData = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspVehInfo/DcmDspVehInfoData.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

]

8.8.1.15 Dcm_RequestControlServicesInArray_{Tid}Type

[SWS_Dcm_01014] Definition of ImplementationDataType Dcm_RequestControlServicesInArray_{Tid}Type [

Name	Dcm_RequestControlServicesInArray_{Tid}Type		
Kind	Array	Element type	uint8
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestControl.DcmDspRequestControlInBuffer Size)) Elements		
Description	–		
Variation	Tid = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestControl.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

]

8.8.1.16 Dcm_RequestControlServicesOutArray_{Tid}Type**[SWS_Dcm_01015] Definition of ImplementationDataType Dcm_RequestControlServicesOutArray_{Tid}Type** [

Name	Dcm_RequestControlServicesOutArray_{Tid}Type		
Kind	Array	Element type	uint8
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestControl.DcmDspRequestControlOutBufferSize)} Elements		
Description	–		
Variation	Tid = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestControl.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

]

8.8.1.17 Dcm_ScalingInfoArray_{Data}Type**[SWS_Dcm_01017] Definition of ImplementationDataType Dcm_ScalingInfoArray_{Data}Type** [

Name	Dcm_ScalingInfoArray_{Data}Type		
Kind	Array	Element type	uint8
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDataInfo.DcmDspDataScalingInfoSize)} Elements		
Description	–		
Variation	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_ASYNC_CLIENT_SERVER) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNC_CLIENT_SERVER) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData->DcmDspDataInfoRef.DcmDspDataScalingInfoSize)} != NULL) Data = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

]

8.8.1.18 Dcm_RequestDataOut_{Routine}_{Signal}PrimitivType**[SWS_Dcm_01018] Definition of ImplementationDataType Dcm_RequestDataOut_{Routine}_{Signal}PrimitivType** [

Name	Dcm_RequestDataOut_{Routine}_{Signal}PrimitivType		
Kind	Type		
Derived from	Basetype	Variation	
	float32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == FLOAT	





	float64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == DOUBLE
	sint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == SINT16
	sint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == SINT32
	sint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == SINT64
	sint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == SINT8
	uint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == UINT16
	uint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == UINT32
	uint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == UINT64
	uint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == UINT8
Description		–
Variation		({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == [U S]INT[8 16 32 64] FLOAT DOUBLE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)} == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
Available via		Rte_Dcm_Type.h

]

8.8.1.19 Dcm_RequestDataIn_{Routine}_{Signal}PrimitiveType

[SWS_Dcm_91054] Definition of ImplementationDataType Dcm_RequestDataIn_{Routine}_{Signal}PrimitiveType [

Name	Dcm_RequestDataIn_{Routine}_{Signal}PrimitiveType	
Kind	Type	
Derived from	Basetype	Variation
	float32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == FLOAT
	float64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == DOUBLE
	sint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == SINT16
	sint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == SINT32
	sint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == SINT64
	sint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == SINT8
	uint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == UINT16
	uint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == UINT32
	uint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == UINT64
	uint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == UINT8
Description	–	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == [U S]INT[8 16 32 64] FLOAT DOUBLE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)} == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.20 Dcm_RequestDataOut_{Routine}_{Signal}Type

[SWS_Dcm_91040] Definition of ImplementationDataType Dcm_RequestDataOut_{Routine}_{Signal}Type [

Name	Dcm_RequestDataOut_{Routine}_{Signal}Type	
Kind	Type	
Derived from	Basetype	Variation
	Dcm_RequestDataOut_{Routine}_{Signal}PrimitiveType	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)) == ([U S]INT[8 16 32 64]FLOAT DOUBLE))
	Dcm_RequestDataOut_{Routine}_{Signal}ArrayType	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)) == ([U S]INT[8 16 32 64]FLOAT DOUBLE)_N)
	Dcm_RequestDataOut_{Routine}_{Signal}StructuredType	DcmDspRequestRoutineResultsOutSignal with at least one DcmDspRequestRoutineResultsOutSignalCompositeSignalRef.
Description	–	
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)) == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.21 Dcm_RequestDataOut_{Routine}_{Signal}StructuredType

[SWS_Dcm_91119] Definition of ImplementationDataType Dcm_RequestDataOut_{Routine}_{Signal}StructuredType [

Name	Dcm_RequestDataOut_{Routine}_{Signal}StructuredType	
Kind	Structure	
Elements	structureMember	
	Type	Dcm_RequestDataOut_{Routine}_{Signal}Type
	Comment	–
Description	–	
Variation	DcmDspRoutine.DcmDspRequestRoutineResults.DcmDspRequestRoutineResultsOut.DcmDspRequestRoutineResultsOutSignalCompositePool	
Available via	Rte_Dcm_Type.h	

]

8.8.1.22 Dcm_RequestDataIn_{Routine}_{Signal}Type

[SWS_Dcm_91052] Definition of ImplementationDataType Dcm_RequestDataIn_{Routine}_{Signal}Type [

Name	Dcm_RequestDataIn_{Routine}_{Signal}Type	
Kind	Type	
Derived from	Basetype	Variation
	Dcm_RequestDataIn_{Routine}_{Signal}PrimitiveType	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)) == (([U S]INT[8 16 32 64] FLOAT DOUBLE)))
	Dcm_RequestDataIn_{Routine}_{Signal}ArrayType	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)) == (([U S]INT[8 16 32 64] FLOAT DOUBLE)_N))
	Dcm_RequestDataIn_{Routine}_{Signal}StructuredType	DcmDspRequestRoutineResultsInSignal with at least one DcmDspRequestRoutineResultsInSignalCompositeSignalRef.
Description	–	
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)) == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.23 Dcm_RequestDataIn_{Routine}_{Signal}StructuredType

[SWS_Dcm_91120] Definition of ImplementationDataType Dcm_RequestDataIn_{Routine}_{Signal}StructuredType [

Name	Dcm_RequestDataIn_{Routine}_{Signal}StructuredType	
Kind	Structure	
Elements	StructureMember	
	Type	Dcm_RequestDataIn_{Routine}_{Signal}Type
	Comment	–
Description	–	
Variation	DcmDspRoutine.DcmDspRequestRoutineResults.DcmDspRequestRoutineResultsIn.DcmDspRequestRoutineResultsInSignalCompositePool	
Available via	Rte_Dcm_Type.h	

]

8.8.1.24 Dcm_RequestDataOut_{Routine}_{Signal}ArrayType

[SWS_Dcm_91041] Definition of ImplementationDataType Dcm_RequestDataOut_{Routine}_{Signal}ArrayType [

Name	Dcm_RequestDataOut_{Routine}_{Signal}ArrayType	
Kind	Array	
Element type	Type	Variation
	float32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == FLOAT_N)
	float64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == DOUBLE_N)
	sint16	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == SINT16_N)
	sint32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == SINT32_N)
	sint64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == SINT64_N)
	sint8	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == SINT8_N)
	uint16	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == UINT16_N)
	uint32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == UINT32_N)
	uint64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == UINT64_N)
	uint8	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == UINT8_N)
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineParameterSize)} Elements	
Description	–	
Variation	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == ([U S]INT[8 16 32 64] FLOAT DOUBLE)_N) && {{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)} == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.25 Dcm_RequestDataIn_{Routine}_{Signal}ArrayType

[SWS_Dcm_91055] Definition of ImplementationDataType Dcm_RequestDataIn_{Routine}_{Signal}ArrayType [

Name	Dcm_RequestDataIn_{Routine}_{Signal}ArrayType	
Kind	Array	
Element type	Type	Variation
	float32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == FLOAT_N
	float64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == DOUBLE_N
	sint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == SINT16_N
	sint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == SINT32_N
	sint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == SINT64_N
	sint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == SINT8_N
	uint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == UINT16_N
	uint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == UINT32_N
	uint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == UINT64_N
	uint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == UINT8_N
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineParameterSize)} Elements	
Description	–	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)} == ([U S]INT[8 16 32 64] FLOAT DOUBLE)_N) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)} == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.26 Dcm_RequestFlexibleOutArrayData_{Routine}_{Signal}Type

[SWS_Dcm_01019] Definition of ImplementationDataType Dcm_RequestFlexibleOutArrayData_{Routine}_{Signal}Type 「

Name	Dcm_RequestFlexibleOutArrayData_{Routine}_{Signal}Type		
Kind	Array	Element type	uint8
Size	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineParameterSize) Elements}		
Description	–		
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)) == VARIABLE_LENGTH) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)) == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

」

8.8.1.27 Dcm_RequestFlexibleInArrayData_{Routine}_{Signal}Type

[SWS_Dcm_91053] Definition of ImplementationDataType Dcm_RequestFlexibleInArrayData_{Routine}_{Signal}Type 「

Name	Dcm_RequestFlexibleInArrayData_{Routine}_{Signal}Type		
Kind	Array	Element type	uint8
Size	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineParameterSize) Elements}		
Description	–		
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)) == VARIABLE_LENGTH) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)) == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

」

8.8.1.28 Dcm_StartDataIn_{Routine}_{Signal}PrimitivType

[SWS_Dcm_01020] Definition of ImplementationDataType Dcm_StartDataIn_{Routine}_{Signal}PrimitivType [

Name	Dcm_StartDataIn_{Routine}_{Signal}PrimitivType	
Kind	Type	
Derived from	Basetype	Variation
	float32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == FLOAT
	float64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == DOUBLE
	sint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == SINT16
	sint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == SINT32
	sint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == SINT64
	sint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == SINT8
	uint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == UINT16
	uint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == UINT32
	uint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == UINT64
	uint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == UINT8
Description	–	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == [U S]INT[8 16 32 64] FLOAT DOUBLE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)} == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.29 Dcm_StartDataIn_{Routine}_{Signal}Type

[SWS_Dcm_91042] Definition of ImplementationDataType Dcm_StartDataIn_{Routine}_{Signal}Type [

Name	Dcm_StartDataIn_{Routine}_{Signal}Type	
Kind	Type	
Derived from	Basetype	Variation
	Dcm_StartDataIn_{Routine}_{Signal}PrimitiveType	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)) == ([U S]INT[8 16 32 64] FLOAT DOUBLE))
	Dcm_StartDataIn_{Routine}_{Signal}ArrayType	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)) == ([U S]INT[8 16 32 64] FLOAT DOUBLE)_N)
	Dcm_StartDataIn_{Routine}_{Signal}StructuredType	DcmDspStartRoutineInSignal with at least one DcmDspStartRoutineInSignalCompositeSignalRef.
Description	–	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)} == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.30 Dcm_StartDataIn_{Routine}_{Signal}StructuredType

[SWS_Dcm_91121] Definition of ImplementationDataType Dcm_StartDataIn_{Routine}_{Signal}StructuredType [

Name	Dcm_StartDataIn_{Routine}_{Signal}StructuredType	
Kind	Structure	
Elements	structureMember	
	Type	Dcm_StartDataIn_{Routine}_{Signal}Type
	Comment	–
Description	–	
Variation	DcmDspRoutine.DcmDspStartRoutine.DcmDspStartRoutineIn.DcmDspStartRoutineInSignalCompositePool	
Available via	Rte_Dcm_Type.h	

]

8.8.1.31 Dcm_StartDataIn_{Routine}_{Signal}ArrayType

[SWS_Dcm_91043] Definition of ImplementationDataType Dcm_StartDataIn_{Routine}_{Signal}ArrayType [

Name	Dcm_StartDataIn_{Routine}_{Signal}ArrayType	
Kind	Array	
Element type	Type	Variation
	float32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == FLOAT_N}
	float64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == DOUBLE_N}
	sint16	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == SINT16_N}
	sint32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == SINT32_N}
	sint64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == SINT64_N}
	sint8	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == SINT8_N}
	uint16	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == UINT16_N}
	uint32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == UINT32_N}
	uint64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == UINT64_N}
	uint8	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == UINT8_N}
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineParameterSize)} Elements	
Description	–	
Variation	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == ([U S]INT[8 16 32 64] FLOAT DOUBLE)_N} && {{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineUsePort)} == TRUE} Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.32 Dcm_StartDataOut_{Routine}_{Signal}PrimitivType

[SWS_Dcm_01021] Definition of ImplementationDataType Dcm_StartDataOut_{Routine}_{Signal}PrimitivType [

Name	Dcm_StartDataOut_{Routine}_{Signal}PrimitivType	
Kind	Type	
Derived from	Basetype	Variation
	float32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == FLOAT
	float64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == DOUBLE
	sint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == SINT16
	sint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == SINT32
	sint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == SINT64
	sint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == SINT8
	uint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == UINT16
	uint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == UINT32
	uint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == UINT64
	uint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == UINT8
Description	–	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == [U S]INT[8 16 32 64] FLOAT DOUBLE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/ DcmDspRoutineUsePort)} == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.33 Dcm_StartDataOut_{Routine}_{Signal}Type

[SWS_Dcm_91044] Definition of ImplementationDataType Dcm_StartDataOut_{Routine}_{Signal}Type [

Name	Dcm_StartDataOut_{Routine}_{Signal}Type	
Kind	Type	
Derived from	Basetype	Variation
	Dcm_StartDataOut_{Routine}_{Signal}PrimitiveType	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)) == ([U S]INT[8 16 32 64] FLOAT DOUBLE))
	Dcm_StartDataOut_{Routine}_{Signal}ArrayType	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)) == ([U S]INT[8 16 32 64] FLOAT DOUBLE)_N)
	Dcm_StartDataOut_{Routine}_{Signal}StructuredType	DcmDspStartRoutineOutSignal with at least one DcmDspStartRoutineOutSignalCompositeSignalRef.
Description	–	
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)) == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.34 Dcm_StartDataOut_{Routine}_{Signal}StructuredType

[SWS_Dcm_91122] Definition of ImplementationDataType Dcm_StartDataOut_{Routine}_{Signal}StructuredType [

Name	Dcm_StartDataOut_{Routine}_{Signal}StructuredType	
Kind	Structure	
Elements	structureMember	
	Type	Dcm_StartDataOut_{Routine}_{Signal}Type
	Comment	–
Description	–	
Variation	DcmDspRoutine.DcmDspStartRoutine.DcmDspStartRoutineOut.DcmDspStartRoutineOutSignalCompositePool	
Available via	Rte_Dcm_Type.h	

]

8.8.1.35 Dcm_StartDataOut_{Routine}_{Signal}ArrayType

[SWS_Dcm_91045] Definition of ImplementationDataType Dcm_StartDataOut_{Routine}_{Signal}ArrayType [

Name	Dcm_StartDataOut_{Routine}_{Signal}ArrayType	
Kind	Array	
Element type	Type	Variation
	float32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == FLOAT_N}
	float64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == DOUBLE_N}
	sint16	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == SINT16_N}
	sint32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == SINT32_N}
	sint64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == SINT64_N}
	sint8	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == SINT8_N}
	uint16	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == UINT16_N}
	uint32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == UINT32_N}
	uint64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == UINT64_N}
	uint8	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == UINT8_N}
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineParameterSize)} Elements	
Description	–	
Variation	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == ([U S]INT[8 16 32 64] FLOAT DOUBLE)_N} && {{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)} == TRUE} Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.36 Dcm_StartFlexibleInArrayData_{Routine}_{Signal}Type

[SWS_Dcm_01022] Definition of ImplementationDataType Dcm_StartFlexibleInArrayData_{Routine}_{Signal}Type [

Name	Dcm_StartFlexibleInArrayData_{Routine}_{Signal}Type		
Kind	Array	Element type	uint8
Size	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineParameterSize) Elements		
Description	–		
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)) == VARIABLE_LENGTH) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)) == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

]

8.8.1.37 Dcm_StartFlexibleOutArrayData_{Routine}_{Signal}Type

[SWS_Dcm_01023] Definition of ImplementationDataType Dcm_StartFlexibleOutArrayData_{Routine}_{Signal}Type [

Name	Dcm_StartFlexibleOutArrayData_{Routine}_{Signal}Type		
Kind	Array	Element type	uint8
Size	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineParameterSize) Elements		
Description	–		
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)) == VARIABLE_LENGTH) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)) == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

]

8.8.1.38 Dcm_StopDataIn_{Routine}_{Signal}PrimitivType

[SWS_Dcm_01024] Definition of ImplementationDataType Dcm_StopDataIn_{Routine}_{Signal}PrimitivType [

Name	Dcm_StopDataIn_{Routine}_{Signal}PrimitivType		
Kind	Type		
Derived from	Basetype	Variation	





	float32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == FLOAT
	float64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == DOUBLE
	sint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == SINT16
	sint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == SINT32
	sint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == SINT64
	sint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == SINT8
	uint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == UINT16
	uint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == UINT32
	uint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == UINT64
	uint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == UINT8
Description	–	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == ({U S INT[8 16 32 64] FLOAT DOUBLE)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)} == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

└

8.8.1.39 Dcm_StopDataIn_{Routine}_{Signal}Type

[SWS_Dcm_91046] Definition of ImplementationDataType Dcm_StopDataIn_{Routine}_{Signal}Type [

Name	Dcm_StopDataIn_{Routine}_{Signal}Type	
Kind	Type	
Derived from	Basetype	Variation





	Dcm_StopDataIn_{Routine}_{Signal}PrimitiveType	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == ([U S]INT[8 16 32 64] FLOAT DOUBLE))
	Dcm_StopDataIn_{Routine}_{Signal}ArrayType	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == ([U S]INT[8 16 32 64] FLOAT DOUBLE)_N)
	Dcm_StopDataIn_{Routine}_{Signal}StructuredType	DcmDspStopRoutineInSignal with at least one DcmDspStopRoutineInSignalCompositeSignalRef.
Description	–	
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)} == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.40 Dcm_StopDataIn_{Routine}_{Signal}StructuredType

[SWS_Dcm_91123] Definition of ImplementationDataType Dcm_StopDataIn_{Routine}_{Signal}StructuredType [

Name	Dcm_StopDataIn_{Routine}_{Signal}StructuredType	
Kind	Structure	
Elements	structureMember	
	Type	Dcm_StopDataIn_{Routine}_{Signal}Type
	Comment	–
Description	–	
Variation	DcmDspRoutine.DcmDspStopRoutine.DcmDspStopRoutineIn.DcmDspStopRoutineInSignalCompositePool	
Available via	Rte_Dcm_Type.h	

]

8.8.1.41 Dcm_StopDataIn_{Routine}_{Signal}ArrayType

[SWS_Dcm_91047] Definition of ImplementationDataType Dcm_StopDataIn_{Routine}_{Signal}ArrayType [

Name	Dcm_StopDataIn_{Routine}_{Signal}ArrayType	
Kind	Array	
Element type	Type	Variation





	float32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)}} == FLOAT_N
	float64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)}} == DOUBLE_N
	sint16	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)}} == SINT16_N
	sint32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)}} == SINT32_N
	sint64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)}} == SINT64_N
	sint8	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)}} == SINT8_N
	uint16	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)}} == UINT16_N
	uint32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)}} == UINT32_N
	uint64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)}} == UINT64_N
	uint8	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)}} == UINT8_N
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineParameterSize)} Elements	
Description	–	
Variation	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)}} == ([U S]INT[8 16 32 64]FLOAT DOUBLE)_N && {{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)}} == TRUE Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.42 Dcm_StopDataOut_{Routine}_{Signal}PrimitivType

[SWS_Dcm_01025] Definition of ImplementationDataType Dcm_StopDataOut_{Routine}_{Signal}PrimitivType [

Name	Dcm_StopDataOut_{Routine}_{Signal}PrimitivType	
Kind	Type	
Derived from	Basetype	Variation





	float32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == FLOAT
	float64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == DOUBLE
	sint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == SINT16
	sint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == SINT32
	sint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == SINT64
	sint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == SINT8
	uint16	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == UINT16
	uint32	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == UINT32
	uint64	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == UINT64
	uint8	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == UINT8
Description	–	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == ([U S]INT[8 16 32 64] FLOAT DOUBLE)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)} == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

└

8.8.1.43 Dcm_StopDataOut_{Routine}_{Signal}Type

[SWS_Dcm_91048] Definition of ImplementationDataType Dcm_StopDataOut_{Routine}_{Signal}Type 「

Name	Dcm_StopDataOut_{Routine}_{Signal}Type	
Kind	Type	
Derived from	Basetype	Variation





	Dcm_StopDataOut_{Routine}_{Signal}Primitiv Type	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)) == ([U S]INT[8 16 32 64] FLOAT DOUBLE))
	Dcm_StopDataOut_{Routine}_{Signal}ArrayType	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)) == ([U S]INT[8 16 32 64] FLOAT DOUBLE)_N)
	Dcm_StopDataOut_{Routine}_{Signal}StructuredType	DcmDspStopRoutineOutSignal with at least one DcmDspStopRoutineOutSignalCompositeSignalRef.
Description	–	
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)) == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}	
Available via	Rte_Dcm_Type.h	

]

8.8.1.44 Dcm_StopDataOut_{Routine}_{Signal}StructuredType

[SWS_Dcm_91124] Definition of ImplementationDataType Dcm_StopDataOut_{Routine}_{Signal}StructuredType [

Name	Dcm_StopDataOut_{Routine}_{Signal}StructuredType	
Kind	Structure	
Elements	structureMember	
	Type	Dcm_StopDataOut_{Routine}_{Signal}Type
	Comment	–
Description	–	
Variation	DcmDspRoutine.DcmDspStopRoutine.DcmDspStopRoutineOut.DcmDspStopRoutineOutSignalCompositePool	
Available via	Rte_Dcm_Type.h	

]

8.8.1.45 Dcm_StopDataOut_{Routine}_{Signal}ArrayType

[SWS_Dcm_91049] Definition of ImplementationDataType Dcm_StopDataOut_{Routine}_{Signal}ArrayType [

Name	Dcm_StopDataOut_{Routine}_{Signal}ArrayType	
Kind	Array	
Element type	Type	Variation





	float32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == FLOAT_N}
	float64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == DOUBLE_N}
	sint16	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == SINT16_N}
	sint32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == SINT32_N}
	sint64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == SINT64_N}
	sint8	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == SINT8_N}
	uint16	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == UINT16_N}
	uint32	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == UINT32_N}
	uint64	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == UINT64_N}
	uint8	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == UINT8_N}
Size		{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineParameterSize)} Elements
Description		–
Variation		{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == ([U S]INT[8 16 32 64] FLOAT DOUBLE)_N} && {{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)} == TRUE} Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
Available via		Rte_Dcm_Type.h

]

8.8.1.46 Dcm_StopFlexibleInArrayData_{Routine}_{Signal}Type

[SWS_Dcm_01026] Definition of ImplementationDataType Dcm_StopFlexibleInArrayData_{Routine}_{Signal}Type [

Name	Dcm_StopFlexibleInArrayData_{Routine}_{Signal}Type		
Kind	Array	Element type	uint8
Size	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineParameterSize) Elements		
Description	–		
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)) == VARIABLE_LENGTH) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)) == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

]

8.8.1.47 Dcm_StopFlexibleOutArrayData_{Routine}_{Signal}Type

[SWS_Dcm_01027] Definition of ImplementationDataType Dcm_StopFlexibleOutArrayData_{Routine}_{Signal}Type [

Name	Dcm_StopFlexibleOutArrayData_{Routine}_{Signal}Type		
Kind	Array	Element type	uint8
Size	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineParameterSize) Elements		
Description	–		
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)) == VARIABLE_LENGTH) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRoutineUsePort)) == TRUE) Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

]

8.8.1.48 Dcm_KeyArray_{SecurityLevel}Type

[SWS_Dcm_01028] Definition of ImplementationDataType Dcm_KeyArray_{SecurityLevel}Type [

Name	Dcm_KeyArray_{SecurityLevel}Type		
Kind	Array	Element type	uint8
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.DcmDspSecurityKeySize)} Elements		
Description	–		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.DcmDspSecurityUsePort)} == USE_ASYNC_CLIENT_SERVER SecurityLevel = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

]

8.8.1.49 Dcm_SeedArray_{SecurityLevel}Type

[SWS_Dcm_01029] Definition of ImplementationDataType Dcm_SeedArray_{SecurityLevel}Type [

Name	Dcm_SeedArray_{SecurityLevel}Type		
Kind	Array	Element type	uint8
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.DcmDspSecuritySeedSize)} Elements		
Description	–		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.DcmDspSecurityUsePort)} == USE_ASYNC_CLIENT_SERVER SecurityLevel = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.SHORT-NAME)}		
Available via	Rte_Dcm_Type.h		

]

8.8.1.50 Dcm_SecurityAccessDataRecordArray_{SecurityLevel}Type

[SWS_Dcm_01159] Definition of ImplementationDataType Dcm_SecurityAccessDataRecordArray_{SecurityLevel}Type [

Name	Dcm_SecurityAccessDataRecordArray_{SecurityLevel}Type		
Kind	Array	Element type	uint8
Size	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow/DcmDspSecurityADRSIZE)} Elements		
Description	–		





Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.DcmDspSecurityUsePort)} == USE_ASYNC_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.DcmDspSecurityADRSIZE)} != NULL) SecurityLevel = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.SHORT-NAME)}
Available via	Rte_Dcm_Type.h

]

8.8.1.51 Dcm_RequestDataArrayType

[SWS_Dcm_01165] Definition of ImplementationDataType Dcm_RequestDataAr-rayType 「

Name	Dcm_RequestDataArrayType		
Kind	Array	Element type	uint8
Size	(MAX({ecuc(Dcm/DcmConfigSet/DcmDsl/DcmDslProtocol/DcmDslProtocolRow/DcmDslProtocolRxBufferID->DcmDslBuffer.DcmDslBufferSize)}) - 1) Elements		
Description	–		
Variation	–		
Available via	Rte_Dcm_Type.h		

]

8.8.1.52 Dcm_ControlMask_{DID}Type

[SWS_Dcm_01320] Definition of ImplementationDataType Dcm_ControlMask_{DID}Type 「

Name	Dcm_ControlMask_{DID}Type	
Kind	Type	
Derived from	Basetype	Variation
	Dcm_ControlMask_{Data}ArrayType	((({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} >= 0x05) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_SYNCH_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR))))
	Dcm_ControlMask_{Data}_PrimitiveType	((ecuc(Dcm/DcmConfigSet/ DcmDsp/DcmDspDid/ DcmDspDidInfoRef->DcmDspDidInfo/ DcmDspDidControl/ DcmDspDidControlMaskSize)} <= 0x04)
Description	—	





Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == (USE_DATA_SYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)) == DCM_CONTROLMASK_EXTERNAL) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlEnableMask)) == NULL) DID = ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)))}
Available via	Rte_Dcm_Type.h

└

8.8.1.53 Dcm_inputOutputControlParameterType

[SWS_Dcm_01305] Definition of ImplementationDataType Dcm_InputOutputControlParameterType 「

Name	Dcm_InputOutputControlParameterType		
Kind	Type		
Derived from	uint8		
Range	DCM_RETURN_CONTROL_TO_ECU	0x00	returnControlToECU
	DCM_RESET_TO_DEFAULT	0x01	resetToDefault
	DCM_FREEZE_CURRENT_STATE	0x02	freezeCurrentState
	DCM_SHORT_TERM_ADJUSTMENT	0x03	shortTermAdjustment
	DCM_IDLE	0xff	Idle state, no request in processing (initial value)
Description	–		
Variation	–		
Available via	Rte_Dcm_Type.h		

└

8.8.1.54 Dcm_IOOperationRequest_{DID}Type

[SWS_Dcm_01306] Definition of ImplementationDataType Dcm_IOOperationRequest_{DID}Type 「

Name	Dcm_IOOperationRequest_{DID}Type		
Kind	Structure		
Elements	inputOutputControlParameter		
	Type	Dcm_InputOutputControlParameterType	
	Comment	–	
	controlEnableMask		
	Type	Dcm_Cemr_{DID}Type	





	Comment	–
Description	–	
Variation	<pre>{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)) == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE)) ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)) == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE))}&& {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef-> DcmDspDidInfo/DcmDspDidControl)) != NULL) DID = {(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME))}</pre>	
Available via	Rte_Dcm_Type.h	

]

8.8.1.55 Dcm_IOOperationResponseType

[SWS_Dcm_01307] Definition of ImplementationDataType Dcm_IOOperationResponseType [

Name	Dcm_IOOperationResponseType		
Kind	Type		
Derived from	uint8		
Range	DCM_POSITIVE_RESPONSE	0x00	positive response (similar to E_OK)
	DCM_GENERAL_REJECT	0x10	NRC generalReject
	DCM_BUSY_REPEAT_REQUEST	0x21	NRC busyRepeatRequest
	DCM_CONDITIONS_NOT_CORRECT	0x22	NRC conditionsNotCorrect
	DCM_FAILURE_PREVENTS_EXECUTION	0x26	NRC FailurePreventsExecutionOf RequestedAction
	DCM_REQUEST_OUT_OF_RANGE	0x31	NRC requestOutOfRange
	DCM_RESPONSE_PENDING	0x78	ResponsePending (similar to E_PENDING)
Description	–		
Variation	–		
Available via	Rte_Dcm_Type.h		

]

8.8.1.56 Dcm_DidSupportedType

[SWS_Dcm_01138] Definition of ImplementationDataType Dcm_DidSupportedType [

Name	Dcm_DidSupportedType
Kind	Type
Derived from	uint8





Range	DCM_DID_SUPPORTED	0x00	–
	DCM_DID_NOT_SUPPORTED	0x01	–
Description	–		
Variation	–		
Available via	Rte_Dcm_Type.h		

]

8.8.1.57 Dcm_FileAndDirNameType

[SWS_Dcm_91066] Definition of ImplementationDataType Dcm_FileAndDirNameType

Upstream requirements: [RS_Diag_04135](#)

[

Name	Dcm_FileAndDirNameType		
Kind	Array	Element type	uint8
Size	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestFileTransfer.DcmRequestFileTransferMaxFileAndDirName)}) Elements		
Description	–		
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestFileTransfer/DcmRequestFileTransferUsePort)}) == TRUE)		
Available via	Rte_Dcm_Type.h		

]

8.8.1.58 Dcm_ResponseDataArrayType

[SWS_Dcm_91064] Definition of ImplementationDataType Dcm_ResponseDataArrayType

Upstream requirements: [RS_Diag_04135](#)

[

Name	Dcm_ResponseDataArrayType		
Kind	Array	Element type	uint8
Size	(MAX({ecuc(Dcm/DcmConfigSet/DcmDsl/DcmDslProtocol/DcmDslProtocolRow/DcmDslProtocolTxBufferRef->DcmDslBuffer.DcmDslBufferSize)}) -1) Elements		
Description	–		
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestFileTransfer/DcmRequestFileTransferUsePort)}) == TRUE)		
Available via	Rte_Dcm_Type.h		

]

8.8.1.59 Dcm_AuthenticationRoleType

[SWS_Dcm_91068] Definition of ImplementationDataType Dcm_AuthenticationRoleType [

Name	Dcm_AuthenticationRoleType		
Kind	Array	Element type	uint8
Size	(({ecuc(Dcm/DcmDsp/DcmDspAuthentication.DcmDspAuthenticationRoleSize))) Elements		
Description	–		
Variation	–		
Available via	Rte_Dcm_Type.h		

]

8.8.1.60 Dcm_ControlMask_{Data}ArrayType

[SWS_Dcm_91050] Definition of ImplementationDataType Dcm_ControlMask_{Data}ArrayType [

Name	Dcm_ControlMask_{Data}ArrayType		
Kind	Array	Element type	uint8
Size	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize))) Elements		
Description	The current DcmDspData is referenced by the DcmDspDID.		
Variation	((({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)) >= 0x05) && (({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_SYNCH_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR))) Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}))		
Available via	Rte_Dcm_Type.h		

]

8.8.1.61 Dcm_ControlMask_{Data}PrimitiveType

[SWS_Dcm_91089] Definition of ImplementationDataType Dcm_ControlMask_{Data}_PrimitiveType [

Name	Dcm_ControlMask_{Data}_PrimitiveType		
Kind	Type		
Derived from	Basetype	Variation	
	uint16	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)) == 0x02)	





	uint32	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)}== 0x03) {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)}== 0x04))
	uint8	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} == 0x01)
Description	The current DcmDspData is referenced by the DcmDspDID.	
Variation	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_SYNCH_CLIENT_SERVER USE_DATA_ASYNCH_CLIENT_SERVER USE_DATA_ASYNCH_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlEnableMask)} == NULL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04)) Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})	
Available via	Rte_Dcm_Type.h	

]

8.8.1.62 Dcm_Cemr_{DID}Type

[SWS_Dcm_91087] Definition of ImplementationDataType Dcm_Cemr_{DID}Type

[

Name	Dcm_Cemr_{DID}Type	
Kind	Bitfield	
Derived from	Dcm_ControlMask_{DID}Type	
Elements	Variable bit	
	Kind	bit
	Name	{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidControl.DcmDspDidControlEnableMask.SHORT-NAME)}}
	Mask	2 ^{(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} * 8 - 1) - {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlEnableMask/DcmDspDidControlMaskBitPosition)})}
	Description	Bitmask of data element in control enable mask record.
Description	–	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidControl/DcmDspDidControlEnableMask)} != NULL) DID = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)})	
Available via	Rte_Dcm_Type.h	

]

The [Dcm_Cemr_Type](#) data type defines a bitfield text table with a single mask entry for each valid bit in the control enable mask record. A value of 0 in the [DcmDspDidControlMaskBitPosition](#) represents the most significant bit in the most significant byte of the control enable mask. Therefore it is mapped to Bit 7 (0x80) of the first byte of the control enable mask. In a two byte [CEMR](#) the bit position 0 would map to 0x8000,

the bit position 1 would map to 0x4000, ... Also this is a unusual representation it helps to keep the bits stable if the size of the [CEMR](#) is changed. The definition of the bits is directly defined by [DcmDspDidControlEnableMask](#).

8.8.2 Sender-Receiver-Interfaces

Using the concepts of the [SW-C](#) template, the interface is defined as follows if Sender-Receiver interface is used ([DcmDspDataUsePort](#) set to

[USE_DATA_SENDER_RECEIVER](#) || [USE_DATA_SENDER_RECEIVER_AS_SERVICE](#)):

8.8.2.1 DataServices_{DID}

[SWS_Dcm_91057] Definition of SenderReceiverInterface DataServices_{DID}

Upstream requirements: [RS_Diag_04218](#)

[

Name	DataServices_{DID}	
Comment	–	
IsService	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE)	
Variation	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE))) DID = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)}	
Data Elements	data	
	Type	{DID}_Struct_DataType
	Variation	DID = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)}

]

8.8.2.2 DataServices_{Data}

[SWS_Dcm_00687] Definition of SenderReceiverInterface DataServices_{Data} [

Name	DataServices_{Data}	
Comment	–	
IsService	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataUsePort)} == USE_DATA_SENDER_RECEIVER_AS_SERVICE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01/DcmDspPidDataUsePort)} == USE_DATA_SENDER_RECEIVER_AS_SERVICE))	





Variation	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == USE_DATA_ELEMENT_SPECIFIC_INTERFACES)&&{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataUsePort)} == (USE_DATA_SENDER_RECEIVER USE_DATA_SENDER_RECEIVER_AS_SERVICE)}) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01/DcmDspPidDataUsePort)} == (USE_DATA_SENDER_RECEIVER USE_DATA_SENDER_RECEIVER_AS_SERVICE)) Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME)})	
Data Elements	data	
	Type	Dcm_DataElement_{Data}Type
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME)})

」

8.8.2.3 IOControlRequest_{DID}

[SWS_Dcm_01308] **Definition of SenderReceiverInterface IOControlRequest_{DID}** 「

Name	IOControlRequest_{DID}	
Comment	Attention: controlState is only valid in case of IOOperationRequest is set to shortTerm Adjustment. The DCM provides a byte stream which could be transformed via transformer into an complex type.	
IsService	true	
Variation	((({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE))&&{{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef-> DcmDspDidInfo/ DcmDspDidControl)}!= NULL}) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)}== DCM_CONTROLMASK_EXTERNAL))) DID = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)})	
Data Elements	underControl	
	Type	Dcm_Cemr_{DID}Type
	Variation	—
	IOOperationRequest	
	Type	Dcm_IOOperationRequest_{DID}Type
	Variation	DID = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)})
	controlState	
	Type	{DID}_Struct_DataType
	Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef-> DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == True) DID = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)})

」

8.8.2.4 IOControlResponse_{DID}

[SWS_Dcm_01309] Definition of SenderReceiverInterface IOControlResponse_{DID} [

Name	IOControlResponse_{DID}		
Comment	–		
IsService	true		
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE))) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef-> DcmDspDidInfo/DcmDspDidControl)} != NULL) DID = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)})		
Data Elements	IOOperationResponse		
	Type	Dcm_IOOperationResponseType	
	Variation	–	

]

8.8.3 Client-Server-Interfaces

8.8.3.1 SecurityAccess_{SecurityLevel}

Provides functions required for the UDS Service SecurityAccess (see [SWS_Dcm_00323],[SWS_Dcm_00862] and [SWS_Dcm_00863]).

Using the concepts of the SW-C template, the interface is defined as follows if ClientServer interface is used (DcmDspSecurityUsePort set to or USE_ASYNC_CLIENT_SERVER):

[SWS_Dcm_00685] Definition of ClientServerInterface SecurityAccess_{SecurityLevel} [

Name	SecurityAccess_{SecurityLevel}		
Comment	–		
IsService	true		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.DcmDspSecurityUsePort)} == USE_ASYNC_CLIENT_SERVER SecurityLevel = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	10	DCM_E_PENDING	Request is not yet finished. Further call(s) required to finish.
	11	DCM_E_COMPARE_KEY_FAILED	Key did not match.

Operation	CompareKey	
Comment	–	
Relates to	Xxx_CompareKey	
Variation	–	
Parameters	Key	
	Type	Dcm_KeyArray_{SecurityLevel}Type
	Direction	IN
	Comment	Key, which needs to be compared
	Variation	SecurityLevel = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.SHORT-NAME)}
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	return Error Code
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_COMPARE_KEY_FAILED	

Operation	GetSecurityAttemptCounter	
Comment	Restore the attempt counter from the application	
Relates to	Xxx_GetSecurityAttemptCounter	
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.DcmDspSecurityAttemptCounterEnabled)} == TRUE	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	AttemptCounter	
	Type	uint8
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	GetSeed	
Comment	–	
Relates to	Xxx_GetSeed	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.DcmDspSecurityADRSIZE)} == NULL)	





Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	Seed	
	Type	Dcm_SeedArray_{SecurityLevel}Type
	Direction	OUT
	Comment	Pointer for provided seed
	Variation	SecurityLevel = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.SHORT-NAME)}
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	GetSeed	
Comment	–	
Relates to	Xxx_GetSeed	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.DcmDspSecurityADRSIZE)} != NULL)	
Parameters	SecurityAccessDataRecord	
	Type	Dcm_SecurityAccessDataRecordArray_{SecurityLevel}Type
	Direction	IN
	Comment	–
	Variation	SecurityLevel = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.SHORT-NAME)}
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	Seed	
	Type	Dcm_SeedArray_{SecurityLevel}Type
	Direction	OUT
	Comment	Pointer for provided seed
	Variation	SecurityLevel = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.SHORT-NAME)}
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–





Possible Errors	E_OK E_NOT_OK DCM_E_PENDING		
Operation	SetSecurityAttemptCounter		
Comment	Store the attempt counter in the application		
Relates to	Xxx_SetSecurityAttemptCounter		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.DcmDspSecurityAttemptCounterEnabled)} == TRUE		
Parameters	OpStatus		
	Type	Dcm_OpStatusType	
	Direction	IN	
	Comment	–	
	Variation	–	
	AttemptCounter		
	Type	uint8	
	Direction	IN	
	Comment	–	
	Variation	–	
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING		

]

8.8.3.2 DataServices_{Data}

Using the concepts of the SW-C template, the interface is defined as follows if ClientServer interface is used (DcmDspDataUsePort set to USE_DATA_SYNCH_CLIENT_SERVER or USE_DATA_ASYNCH_CLIENT_SERVER or USE_DATA_ASYNCH_CLIENT_SERVER_ERROR):

[SWS_Dcm_00686] Definition of ClientServerInterface DataServices_{Data} [

Name	DataServices_{Data}		
Comment	–		
IsService	true		
Variation	((({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == USE_DATA_ELEMENT_SPECIFIC_INTERFACES)) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_ASYNCH_CLIENT_SERVER) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_ASYNCH_CLIENT_SERVER_ERROR) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER)) Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME)})		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed





	10	DCM_E_PENDING	Request is not yet finished. Further call(s) required to finish.

Operation	ConditionCheckRead	
Comment	–	
Relates to	Xxx_ConditionCheckRead	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidRead)} != NULL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataConditionCheckReadFncUsed)} == TRUE)	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ConditionCheckRead		
Comment	–		
Relates to	Xxx_ConditionCheckRead		
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNC_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidRead)} != NULL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataConditionCheckReadFncUsed)} == TRUE)		
Parameters	ErrorCode		
	Type	Dcm_NegativeResponseCodeType	
	Direction	OUT	
	Comment	–	
	Variation	–	
Possible Errors	E_OK E_NOT_OK		

Operation	FreezeCurrentState		
Comment	–		
Relates to	Xxx_FreezeCurrentState		
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreezeCurrentState)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} != DCM_CONTROLMASK_EXTERNAL)		
Parameters	OpStatus		
	Type	Dcm_OpStatusType	
	Direction	IN	





	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	FreezeCurrentState	
Comment	–	
Relates to	Xxx_FreezeCurrentState	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ ASYNCH_CLIENT_SERVER USE_DATA_ ASYNCH_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreeze CurrentState)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfo Ref->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_ CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDsp DidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04)	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	FreezeCurrentState	
Comment	–	
Relates to	Xxx_FreezeCurrentState	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ ASYNCH_CLIENT_SERVER USE_DATA_ ASYNCH_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreeze CurrentState)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfo Ref->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_ CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDsp DidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} >= 0x05)	
Parameters	OpStatus	
	Type	Dcm_OpStatusType





	Direction	IN
	Comment	–
	Variation	–
	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	FreezeCurrentState	
Comment	–	
Relates to	Xxx_FreezeCurrentState	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreezeCurrentState)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} != DCM_CONTROLMASK_EXTERNAL)	
Parameters	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	FreezeCurrentState	
Comment	–	
Relates to	Xxx_FreezeCurrentState	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreezeCurrentState)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04)	
Parameters	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT





	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	FreezeCurrentState	
Comment	–	
Relates to	Xxx_FreezeCurrentState	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreezeCurrentState)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} >= 0x05)	
Parameters	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	GetScalingInformation	
Comment	–	
Relates to	Xxx_GetScalingInformation	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ASYNCH_CLIENT_SERVER USE_DATA_ASYNCH_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataInfoRef->DcmDspDataScalingInfoSize)} != NULL)	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	ScalingInfo	
	Type	Dcm_ScalingInfoArray_{Data}Type
	Direction	OUT
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–





Possible Errors	E_OK E_NOT_OK DCM_E_PENDING

Operation	GetScalingInformation	
Comment	–	
Relates to	Xxx_GetScalingInformation	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataInfoRef->DcmDspDataScalingInfoSize)} != NULL)	
Parameters	ScalingInfo	
	Type	Dcm_ScalingInfoArray_{Data}Type
	Direction	OUT
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	ReadData	
Comment	The server is not allowed to return E_NOT_OK, but shall always provide a valid data value (e.g. a default/replacement value in an error-case) to Dcm/Dem nevertheless the signature of the operation includes E_NOT_OK to ensure compatibility between server runnable and RTE Call API, since the RTE may return negative Std_Return values in certain cases (e.g. partition of server stopped)	
Relates to	Xxx_ReadData	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_ASYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidRead)} != NULL)	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	Data	
	Type	Dcm_DataElement_{Data}Type
	Direction	OUT
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME)})
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ReadData
Comment	The server is not allowed to return E_NOT_OK, but shall always provide a valid data value (e.g. a default/replacement value in an error-case) to Dcm/Dem nevertheless the signature of the operation includes E_NOT_OK to ensure compatibility between server runnable and RTE Call API, since the RTE may return negative Std_Return values in certain cases (e.g. partition of server stopped)
Relates to	Xxx_ReadData
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == USE_DATA_ASYNC_CLIENT_SERVER_ERROR) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidRead)) != NULL)
Parameters	OpStatus
	Type Dcm_OpStatusType
	Direction IN
	Comment –
	Variation –
	Data
	Type Dcm_DataElement_{Data}Type
	Direction OUT
	Comment –
	Variation Data = ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME))) ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME)))
	ErrorCode
	Type Dcm_NegativeResponseCodeType
	Direction OUT
	Comment –
	Variation –
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING

Operation	ReadData
Comment	The server is not allowed to return E_NOT_OK, but shall always provide a valid data value (e.g. a default/replacement value in an error-case) to Dcm/Dem nevertheless the signature of the operation includes E_NOT_OK to ensure compatibility between server runnable and RTE Call API, since the RTE may return negative Std_Return values in certain cases (e.g. partition of server stopped)
Relates to	Xxx_ReadData
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == USE_DATA_SYNC_CLIENT_SERVER) && (((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidRead)) != NULL) ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01.DcmDspPidDataUsePort)) == USE_DATA_SYNC_CLIENT_SERVER))
Parameters	Data
	Type Dcm_DataElement_{Data}Type
	Direction OUT
	Comment –
	Variation Data = ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME))) ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME)))
Possible Errors	E_OK E_NOT_OK

Operation	ReadDataLength
Comment	–
Relates to	Xxx_ReadDataLength
Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == (USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidRead)) != NULL) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType) == UINT8_DYN)}
Parameters	OpStatus
	Type Dcm_OpStatusType
	Direction IN
	Comment –
	Variation –
	DataLength
	Type uint16
	Direction OUT
	Comment –
	Variation –
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING

Operation	ReadDataLength
Comment	–
Relates to	Xxx_ReadDataLength
Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == USE_DATA_SYNC_CLIENT_SERVER) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidRead)) != NULL) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType) == UINT8_DYN)}
Parameters	DataLength
	Type uint16
	Direction OUT
	Comment –
	Variation –
Possible Errors	E_OK E_NOT_OK

Operation	ResetToDefault
Comment	–
Relates to	Xxx_ResetToDefault
Variation	{(ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)) == (USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)) == TRUE) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)) != DCM_CONTROLMASK_EXTERNAL)
Parameters	OpStatus
	Type Dcm_OpStatusType
	Direction IN
	Comment –
	Variation –
	ErrorCode





	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ResetToDefault	
Comment	–	
Relates to	Xxx_ResetToDefault	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ ASYNCH_CLIENT_SERVER USE_DATA_ ASYNCH_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04)	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ResetToDefault	
Comment	–	
Relates to	Xxx_ResetToDefault	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ ASYNCH_CLIENT_SERVER USE_DATA_ ASYNCH_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} >= 0x05)	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–





	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ResetToDefault	
Comment	–	
Relates to	Xxx_ResetToDefault	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} != DCM_CONTROLMASK_EXTERNAL)	
Parameters	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	ResetToDefault	
Comment	–	
Relates to	Xxx_ResetToDefault	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04)	
Parameters	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–





Possible Errors	E_OK E_NOT_OK	
Operation	ResetToDefault	
Comment	–	
Relates to	Xxx_ResetToDefault	
Variation	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} >= 0x05))	
Parameters	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	ReturnControlToECU	
Comment	–	
Relates to	Xxx_ReturnControlToECU	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_SYNCH_CLIENT_SERVER USE_DATA_ASYNCH_CLIENT_SERVER USE_DATA_ASYNCH_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreezeCurrentState)} == TRUE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)} == TRUE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} != DCM_CONTROLMASK_EXTERNAL)	
Parameters	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	ReturnControlToECU	
Comment	–	
Relates to	Xxx_ReturnControlToECU	





Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreezeCurrentState)} == TRUE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)} == TRUE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04)	
Parameters	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	ReturnControlToECU	
Comment	–	
Relates to	Xxx_ReturnControlToECU	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidFreezeCurrentState)} == TRUE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidResetToDefault)} == TRUE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} >= 0x05)	
Parameters	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	ShortTermAdjustment	
Comment	–	
Relates to	Xxx_ShortTermAdjustment	





Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType) != UINT8_DYN} && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} != DCM_CONTROLMASK_EXTERNAL)	
Parameters	ControlStateInfo	
	Type	Dcm_DataArrayTypeUint8_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ShortTermAdjustment	
Comment	–	
Relates to	Xxx_ShortTermAdjustment	
Variation	<pre> ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ ASYNCH_CLIENT_SERVER USE_DATA_ASYNCH_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShort TermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDsp DataType) != UINT8_DYN) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDid InfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_ CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDsp DidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04) </pre>	
Parameters	ControlStateInfo	
	Type	Dcm_DataArrayTypeUint8_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–





	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ShortTermAdjustment	
Comment	–	
Relates to	Xxx_ShortTermAdjustment	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ ASYNCH_CLIENT_SERVER USE_DATA_ ASYNCH_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShort TermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDsp Data Type) != UINT8_DYN) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDid InfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_ CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDsp DidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} >= 0x05)	
Parameters	ControlStateInfo	
	Type	Dcm_DataArrayTypeUint8_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ShortTermAdjustment	
Comment	–	
Relates to	Xxx_ShortTermAdjustment	





Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == UINT8_DYN) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} != DCM_CONTROLMASK_EXTERNAL)	
Parameters	ControlStateInfo	
	Type	Dcm_DataArrayTypeUint8_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	DataLength	
	Type	uint16
	Direction	IN
	Comment	–
	Variation	–
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ShortTermAdjustment	
Comment	–	
Relates to	Xxx_ShortTermAdjustment	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == UINT8_DYN) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04)	
Parameters	ControlStateInfo	
	Type	Dcm_DataArrayTypeUint8_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	DataLength	
	Type	uint16
	Direction	IN
	Comment	–





	Variation	–
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ShortTermAdjustment	
Comment	–	
Relates to	Xxx_ShortTermAdjustment	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ ASYNCH_CLIENT_SERVER USE_DATA_ ASYNCH_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShort TermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDsp DataType)} == UINT8_DYN) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDid InfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_ CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDsp DidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} >= 0x05)	
Parameters	ControlStateInfo	
	Type	Dcm_DataArrayTypeUint8_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	DataLength	
	Type	uint16
	Direction	IN
	Comment	–
	Variation	–
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN





	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ShortTermAdjustment	
Comment	–	
Relates to	Xxx_ShortTermAdjustment	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType) != UINT8_DYN) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} != DCM_CONTROLMASK_EXTERNAL)	
Parameters	ControlStateInfo	
	Type	Dcm_DataArrayTypeUint8_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	ShortTermAdjustment	
Comment	–	
Relates to	Xxx_ShortTermAdjustment	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType) != UINT8_DYN) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04)	
Parameters	ControlStateInfo	
	Type	Dcm_DataArrayTypeUint8_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	controlMask	
	Type	Dcm_ControlMask_{DID}Type





	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	ShortTermAdjustment	
Comment	–	
Relates to	Xxx_ShortTermAdjustment	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == UINT8_DYN) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} != DCM_CONTROLMASK_EXTERNAL)	
Parameters	ControlStateInfo	
	Type	Dcm_DataArrayTypeUint8_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	DataLength	
	Type	uint16
	Direction	IN
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	ShortTermAdjustment	
Comment	–	
Relates to	Xxx_ShortTermAdjustment	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == UINT8_DYN) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} <= 0x04)	
Parameters	ControlStateInfo	
	Type	Dcm_DataArrayTypeUint8_{Data}Type





	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	DataLength	
	Type	uint16
	Direction	IN
	Comment	–
	Variation	–
	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	ShortTermAdjustment	
Comment	–	
Relates to	Xxx_ShortTermAdjustment	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidControl/DcmDspDidShortTermAdjustment)} == TRUE) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == UINT8_DYN) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMask)} == DCM_CONTROLMASK_EXTERNAL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl/DcmDspDidControlMaskSize)} >= 0x05)	
Parameters	ControlStateInfo	
	Type	Dcm_DataArrayTypeUint8_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	DataLength	
	Type	uint16
	Direction	IN
	Comment	–
	Variation	–
	controlMask	
	Type	Dcm_ControlMask_{DID}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	





	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	WriteData	
Comment	–	
Relates to	Xxx_WriteData	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidWrite)} != NULL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} != UINT8_DYN)	
Parameters	Data	
	Type	Dcm_DataElement_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	WriteData	
Comment	–	
Relates to	Xxx_WriteData	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == (USE_DATA_ASYNC_CLIENT_SERVER USE_DATA_ASYNC_CLIENT_SERVER_ERROR)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidWrite)} != NULL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == UINT8_DYN)	
Parameters	Data	
	Type	Dcm_DataElement_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	DataLength	
	Type	uint16
	Direction	IN
	Comment	–





	Variation	–
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	WriteData	
Comment	–	
Relates to	Xxx_WriteData	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidWrite)} != NULL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} != UINT8_DYN)	
Parameters	Data	
	Type	Dcm_DataElement_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	WriteData	
Comment	–	
Relates to	Xxx_WriteData	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataUsePort)} == USE_DATA_SYNCH_CLIENT_SERVER) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidInfo/DcmDspDidWrite)} != NULL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.DcmDspDataType)} == UINT8_DYN)	
Parameters	Data	
	Type	Dcm_DataElement_{Data}Type
	Direction	IN
	Comment	–
	Variation	Data = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)})
	DataLength	
	Type	uint16





	Direction	IN
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

J

One DataServices interface will be generated for each Data of each DID/PID, with following possible operations:

8.8.3.2.1 ReadData

ReadData allows requesting to the application a data value of a DID/PID. A ReadData interface is defined for every data of each DID/PID with read access when the data element is configured to use a C/S interface. If the length is variable, the operation ReadDataLength has to provide the current valid data length of this Data.

This interface is used for [UDS](#) Service ReadDataByIdentifier and for [UDS](#) Service ReadDataByPeriodicIdentifier (0x2A) and for [UDS](#) Service InputOutputControlByIdentifier (0x2F).

The ReadData interface can be defined as synchronous or asynchronous according to configuration parameter [DcmDspDataUsePort](#). The synchronous mechanism of the ReadData interface is compatible to the related DEM interface to allow the provider to use the same interface for both [Dcm](#) and DEM.

8.8.3.2.2 WriteData

WriteData requests the application to write a data value of a DID. The Data specific type is an array of uint8 which represent either the fix length of this Data or the maximum possible length of this Data. A WriteData interface is defined for every data of each [DID](#) with write access. This interface is used for the [UDS](#) Service WriteDataByIdentifier (0x2E).

8.8.3.2.3 ReadDataLength

ReadDataLength requests the application to return the data length of a Data. A ReadDataLength interface is defined for every data of each [DID](#) with variable data length.

This interface is used for [UDS](#) Service `ReadDataByIdentifier` and for [UDS](#) Service `ReadDataByPeriodicIdentifier` (0x2A).

8.8.3.2.4 ConditionCheckRead

`ConditionCheckRead` requests to the application if the conditions (System state,...) to read the Data are correct. This operation is called for all requested DIDs before requesting the data of each DID. A `ConditionCheckRead` interface is defined for every data of each [DID](#) with read access.

8.8.3.2.5 GetScalingInformation

Request to the application for the scaling information of a Data (see [\[SWS_Dcm_00394\]](#)).

8.8.3.2.6 ReturnControlToEcu

Request to the application to return control to ECU of an `IOControl` (see [\[SWS_Dcm_00396\]](#)).

8.8.3.2.7 ResetToDefault

Request to the application to reset an `IOControl` to default value (see [\[SWS_Dcm_00397\]](#)).

8.8.3.2.8 FreezeCurrentState

Request to the application to freeze the current state of an `IOControl` (see [\[SWS_Dcm_00398\]](#)).

8.8.3.2.9 ShortTermAdjustment

Request to the application to adjust the IO signal (see [\[SWS_Dcm_00399\]](#)).

8.8.3.3 DataServices_DIDRange_{Range}

The following interface defines an operation needed to get the [DID](#) range. Using the concepts of the [SW-C](#) template, the interface is defined as follows:

[SWS_Dcm_00769] Definition of ClientServerInterface DataServices_DIDRange_{Range} [

Name	DataServices_DIDRange_{Range}		
Comment	–		
IsService	true		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRange.DcmDspDidRangeUsePort)} == TRUE Range = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRange.SHORT-NAME)})		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	10	DCM_E_PENDING	Request is not yet finished. Further call(s) required to finish.

Operation	IsDidAvailable		
Comment	–		
Relates to	Xxx_IsDidAvailable		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRange.DcmDspDidRangeHasGaps)} == TRUE		
Parameters	DID		
	Type	uint16	
	Direction	IN	
	Comment	–	
	Variation	–	
	OpStatus		
	Type	Dcm_OpStatusType	
	Direction	IN	
	Comment	–	
	Variation	–	
	supported		
	Type	Dcm_DidSupportedType	
	Direction	OUT	
	Comment	–	
Variation	–		
Possible Errors	E_OK DCM_E_PENDING		

Operation	ReadDidData	
Comment	–	
Relates to	Xxx_ReadDidData	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRange/DcmDspDidRangeInfoRef->DcmDspDidRead)}) != NULL)	
Parameters	DID	
	Type	uint16
	Direction	IN
	Comment	–
	Variation	–
	Data	
	Type	Dcm_RangeArray_{Range}Type
	Direction	OUT
	Comment	–





	Variation	Range = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRange.SHORT-NAME)})
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataLength	
	Type	uint16
	Direction	OUT
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ReadDidRangeDataLength	
Comment	–	
Relates to	Xxx_ReadDidRangeDataLength	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRange/DcmDspDidRangeInfoRef->DcmDspDidRead)}) != NULL	
Parameters	DID	
	Type	uint16
	Direction	IN
	Comment	–
	Variation	–
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataLength	
	Type	uint16
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	WriteDidData	
Comment	–	
Relates to	Xxx_WriteDidData	





Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRange/DcmDspDidRangeInfoRef->DcmDspDidWrite)} != NULL)	
Parameters	DID	
	Type	uint16
	Direction	IN
	Comment	–
	Variation	–
	Data	
	Type	Dcm_RangeArray_{Range}Type
	Direction	IN
	Comment	–
	Variation	Range = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRange.SHORT-NAME)})
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataLength	
	Type	uint16
	Direction	IN
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

」

8.8.3.4 InfotypeServices_{VehInfoData}

The following interface defines an operation needed to get data from one or several [SW-C](#) in order to supply [OBD](#) Service \$09 (see [[SWS_Dcm_00423](#)]).

Using the concepts of the [SW-C](#) template, the interface is defined as follows:

[SWS_Dcm_00688] Definition of ClientServerInterface InfotypeServices_{VehInfoData} 「

Name	InfotypeServices_{VehInfoData}		
Comment	–		
IsService	true		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspVehInfo/DcmDspVehInfoData/DcmDspVehInfoDataUsePort)}==TRUE VehInfoData = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspVehInfo/DcmDspVehInfoData.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	10	DCM_E_PENDING	Request is not yet finished. Further call(s) required to finish.

Operation	GetInfotypeValueData		
Comment	–		
Relates to	Xxx_GetInfotypeValueData		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspVehInfo/DcmDspVehInfoData/DcmDspVehInfoDataSize)} < 256		
Parameters	OpStatus		
	Type	Dcm_OpStatusType	
	Direction	IN	
	Comment	–	
	Variation	–	
	DataValueBuffer		
	Type	Dcm_InfoTypeServicesArray_{VehInfoData}Type	
	Direction	OUT	
	Comment	–	
	Variation	VehInfoData = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspVehInfo/DcmDspVehInfoData.SHORT-NAME)}	
	DataValueBufferSize		
	Type	uint8	
	Direction	INOUT	
	Comment	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The function returns the actual number of written data bytes in DataValueBuffer	
	Variation	–	
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING		

Operation	GetInfotypeValueData	
Comment	–	
Relates to	Xxx_GetInfotypeValueData	
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspVehInfo/DcmDspVehInfoData/DcmDspVehInfoDataSize)} >= 256	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataValueBuffer	





	Type	Dcm_InfoTypeServicesArray_{VehInfoData}Type
	Direction	OUT
	Comment	–
	Variation	VehInfoData = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspVehInfo/DcmDspVehInfoData.SHORT-NAME)}
	DataValueBufferSize	
	Type	uint16
	Direction	INOUT
	Comment	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The function returns the actual number of written data bytes in DataValueBuffer
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

]

8.8.3.5 RoutineServices_{RoutineName}

The following interface defines operations needed for the [UDS](#) Service RoutineControl (0x31) (see [[SWS_Dcm_00400](#)], [[SWS_Dcm_00401](#)], [[SWS_Dcm_00402](#)], [[SWS_Dcm_00403](#)], [[SWS_Dcm_00404](#)], [[SWS_Dcm_00405](#)]).

Using the concepts of the [SW-C](#) template, the interface is defined as follows:

[SWS_Dcm_00690] Definition of ClientServerInterface RoutineServices_{Routine Name} [

Name	RoutineServices_{RoutineName}		
Comment	–		
IsService	true		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.DcmDspRoutineUsePort)} == TRUE RoutineName = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	10	DCM_E_PENDING	Request is not yet finished. Further call(s) required to finish.
	12	DCM_E_FORCE_RCRP	application request the transmission of a response Response Pending (NRC 0x78)

Operation	RequestResults
Comment	–
Relates to	Xxx_RequestResults





Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)) != VARIABLE_LENGTH) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)) != VARIABLE_LENGTH)	
Parameters	DataIn_{Signal}	
	Type	Dcm_RequestDataIn_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataOut_{Signal}	
	Type	Dcm_RequestDataOut_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	RequestResults	
Comment	–	
Relates to	Xxx_RequestResults	
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)) == VARIABLE_LENGTH) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)) != VARIABLE_LENGTH)	
Parameters	DataIn_{Signal}	
	Type	Dcm_RequestDataIn_{Routine}_{Signal}Type
	Direction	IN
	Comment	–





	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataOut_{Signal}	
	Type	Dcm_RequestDataOut_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	DataOut_{Signal}	
	Type	Dcm_RequestFlexibleOutArrayData_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	currentDataLength	
	Type	uint16
	Direction	OUT
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	–	
Operation	RequestResults	
Comment	–	
Relates to	Xxx_RequestResults	





Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)) != VARIABLE_LENGTH) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)) == VARIABLE_LENGTH)	
Parameters	DataIn_{Signal}	
	Type	Dcm_RequestDataIn_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	DataIn_{Signal}	
	Type	Dcm_RequestFlexibleInArrayData_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataOut_{Signal}	
	Type	Dcm_RequestDataOut_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	–	

Operation	RequestResults
Comment	–
Relates to	Xxx_RequestResults
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)) == VARIABLE_LENGTH) && ((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.DcmDspRoutineSignalType)) == VARIABLE_LENGTH)





Parameters	DataIn_{Signal}	
	Type	Dcm_RequestDataIn_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsIn/DcmDspRequestRoutineResultsInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	DataIn_{Signal}	
	Type	Dcm_StartFlexibleInArrayData_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	–
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataOut_{Signal}	
	Type	Dcm_RequestDataOut_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	DataOut_{Signal}	
	Type	Dcm_RequestFlexibleOutArrayData_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsOut/DcmDspRequestRoutineResultsOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	currentDataLength	
	Type	uint16
	Direction	OUT
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType





	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	RequestResultsConfirmation	
Comment	This operation indicates the transmission of a response to a RequestResultsRoutine request	
Relates to	Xxx_RequestResultsConfirmation	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspRequestRoutineResults/DcmDspRequestRoutineResultsConfirmationEnabled)})==TRUE)	
Parameters	ConfirmationStatus	
	Type	Dcm_ConfirmationStatusType
	Direction	IN
	Comment	Confirmation status of a RequestResultsRoutinerequest
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	Start	
Comment	–	
Relates to	Xxx_Start	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH)	
Parameters	DataIn_{Signal}	
	Type	Dcm_StartDataIn_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataOut_{Signal}	
	Type	Dcm_StartDataOut_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–





	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	Start	
Comment	–	
Relates to	Xxx_Start	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH)	
Parameters	DataIn_{Signal}	
	Type	Dcm_StartDataIn_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)} Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.SHORT-NAME)}
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataOut_{Signal}	
	Type	Dcm_StartDataOut_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	DataOut_{Signal}	
	Type	Dcm_StartFlexibleOutArrayData_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	–





	currentDataLength	
	Type	uint16
	Direction	OUT
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRPP	

Operation	Start	
Comment	–	
Relates to	Xxx_Start	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH)	
Parameters	DataIn_{Signal}	
	Type	Dcm_StartDataIn_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	DataIn_{Signal}	
	Type	Dcm_StartFlexibleInArrayData_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	–
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataOut_{Signal}	
	Type	Dcm_StartDataOut_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–





	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	currentDataLength	
	Type	uint16
	Direction	IN
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	Start	
Comment	–	
Relates to	Xxx_Start	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH)	
Parameters	DataIn_{Signal}	
	Type	Dcm_StartDataIn_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	DataIn_{Signal}	
	Type	Dcm_StartFlexibleInArrayData_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineIn/DcmDspStartRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	OpStatus	





	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataOut_{Signal}	
	Type	Dcm_StartDataOut_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	DataOut_{Signal}	
	Type	Dcm_StartFlexibleOutArrayData_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineOut/DcmDspStartRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	currentDataLength	
	Type	uint16
	Direction	INOUT
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRP	

Operation	StartConfirmation	
Comment	This operation indicates the transmission of a response to a StartRoutine request	
Relates to	Xxx_StartConfirmation	
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStartRoutine/DcmDspStartRoutineConfirmationEnabled)}==TRUE	
Parameters	ConfirmationStatus	
	Type	Dcm_ConfirmationStatusType
	Direction	IN
	Comment	Confirmation status of a StartRoutine request





	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	Stop	
Comment	–	
Relates to	Xxx_Stop	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH)	
Parameters	DataIn_{Signal}	
	Type	Dcm_StopDataIn_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataOut_{Signal}	
	Type	Dcm_StopDataOut_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	Stop	
Comment	–	
Relates to	Xxx_Stop	





Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH)	
Parameters	DataIn_{Signal}	
	Type	Dcm_StopDataIn_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataOut_{Signal}	
	Type	Dcm_StopDataOut_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	DataOut_{Signal}	
	Type	Dcm_StopFlexibleOutArrayData_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	–
	currentDataLength	
	Type	uint16
	Direction	OUT
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	Stop
Comment	–
Relates to	Xxx_Stop





Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH)	
Parameters	DataIn_{Signal}	
	Type	Dcm_StopDataIn_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	DataIn_{Signal}	
	Type	Dcm_StopFlexibleInArrayData_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	–
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataOut_{Signal}	
	Type	Dcm_StopDataOut_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	currentDataLength	
	Type	uint16
	Direction	IN
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	Stop
Comment	–
Relates to	Xxx_Stop





Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH)	
Parameters	DataIn_{Signal}	
	Type	Dcm_StopDataIn_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	DataIn_{Signal}	
	Type	Dcm_StopFlexibleInArrayData_{Routine}_{Signal}Type
	Direction	IN
	Comment	–
	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineIn/DcmDspStopRoutineInSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	–
	Variation	–
	DataOut_{Signal}	
	Type	Dcm_StopDataOut_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–
	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} != VARIABLE_LENGTH Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	DataOut_{Signal}	
	Type	Dcm_StopFlexibleOutArrayData_{Routine}_{Signal}Type
	Direction	OUT
	Comment	–





	Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.DcmDspRoutineSignalType)} == VARIABLE_LENGTH Signal = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineOut/DcmDspStopRoutineOutSignal.SHORT-NAME)} Routine = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}
	currentDataLength	
	Type	uint16
	Direction	INOUT
	Comment	–
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	StopConfirmation	
Comment	This operation indicates the transmission of a response to a StopRoutine request	
Relates to	Xxx_StopConfirmation	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine/DcmDspStopRoutine/DcmDspStopRoutineConfirmationEnabled)})==TRUE)	
Parameters	ConfirmationStatus	
	Type	Dcm_ConfirmationStatusType
	Direction	IN
	Comment	Confirmation status of a StopRoutine request
	Variation	–
Possible Errors	E_OK E_NOT_OK	

┌

From the point of view of the DCM, the operations have the following signatures:

8.8.3.6 RequestControlServices_{Tid}

The following interface allows the [Dcm](#) to provide [OBD](#) Service \$08 (see [\[SWS_Dcm_00419\]](#)).

Using the concepts of the [SW-C](#) template, the interface is defined as follows:

[SWS_Dcm_00691] Definition of ClientServerInterface RequestControlServices_{Tid} ┌

Name	RequestControlServices_{Tid}		
Comment	–		
IsService	true		
Variation	Tid = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestControl.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed

Operation	RequestControl	
Comment	–	
Relates to	Xxx_RequestControl	
Variation	–	
Parameters	OutBuffer	
	Type	Dcm_RequestControlServicesOutArray_{Tid}Type
	Direction	OUT
	Comment	–
	Variation	Tid = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestControl.SHORT-NAME)}
	InBuffer	
	Type	Dcm_RequestControlServicesInArray_{Tid}Type
	Direction	IN
	Comment	–
	Variation	Tid = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestControl.SHORT-NAME)}
Possible Errors	E_OK E_NOT_OK	

]

8.8.3.7 ServiceRequestNotification

The following interface indicates to the Application that a service is about to be executed and allows the Application to reject the execution of the service request (see [\[SWS_Dcm_00218\]](#), [\[SWS_Dcm_00462\]](#), [\[SWS_Dcm_00463\]](#)).

Using the concepts of the [SW-C](#) template, the interface is defined as follows:

[SWS_Dcm_00694] Definition of ClientServerInterface ServiceRequestNotification

Name	ServiceRequestNotification		
Comment	–		
IsService	true		
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsd/DcmDsdServiceRequestManufacturerNotification)} != NULL) ({ecuc(Dcm/DcmConfigSet/DcmDsd/DcmDsdServiceRequestSupplierNotification)} != NULL)		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed





	8	E_REQUEST_NOT_ACCEPTED	no response will be sent

Operation	Confirmation	
Comment	–	
Relates to	Xxx_Confirmation	
Variation	–	
Parameters	SID	
	Type	uint8
	Direction	IN
	Comment	Value of service identifier
	Variation	–
	ReqType	
	Type	uint8
	Direction	IN
	Comment	Addressing type of the request(0=physical request, 1=functional request)
	Variation	–
	ConnectionId	
	Type	uint16
	Direction	IN
	Comment	Unique connection identifier
	Variation	–
	ConfirmationStatus	
	Type	Dcm_ConfirmationStatusType
	Direction	IN
	Comment	Confirmation of a successful transmission or a transmission error of a diagnostic service.
	Variation	–
	ProtocolType	
	Type	Dcm_ProtocolType
	Direction	IN
	Comment	–
	Variation	–
	TesterSourceAddress	
	Type	uint16
	Direction	IN
Comment	–	
Variation	–	
Possible Errors	E_OK E_NOT_OK	

Operation	Indication		
Comment	–		
Relates to	Xxx_Indication		
Variation	–		
Parameters	SID		
	Type	uint8	
	Direction	IN	





	Comment	Value of service identifier
	Variation	–
	RequestData	
	Type	Dcm_RequestDataArrayType
	Direction	IN
	Comment	This parameter contains the complete request data (diagnostic buffer), except the service ID
	Variation	–
	DataSize	
	Type	uint32
	Direction	IN
	Comment	This parameter defines how many bytes in the RequestData parameter are valid
	Variation	–
	ReqType	
	Type	uint8
	Direction	IN
	Comment	Addressing type of the request(0=physical request, 1=functional request)
	Variation	–
	ConnectionId	
	Type	uint16
	Direction	IN
	Comment	Unique connection identifier
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	–
	Variation	–
	ProtocolType	
	Type	Dcm_ProtocolType
	Direction	IN
	Comment	–
	Variation	–
	TesterSourceAddress	
	Type	uint16
	Direction	IN
	Comment	–
	Variation	–
	Possible Errors	E_OK E_NOT_OK E_REQUEST_NOT_ACCEPTED

]

8.8.3.8 UploadDownloadServices

[SWS_Dcm_91065] Definition of ClientServerInterface UploadDownloadServices

Upstream requirements: [RS_Diag_04135](#)

Name	UploadDownloadServices		
Comment	–		
IsService	true		
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspMemoryTransfer.DcmDspMemoryTransferUsePort)} == TRUE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestFileTransfer/DcmRequestFileTransferUsePort)} == TRUE)		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	10	DCM_E_PENDING	Request is not yet finished. Further call(s) required to finish.
	12	DCM_E_FORCE_RCRRP	application request the transmission of a response Response Pending (NRC 0x78)

Operation	ProcessRequestAddFile		
Comment	Callout function. DCM shall call this function to start a RequestFileTransfer process with modeOfOperation equal to 0x01 (AddFile).		
Relates to	Dcm_ProcessRequestAddFile		
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestFileTransfer/DcmRequestFileTransferUsePort)} == TRUE)		
Parameters	OpStatus		
	Type	Dcm_OpStatusType	
	Direction	IN	
	Comment	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRRP_OK: All In-parameters are set to 0x00.	
	Variation	–	
	filePathAndNameLength		
	Type	uint16	
	Direction	IN	
	Comment	Defines the length in bytes for the parameter filePathAndName.	
	Variation	–	
	filePathAndName		
	Type	const Dcm_FileAndDirNameType	
	Direction	IN	
	Comment	Defines the file system location of the server where the file which shall be added, deleted, replaced or read from depending on the parameter modeOfOperation parameter. In addition this parameter includes the file name of the file which shall be added, deleted, replaced or read as part of the file path.	
	Variation	–	
	dataFormatIdentifier		
	Type	uint8	
	Direction	IN	





	Comment	This data-parameter is a one byte value with each nibble encoded separately. The high nibble specifies the "compressionMethod", and the low nibble specifies the "encryptingMethod". The value 0x00 specifies that neither compressionMethod nor encryptingMethod is used. Values other than 0x00 are vehicle manufacturer specific.
	Variation	–
	fileSizeUncompressed	
	Type	uint64
	Direction	IN
	Comment	Defines the size of the uncompressed file to be download in bytes.
	Variation	–
	fileSizeCompressed	
	Type	uint64
	Direction	IN
	Comment	Defines the size of the compressed file to be downloaded in bytes.
	Variation	–
	maxNumberOfBlockLength	
	Type	uint64
	Direction	OUT
	Comment	Max number of bytes to be included in each TransferData request excluding the SID and the blockSequenceCounter.
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	If the operation Dcm_ProcessRequestAddFile returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	ProcessRequestDeleteFile	
Comment	Callout function. DCM shall call this function to start a RequestFileTransfer process with modeOfOperation equal to 0x02 (DeleteFile).	
Relates to	Dcm_ProcessRequestDeleteFile	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestFileTransfer/DcmRequestFileTransferUsePort)}) == TRUE)	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRRP_OK: All In-parameters are set to 0x00.
	Variation	–
filePathAndNameLength		





	Type	uint16
	Direction	IN
	Comment	Defines the length in bytes for the parameter filePathAndName.
	Variation	–
	filePathAndName	
	Type	const Dcm_FileAndDirNameType
	Direction	IN
	Comment	Defines the file system location of the server where the file which shall be added, deleted, replaced or read from depending on the parameter modeOfOperation parameter. In addition this parameter includes the file name of the file which shall be added, deleted, replaced or read as part of the file path.
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	If the operation Dcm_ProcessRequestDeleteFile returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	ProcessRequestDownload	
Comment	Callout function. DCM shall call this callout function to start a download process. This service is needed for the implementation of UDS service RequestDownload.	
Relates to	Dcm_ProcessRequestDownload	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspMemoryTransfer.DcmDspMemoryTransferUsePort)})	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	DCM_INITIAL DCM_PENDING DCM_CANCEL
	Variation	–
	DataFormatIdentifier	
	Type	uint8
	Direction	IN
	Comment	Bit 7 - 4: Compression Method • 0x0: not compressed • 0x1..F: vehicle-manufacturer-specific Bit 3 - 0: Encrypting method • 0x0: not encrypted • 0x1..F: vehicle-manufacturer-specific
	Variation	–
	MemoryIdentifier	
	Type	uint8





	Direction	IN
	Comment	Identifier of the Memory Block, if the parameter is not used it shall be set to 0.
	Variation	–
	MemoryAddress	
	Type	uint32
	Direction	IN
	Comment	Starting address of server memory to which data is to be written
	Variation	–
	MemorySize	
	Type	uint32
	Direction	IN
	Comment	Uncompressed memory size in bytes
	Variation	–
	BlockLength	
	Type	uint32
	Direction	INOUT
	Comment	Max. Number of bytes for one Dcm_WriteMemory
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	If the operation Dcm_ProcessRequestDownload returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ProcessRequestReadDir	
Comment	Callout function. DCM shall call this function to start a RequestFileTransfer process with modeOfOperation equal to 0x05 (ReadDir).	
Relates to	Dcm_ProcessRequestReadDir	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestFileTransfer/DcmRequestFileTransferUsePort)} == TRUE)	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRRP_OK: All In-parameters are set to 0x00.
	Variation	–
	filePathAndNameLength	
	Type	uint16
	Direction	IN
	Comment	Defines the length in bytes for the parameter filePathAndName.
	Variation	–





	filePathAndName	
	Type	const Dcm_FileAndDirNameType
	Direction	IN
	Comment	Defines the file system location of the server where the file which shall be added, deleted, replaced or read from depending on the parameter modeOfOperation parameter. In addition this parameter includes the file name of the file which shall be added, deleted, replaced or read as part of the file path.
	Variation	–
	dirInfoLength	
	Type	uint64
	Direction	OUT
	Comment	Defines the size of directory information to be uploaded in bytes.
	Variation	–
	maxNumberOfBlockLength	
	Type	uint64
	Direction	OUT
	Comment	Max number of bytes to be included in each TransferData request excluding the SID and the blockSequenceCounter.
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	If the operation Dcm_ProcessRequestReadDir returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRP	

Operation	ProcessRequestReadFile	
Comment	Callout function. DCM shall call this function to start a RequestFileTransfer process with modeOfOperation equal to 0x04 (ReadFile).	
Relates to	Dcm_ProcessRequestReadFile	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestFileTransfer/DcmRequestFileTransferUsePort)} == TRUE)	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRP_OK: All In-parameters are set to 0x00.
	Variation	–
	filePathAndNameLength	
	Type	uint16
	Direction	IN
	Comment	Defines the length in bytes for the parameter filePathAndName.





	Variation	–
	filePathAndName	
	Type	const Dcm_FileAndDirNameType
	Direction	IN
	Comment	Defines the file system location of the server where the file which shall be added, deleted, replaced or read from depending on the parameter modeOfOperation parameter. In addition this parameter includes the file name of the file which shall be added, deleted, replaced or read as part of the file path.
	Variation	–
	dataFormatIdentifier	
	Type	uint8
	Direction	IN
	Comment	This data-parameter is a one byte value with each nibble encoded separately. The high nibble specifies the "compressionMethod", and the low nibble specifies the "encryptingMethod". The value 0x00 specifies that neither compressionMethod nor encryptingMethod is used. Values other than 0x00 are vehicle manufacturer specific.
	Variation	–
	fileSizeUncompressed	
	Type	uint64
	Direction	OUT
	Comment	Defines the size of the uncompressed file to be uploaded in bytes.
	Variation	–
	fileSizeCompressed	
	Type	uint64
	Direction	OUT
	Comment	Defines the size of the compressed file to be uploaded in bytes.
	Variation	–
	maxNumberOfBlockLength	
	Type	uint64
	Direction	OUT
	Comment	Max number of bytes to be included in each TransferData response excluding the SID and the blockSequenceCounter.
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	If the operation Dcm_ProcessRequestReadFile returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	
Operation	ProcessRequestReplaceFile	
Comment	Callout function. DCM shall call this function to start a RequestFileTransfer process with modeOfOperation equal to 0x03 (ReplaceFile).	





Relates to	Dcm_ProcessRequestReplaceFile	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestFileTransfer/DcmRequestFileTransferUsePort)} == TRUE)	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRRP_OK: All In-parameters are set to 0x00.
	Variation	–
	filePathAndNameLength	
	Type	uint16
	Direction	IN
	Comment	Defines the length in bytes for the parameter filePathAndName.
	Variation	–
	filePathAndName	
	Type	const Dcm_FileAndDirNameType
	Direction	IN
	Comment	Defines the file system location of the server where the file which shall be added, deleted, replaced or read from depending on the parameter modeOfOperation parameter. In addition this parameter includes the file name of the file which shall be added, deleted, replaced or read as part of the file path.
	Variation	–
	dataFormatIdentifier	
	Type	uint8
	Direction	IN
	Comment	This data-parameter is a one byte value with each nibble encoded separately. The high nibble specifies the "compressionMethod", and the low nibble specifies the "encryptingMethod". The value 0x00 specifies that neither compressionMethod nor encryptingMethod is used. Values other than 0x00 are vehicle manufacturer specific.
	Variation	–
	fileSizeUncompressed	
	Type	uint64
	Direction	IN
	Comment	Defines the size of the uncompressed file to be download in bytes.
	Variation	–
	fileSizeCompressed	
	Type	uint64
	Direction	IN
	Comment	Defines the size of the compressed file to be downloaded in bytes.
	Variation	–
	maxNumberOfBlockLength	
	Type	uint64
	Direction	OUT
	Comment	Max number of bytes to be included in each TransferData request excluding the SID and the blockSequenceCounter.





	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	If the operation Dcm_ProcessRequestReplaceFile returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRP	

Operation	ProcessRequestTransferExit	
Comment	Callout function. DCM shall call this callout function to terminate a download or upload process. This callout is needed for the implementation of UDS service RequestTransferExit.	
Relates to	Dcm_ProcessRequestTransferExit	
Variation	–	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	DCM_INITIAL DCM_PENDING DCM_CANCEL
	Variation	–
	transferRequestParameterRecord	
	Type	Dcm_RequestDataArrayType
	Direction	IN
	Comment	Vehicle manufacturer specific transferRequestParameterRecord
	Variation	–
	transferRequestParameterRecordSize	
	Type	uint32
	Direction	IN
	Comment	Length of transferRequestParameterRecord in bytes
	Variation	–
	transferResponseParameterRecord	
	Type	Dcm_RequestDataArrayType
	Direction	OUT
	Comment	Vehicle manufacturer specific transferResponseParameterRecord
	Variation	–
	transferResponseParameterRecordSize	
	Type	uint32
	Direction	INOUT
	Comment	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The function returns the actual number of written data bytes in transferResponseParameterRecord.
	Variation	–





	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	see below
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ProcessRequestUpload	
Comment	Callout function. DCM shall call this callout function to start an upload process. This service is needed for the implementation of UDS service RequestUpload.	
Relates to	Dcm_ProcessRequestUpload	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspMemoryTransfer.DcmDspMemoryTransferUse Port)})	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	DCM_INITIAL DCM_PENDING DCM_CANCEL DCM_FORCE_RCRRP_OK
	Variation	–
	DataFormatIdentifier	
	Type	uint8
	Direction	IN
	Comment	Bit 7 - 4: Compression Method • 0x0: not compressed • 0x1..F: vehicle-manufacturer-specific Bit 3 - 0: Encrypting method • 0x0: not encrypted • 0x1..F: vehicle-manufacturer-specific
	Variation	–
	MemoryIdentifier	
	Type	uint8
	Direction	IN
	Comment	Identifier of the Memory Block, if the parameter is not used it shall be set to 0.
	Variation	–
	MemoryAddress	
	Type	uint32
	Direction	IN
	Comment	Starting address of server memory from which data are to be copied
	Variation	–





	MemorySize	
	Type	uint32
	Direction	IN
	Comment	Uncompressed memory size in bytes
	Variation	–
	BlockLength	
	Type	uint32
	Direction	INOUT
	Comment	Max. Number of bytes for one Dcm_ReadMemory
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	If the operation Dcm_ProcessRequestUpload returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING	

Operation	ProcessTransferDataRead	
Comment	The ProcessTransferDataRead callout is used to request memory data identified by the parameter memoryAddress and memorySize from the UDS request message. This service is needed for the implementation of UDS services: • ReadMemoryByAddress • RequestUpload • ReadDataByIdentifier (in case of Dynamical DID defined by memory address) • TransferData	
Relates to	Dcm_ProcessTransferDataRead	
Variation	–	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	DCM_INITIAL DCM_PENDING DCM_CANCEL DCM_FORCE_RCRRP_OK
	Variation	–
	MemoryIdentifier	
	Type	uint8
	Direction	IN
	Comment	Identifier of the Memory Block (e.g. used if memory section distinguishing is needed) Note: If it's not used this parameter shall be set to 0.
	Variation	–
	MemoryAddress	
	Type	uint32
	Direction	IN





	Comment	Starting address of server memory from which data is to be retrieved.
	Variation	–
	MemorySize	
	Type	uint32
	Direction	IN
	Comment	Number of bytes in the MemoryData
	Variation	–
	MemoryData	
	Type	Dcm_RequestDataArrayType
	Direction	OUT
	Comment	Data read (Points to the diagnostic buffer in DCM)
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	If the operation Dcm_ReadMemory returns value DCM_READ_FAILED, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	ProcessTransferDataWrite	
Comment	The ProcessTransferDataWrite callout is used to write memory data identified by the parameter memoryAddress and memorySize. This service is needed for the implementation of UDS services : • WriteMemoryByAddress • RequestDownload • TransferData	
Relates to	Dcm_ProcessTransferDataWrite	
Variation	–	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	DCM_INITIAL DCM_PENDING DCM_CANCEL DCM_FORCE_RCRRP_OK
	Variation	–
	MemoryIdentifier	
	Type	uint8
	Direction	IN
	Comment	Identifier of the Memory Block (e.g. used by WriteDataByIdentifier service). Note: If it's not used this parameter shall be set to 0.
	Variation	–
	MemoryAddress	
	Type	uint32





	Direction	IN
	Comment	Starting address of server memory in which data is to be copied. Note: If it's not used (e.g. if the data is compressed) this parameter shall be set to 0.
	Variation	–
	MemorySize	
	Type	uint32
	Direction	IN
	Comment	Number of bytes in MemoryData
	Variation	–
	MemoryData	
	Type	Dcm_RequestDataArrayType
	Direction	IN
	Comment	Data to write (Points to the diagnostic buffer in DCM)
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
Possible Errors	Direction	OUT
	Comment	If the operation Dcm_WriteMemory returns value DCM_WRITE_FAILED , the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
	Variation	–
	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	ReadFileOrDir	
Comment	Callout function. DCM shall call this function when data shall be sent as a response to UDS service TransferData if there's an ongoing RequestFileTransfer process started with 0x04 (ReadFile) or 0x05 (ReadDir).	
Relates to	Dcm_ReadFileOrDir	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestFileTransfer/DcmRequestFileTransferUsePort)} == TRUE)	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to 0x00. DCM_CANCEL: All In-parameters are set to 0x00. DCM_FORCE_RCRRP_OK: All In-parameters are set to 0x00.
	Variation	–
	DataLength	
	Type	uint64
Direction	INOUT	





	Comment	As in, the parameter defines the maximum block length to be used, i.e. the value of <code>maxNumberOfBlockLength</code> sent to the client in the response of <code>RequestFileTransfer</code> . As out, the parameter defines the actual length in bytes for the parameter <code>Data</code> . The value shall not exceed, but might be less, the value provided as in parameter.
	Variation	–
	Data	
	Type	Dcm_ResponseDataArrayType
	Direction	IN
	Comment	Pointer to the data to be written.
	Variation	–
	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	If the operation <code>Dcm_ReadFileOrDir</code> returns value <code>E_NOT_OK</code> , the DCM module shall send a negative response with NRC code equal to the parameter <code>ErrorCode</code> parameter value.
	Variation	–
Possible Errors	E_OK E_NOT_OK DCM_E_PENDING DCM_E_FORCE_RCRRP	

Operation	WriteFile	
Comment	Callout function. DCM shall call this function when data is received using UDS service <code>TransferData</code> if there's an ongoing <code>RequestFileTransfer</code> process started with <code>0x01</code> (<code>AddFile</code>) or <code>0x03</code> (<code>ReplaceFile</code>).	
Relates to	Dcm_WriteFile	
Variation	({ <code>ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestFileTransfer/DcmRequestFileTransferUsePort)</code> == <code>TRUE</code> })	
Parameters	OpStatus	
	Type	Dcm_OpStatusType
	Direction	IN
	Comment	DCM_INITIAL: All In-parameters are valid. DCM_PENDING: All In-parameters are set to <code>0x00</code> . DCM_CANCEL: All In-parameters are set to <code>0x00</code> . DCM_FORCE_RCRRP_OK: All In-parameters are set to <code>0x00</code> .
	Variation	–
	DataLength	
	Type	<code>uint64</code>
	Direction	IN
	Comment	Defines the length in bytes for the parameter <code>Data</code> . The value will not exceed, but might be less, compared to the value of <code>maxNumberOfBlockLength</code> return in <code>Dcm_ProcessRequestFileTransfer</code> .
	Variation	–
	Data	
	Type	Dcm_RequestDataArrayType
	Direction	IN
	Comment	Pointer to the data to be written.
	Variation	–





	ErrorCode	
	Type	Dcm_NegativeResponseCodeType
	Direction	OUT
	Comment	If the operation Dcm_WriteFile returns value E_NOT_OK, the DCM module shall send a negative response with NRC code equal to the parameter Error Code parameter value.
	Variation	—
Possible Errors	—	

]

8.8.3.9 DCMServices

[SWS_Dcm_00698] Definition of ClientServerInterface DCMServices [

Name	DCMServices		
Comment	–		
IsService	true		
Variation	–		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed

Operation	GetSecurityLevel	
Comment	–	
Relates to	Dcm_GetSecurityLevel	
Variation	–	
Parameters	SecLevel	
	Type	Dcm_SecLevelType
	Direction	OUT
	Comment	–
	Variation	–
Possible Errors	E_OK E_NOT_OK	

Operation	GetSesCtrlType		
Comment	–		
Relates to	Dcm_GetSesCtrlType		
Variation	–		
Parameters	SesCtrlType		
	Type	Dcm_SesCtrlType	
	Direction	OUT	
	Comment	–	
	Variation	–	
Possible Errors	E_OK E_NOT_OK		

Operation	ResetToDefaultSession
Comment	–
Relates to	Dcm_ResetToDefaultSession
Variation	–
Possible Errors	E_OK E_NOT_OK

Operation	SetActiveDiagnostic	
Comment	Allows to activate and deactivate the call of ComM_DCM_ActiveDiagnostic() function.	
Relates to	Dcm_SetActiveDiagnostic	
Variation	–	
Parameters	active	
	Type	boolean
	Direction	IN
	Comment	If false Dcm shall not call ComM_DCM_ActiveDiagnostic(). If true Dcm will call ComM_DCM_ActiveDiagnostic().
	Variation	–
Possible Errors	E_OK	

]

8.8.3.10 Authentication

[SWS_Dcm_91072] Definition of ClientServerInterface Authentication [

Name	Authentication		
Comment	–		
IsService	true		
Variation	–		
Possible Errors	0	E_OK	Operation successful

Operation	SetDeauthenticatedRole	
Comment	Sets a new role used in deauthenticated state for that connection. The set role is valid until the connection switches into authenticated state or the ECU is reset.	
Relates to	Dcm_SetDeauthenticatedRole	
Variation	–	
Parameters	deauthenticatedRole	
	Type	Dcm_AuthenticationRoleType
	Direction	IN
	Comment	New deauthenticated role that is assigned to that connection.
	Variation	–
Possible Errors	E_OK	

]

8.8.4 NvDataInterface

8.8.4.1 DataServices_{DID}

[SWS_Dcm_91061] Definition of SenderReceiverInterface DataServices_{DID}

Upstream requirements: [RS_Diag_04218](#)

[		
Name	DataServices_{DID}	
Comment	–	
IsService	false	
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == USE_ATOMIC_NV_DATA_INTERFACE) DID = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)}	
Data Elements	data	
	Type	{DID}_Struct_DataType
	Variation	DID = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)})
]		

8.8.5 Ports

This section formally specifies the corresponding AUTOSAR Service using the concepts of the Software-Component-Template. The following definition can be generated completely out of the configuration of the Dcm, which defines the exact ports that are present and their names.

Naming of the port : The prefix of the port name is fixed and defined hereafter (e.g. DataServices_). The name behind the prefix corresponds to the name of the associated container in the ECU configuration and can be freely defined during the configuration step. e.g. : for a [DcmDspData](#) container called Speed the port name would be DataServices_Speed

```

1 ServiceSwComponentType Dcm {
2
3     //the presence and name of this port is configuration-independent
4     ProvidePort DCMservices DCMservices;
5
6     //see configuration parameter DcmDspSecurityRow
7     RequirePort SecurityAccess_{SecurityLevel\} SecurityAccess_{
8         SecurityLevel\};
9     ...
10
11     //see configuration parameter DcmDspData
12     RequirePort DataServices_{Data\} DataServices_{Data\};
13     ProvidePort DataServices_{Data\} DataServices_{Data\}; // Only if
14         the data can be written and DcmDspDataUsePort is set to
15         USE_DATA_SENDER_RECEIVER or to USE_DATA_SENDER_RECEIVER_AS_SERVICE
16     ...

```

```

17 //see configuration parameter DcmDspVehInfoData
18 RequirePort InfotypeServices_{VehInfoData}
19     InfotypeServices_{VehInfoData}
20 ...
21
22
23 //see configuration parameter DcmDspRoutine
24 RequirePort RoutineServices_{RoutineName}
25     RoutineServices_{RoutineName};
26 ...
27
28 //see configuration parameter DcmDspRequestControl
29 RequirePort RequestControlServices_{Tid\}
30     RequestControlServices_{Tid\};
31 ...
32
33
34 //see configuration parameter
35     DcmDsdServiceRequestManufacturerNotication
36 RequirePort ServiceRequestNotification
37     ServiceRequestManufacturerNotication_{Name};
38 ...
39
40 //see configuration parameter DcmDsdServiceRequestSupplierNotication
41 RequirePort ServiceRequestNotification
42     ServiceRequestSupplierNotication_{_SWC>;
43 ...
44
45 //Note: When service 0x19 subfunctions 0x14 is used (call to //
46     Dem_GetNextFilteredDTCAndFDC), the following is defined:
47 //Non-DEM-internal calculated fault detection counters are typically
48 //requested from SW-Cs through the RTE. To indicate an equivalent call-
49     tree //for these runables, a work-around is used: The Dcm main
50     function //specifies a trigger to the DEM interface GeneralEvtInfo
51     (operation //GetFaultDetectionCounter), which triggers the
52     according ehavior (refer to //RunnableEntity
53     GetFaultDetectionCounter, chapter "Service Interface //
54     DiagnosticInfo & General" in DEM SWS).
55 RequirePort Dem/CallbackGetFaultDetectCounter CBFaultDetectCtrDummy
56 (The client-server interface can be used from the DEM.)
57
58 RunnableEntity MainFunction
59     symbol \ARApiRef{Dcm_MainFunction}"
60     _canbeInvokedConcurrently=_FALSE
61     _SSCP=_port_CBFaultDetectCtrDummy,_GetFaultDetectionCounter
62
63 Connector_from_CBFaultDetectCtrDummy_to_Dem/GeneralEvtInfo
64 }

```

8.8.5.1 DataServices_DIDRange_{Range}

[SWS_Dcm_01034] Definition of Port DataServices_DIDRange_{Range} required by module Dcm

Name	DataServices_DIDRange_{Range}		
Kind	RequiredPort	Interface	DataServices_DIDRange_{Range}
Description	–		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRange.DcmDspDidRangeUsePort)} == TRUE Range = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDidRange.SHORT-NAME)}		

8.8.5.2 DataServices_{DID}

[SWS_Dcm_91058] Definition of Port DataServices_{DID} provided by module Dcm

Upstream requirements: [RS_Diag_04218](#)

Name	DataServices_{DID}		
Kind	ProvidedPort	Interface	DataServices_{DID}, DataServices_{DID}
Description	–		
Variation	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == (USE_ATOMIC_NV_DATA_INTERFACE)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidWrite)} != NULL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidRead)} == NULL) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl)} != NULL)) DID = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)})		

[SWS_Dcm_91060] Definition of Port DataServices_{DID} provided/required by module Dcm

Upstream requirements: [RS_Diag_04218](#)

Name	DataServices_{DID}		
Kind	Provided RequiredPort	Interface	DataServices_{DID}, DataServices_{DID}
Description	–		





Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)) == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)) == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)) == (USE_ATOMIC_NV_DATA_INTERFACE)) && (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidWrite)) != NULL && (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidRead)) != NULL)) DID = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)}

┌

[SWS_Dcm_91059] Definition of Port DataServices_{DID} required by module Dcm

Upstream requirements: [RS_Diag_04218](#)

┌

Name	DataServices_{DID}		
Kind	RequiredPort	Interface	DataServices_{DID} , DataServices_{DID}
Description	–		
Variation	((ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)) == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)) == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE) (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)) == (USE_ATOMIC_NV_DATA_INTERFACE)) && (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidWrite)) == NULL && (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidRead)) != NULL (ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef->DcmDspDidInfo/DcmDspDidControl)) != NULL)) DID = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)}		

└

8.8.5.3 DataServices_{Data}

[SWS_Dcm_01035] Definition of Port DataServices_{Data} required by module Dcm

Name	DataServices_{Data}		
Kind	RequiredPort	Interface	DataServices_{Data} , DataServices_{Data}
Description	–		
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData/DcmDspPidService01/DcmDspPidDataUsePort)} == (USE_DATA_SYNCH_CLIENT_SERVER USE_DATA_SENDER_RECEIVER USE_DATA_SENDER_RECEIVER_AS_SERVICE)) Data = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)} {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspPid/DcmDspPidData.SHORT-NAME)}		

└

[SWS_Dcm_01310] Definition of Port DataServices_{Data} required by module Dcm

Name	DataServices_{Data}		
Kind	RequiredPort	Interface	DataServices_{Data}, DataServices_{Data}
Description	–		
Variation	((({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == USE_DATA_ELEMENT_SPECIFIC_INTERFACES)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataUsePort)} == (USE_DATA_SENDER_RECEIVER USE_DATA_SENDER_RECEIVER_AS_SERVICE))) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef-> DcmDspDidInfo/DcmDspDidWrite)} == NULL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef-> DcmDspDidInfo/DcmDspDidRead)} != NULL)) Data = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}		

[SWS_Dcm_01031] Definition of Port DataServices_{Data} provided by module Dcm

Name	DataServices_{Data}		
Kind	ProvidedPort	Interface	DataServices_{Data}, DataServices_{Data}
Description	–		
Variation	((({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == USE_DATA_ELEMENT_SPECIFIC_INTERFACES)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataUsePort)} == (USE_DATA_SENDER_RECEIVER USE_DATA_SENDER_RECEIVER_AS_SERVICE))) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef-> DcmDspDidInfo/DcmDspDidWrite)} != NULL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef-> DcmDspDidInfo/DcmDspDidRead)} == NULL)) Data = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}		

[SWS_Dcm_01311] Definition of Port DataServices_{Data} provided/required by module Dcm

Name	DataServices_{Data}		
Kind	Provided RequiredPort	Interface	DataServices_{Data}
Description	–		
Variation	((({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == USE_DATA_ELEMENT_SPECIFIC_INTERFACES)) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData/DcmDspDataUsePort)} == (USE_DATA_SENDER_RECEIVER USE_DATA_SENDER_RECEIVER_AS_SERVICE))) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef-> DcmDspDidInfo/DcmDspDidWrite)} != NULL) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef-> DcmDspDidInfo/DcmDspDidRead)} != NULL)) Data = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspData.SHORT-NAME)}		

8.8.5.4 IOControlRequest_{DID}

[SWS_Dcm_01312] Definition of Port IOControlRequest_{DID} provided/required by module Dcm

Name	IOControlRequest_{DID}		
Kind	Provided RequiredPort	Interface	IOControlRequest_{DID}
Description	–		
Variation	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE))) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef-> DcmDspDidInfo/DcmDspDidControl)} != NULL) DID = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)})		

]

8.8.5.5 IOControlResponse_{DID}

[SWS_Dcm_01313] Definition of Port IOControlResponse_{DID} required by module Dcm

Upstream requirements: [RS_Diag_04218](#)

Name	IOControlResponse_{DID}		
Kind	RequiredPort	Interface	IOControlResponse_{DID}
Description	–		
Variation	(({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidUsePort)} == (USE_ATOMIC_SENDER_RECEIVER_INTERFACE USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE))) && ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid/DcmDspDidInfoRef-> DcmDspDidInfo/DcmDspDidControl)} != NULL) DID = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspDid.SHORT-NAME)})		

]

8.8.5.6 DCMServices

[SWS_Dcm_01030] Definition of Port DCMServices provided by module Dcm

Name	DCMServices		
Kind	ProvidedPort	Interface	DCMServices
Description	–		
Variation	–		

]

8.8.5.7 InfotypeServices_{VehInfoData}

[SWS_Dcm_01037] Definition of Port InfotypeServices_{VehInfoData} required by module Dcm

Name	InfotypeServices_{VehInfoData}		
Kind	RequiredPort	Interface	InfotypeServices_{VehInfoData}
Description	–		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspVehInfo/DcmDspVehInfoData/DcmDspVehInfoDataUsePort)}==TRUE VehInfoData = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspVehInfo/DcmDspVehInfoData.SHORT-NAME)}		

]

8.8.5.8 RequestControlServices_{Tid}

[SWS_Dcm_01038] Definition of Port RequestControlServices_{Tid} required by module Dcm

Name	RequestControlServices_{Tid}		
Kind	RequiredPort	Interface	RequestControlServices_{Tid}
Description	–		
Variation	Tid = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestControl.SHORT-NAME)}		

]

8.8.5.9 ServiceRequestManufacturerNotification_{Name}

[SWS_Dcm_01039] Definition of Port ServiceRequestManufacturerNotification_{Name} required by module Dcm

Name	ServiceRequestManufacturerNotification_{Name}		
Kind	RequiredPort	Interface	ServiceRequestNotification
Description	–		
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsd/DcmDsdServiceRequestManufacturerNotification)} != NULL) Name = {ecuc(Dcm/DcmConfigSet/DcmDsd/DcmDsdServiceRequestManufacturerNotification.SHORT-NAME)}		

]

8.8.5.10 ServiceRequestSupplierNotification_{Name}

[SWS_Dcm_01042] Definition of Port ServiceRequestSupplierNotification_{Name} required by module Dcm [

Name	ServiceRequestSupplierNotification_{Name}		
Kind	RequiredPort	Interface	ServiceRequestNotification
Description	–		
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsd/DcmDsdServiceRequestSupplierNotification)} != NULL) Name = {ecuc(Dcm/DcmConfigSet/DcmDsd/DcmDsdServiceRequestSupplierNotification.SHORT-NAME)}		

]

8.8.5.11 RoutineServices_{RoutineName}

[SWS_Dcm_01040] Definition of Port RoutineServices_{RoutineName} required by module Dcm [

Name	RoutineServices_{RoutineName}		
Kind	RequiredPort	Interface	RoutineServices_{RoutineName}
Description	–		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.DcmDspRoutineUsePort)} == TRUE RoutineName = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRoutine.SHORT-NAME)}		

]

8.8.5.12 SecurityAccess_{SecurityLevel}

[SWS_Dcm_01041] Definition of Port SecurityAccess_{SecurityLevel} required by module Dcm [

Name	SecurityAccess_{SecurityLevel}		
Kind	RequiredPort	Interface	SecurityAccess_{SecurityLevel}
Description	–		
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.DcmDspSecurityUsePort)} == USE_ASYNC_CLIENT_SERVER SecurityLevel = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.SHORT-NAME)}		

]

8.8.5.13 Dcm_DiagnosticSessionControlModeSwitchInterface

[SWS_Dcm_91033] Definition of Port DiagnosticSessionControlModeSwitchInterface provided by module Dcm [

Name	DiagnosticSessionControlModeSwitchInterface		
Kind	ProvidedPort	Interface	Dcm_DiagnosticSessionControlModeSwitchInterface
Description	A ModeSwitchInterface PPortPrototype used to notify SW-Cs about the current Diagnostic Session		
Variation	–		

]

8.8.5.14 Dcm_EcuResetModeSwitchInterface

[SWS_Dcm_91034] Definition of Port EcuResetModeSwitchInterface provided by module Dcm [

Name	EcuResetModeSwitchInterface		
Kind	ProvidedPort	Interface	Dcm_EcuResetModeSwitchInterface
Description	A ModeSwitchInterface PPortPrototype used to notify SW-Cs about an upcoming ECU Reset and its type		
Variation	–		

]

8.8.5.15 Dcm_ModeRapidPowerShutDownModeSwitchInterface

[SWS_Dcm_91035] Definition of Port ModeRapidPowerShutDownModeSwitchInterface provided by module Dcm [

Name	ModeRapidPowerShutDownModeSwitchInterface		
Kind	ProvidedPort	Interface	Dcm_ModeRapidPowerShutDownModeSwitchInterface
Description	A ModeSwitchInterface PPortPrototype used to notify SW-Cs about the rapid power shut down mode		
Variation	–		

]

8.8.5.16 Dcm_CommunicationControlModeSwitchInterface_{ComMChannel-Name}

[SWS_Dcm_91036] Definition of Port CommunicationControlModeSwitchInterface_{ComMChannelName} provided by module Dcm [

Name	CommunicationControlModeSwitchInterface_{ComMChannelName}		
Kind	ProvidedPort	Interface	CommunicationControlModeSwitchInterface_{ComMChannelName}
Description	A ModeSwitchInterface PPortPrototype used to notify SW-Cs about the communication control of the indicated ComM channel		
Variation	ComMChannelName = {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspComControl/DcmDspComControlAllChannel/DcmDspAllComMChannelRef->ComMChannel.SHORT-NAME)} or {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspComControl/DcmDspComControlSpecificChannel/DcmDspSpecificComMChannelRef->ComMChannel.SHORT-NAME)}		

]

8.8.5.17 Dcm_ControlDTCSettingModeSwitchInterface

[SWS_Dcm_91037] Definition of Port ControlDTCSettingModeSwitchInterface provided by module Dcm [

Name	ControlDTCSettingModeSwitchInterface		
Kind	ProvidedPort	Interface	Dcm_ControlDTCSettingModeSwitchInterface
Description	A ModeSwitchInterface PPortPrototype used to notify SW-Cs about the DTC Setting mode		
Variation	–		

]

8.8.5.18 Dcm_SecurityAccessModeSwitchInterface

[SWS_Dcm_91039] Definition of Port SecurityAccessModeSwitchInterface provided by module Dcm [

Name	SecurityAccessModeSwitchInterface		
Kind	ProvidedPort	Interface	Dcm_SecurityAccessModeSwitchInterface
Description	A ModeSwitchInterface PPortPrototype used to notify SW-Cs about the current Security Level		
Variation	–		

]

8.8.5.19 Dcm_UploadDownloadServices

[SWS_Dcm_91084] Definition of Port UploadDownloadServices required by module Dcm

Upstream requirements: [RS_Diag_04135](#)

[

Name	UploadDownloadServices		
Kind	RequiredPort	Interface	UploadDownloadServices
Description	–		
Variation	({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspMemoryTransfer.DcmDspMemoryTransferUsePort)} == TRUE) ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspRequestFileTransfer/DcmRequestFileTransferUsePort)} == TRUE)		

]

8.8.5.20 Dcm_Authentication_{Connection}

[SWS_Dcm_91073] Definition of Port Authentication_{Connection} provided by module Dcm

Name	Authentication_{Connection}		
Kind	ProvidedPort	Interface	Authentication
Description	–		
Port Defined Argument Value(s)	Type	uint16	
	Value	{ecuc(Dcm/DcmConfigSet/DcmDsl/DcmDslProtocol/DcmDslProtocolRow/DcmDslConnection/DcmDslMainConnection.DcmDslProtocolRxConnectionId)}	
Variation	Connection = { Dcm/DcmConfigSet/DcmDsl/DcmDslProtocol/DcmDslProtocolRow/DcmDslConnection/DcmDslMainConnection.Short-Name})}		

]

8.8.5.21 Dcm_AuthenticationStateMSI

[SWS_Dcm_91075] Definition of Port AuthenticationStateModeSwitchInterface_{Connection} provided by module Dcm

Name	AuthenticationStateModeSwitchInterface_{Connection}		
Kind	ProvidedPort	Interface	Dcm_AuthenticationStateModeSwitchInterface_{Connection}
Description	–		
Variation	Connection = Dcm/DcmConfigSet/DcmDsl/DcmDslProtocol/DcmDslProtocolRow/DcmDslConnection/DcmDslMainConnection.Short-Name		

]

8.8.6 ModeDeclarationGroups

8.8.6.1 DcmDiagnosticSessionControl

[SWS_Dcm_91019] Definition of ModeDeclarationGroup DcmDiagnosticSessionControl [

Name	DcmDiagnosticSessionControl	
Kind	ModeDeclarationGroup	
Category	EXPLICIT_ORDER	
Initial mode	DCM_DEFAULT_SESSION	
On transition value	255	
Modes	DCM_DEFAULT_SESSION	0
	DCM_PROGRAMMING_SESSION	1
	DCM_EXTENDED_DIAGNOSTIC_SESSION	2
	DCM_SAFETY_SYSTEM_DIAGNOSTIC_SESSION	3
Description	ModeDeclarationGroup representing the different diagnostic sessions Further modes to be added: {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSession/ DcmDspSessionRow.SHORT-NAME)}	

]

Note: According [SWS_Dcm_CONSTR_06001] there are standardized mode declaration which are part of the standardized AUTOSAR interface. Note: Refer [ecuc_sws_2108] defining the symbolic name prefix

8.8.6.2 DcmEcuReset

[SWS_Dcm_91021] Definition of ModeDeclarationGroup DcmEcuReset [

Name	DcmEcuReset	
Kind	ModeDeclarationGroup	
Category	EXPLICIT_ORDER	
Initial mode	DCM_NONE	
On transition value	255	
Modes	DCM_NONE	0
	DCM_HARD	1
	DCM_KEYONOFF	2
	DCM_SOFT	3
	DCM_JUMPTOBOOTLOADER	4
	DCM_JUMPTOSYSSUPPLIERBOOTLOADER	5
	DCM_EXECUTE	6
Description	ModeDeclarationGroup representing the different ECU reset types	

]

8.8.6.3 DcmModeRapidPowerShutDown

[SWS_Dcm_91023] Definition of ModeDeclarationGroup DcmModeRapidPower ShutDown

Name	DcmModeRapidPowerShutDown	
Kind	ModeDeclarationGroup	
Category	EXPLICIT_ORDER	
Initial mode	DCM_ENABLE_RAPIDPOWERSHUTDOWN	
On transition value	255	
Modes	DCM_ENABLE_RAPIDPOWERSHUTDOWN	0
	DCM_DISABLE_RAPIDPOWERSHUTDOWN	1
Description	ModeDeclarationGroup representing the enable/disable state of rapid power shutdown	

8.8.6.4 DcmCommunicationControl

[SWS_Dcm_91025] Definition of ModeDeclarationGroup DcmCommunication Control

Name	DcmCommunicationControl	
Kind	ModeDeclarationGroup	
Category	EXPLICIT_ORDER	
Initial mode	DCM_ENABLE_RX_TX_NORM_NM	
On transition value	255	
Modes	DCM_ENABLE_RX_TX_NORM	0
	DCM_ENABLE_RX_DISABLE_TX_NORM	1
	DCM_DISABLE_RX_ENABLE_TX_NORM	2
	DCM_DISABLE_RX_TX_NORMAL	3
	DCM_ENABLE_RX_TX_NM	4
	DCM_ENABLE_RX_DISABLE_TX_NM	5
	DCM_DISABLE_RX_ENABLE_TX_NM	6
	DCM_DISABLE_RX_TX_NM	7
	DCM_ENABLE_RX_TX_NORM_NM	8
	DCM_ENABLE_RX_DISABLE_TX_NORM_NM	9
	DCM_DISABLE_RX_ENABLE_TX_NORM_NM	10
	DCM_DISABLE_RX_TX_NORM_NM	11
Description	ModeDeclarationGroup representing the different communication control states	

8.8.6.5 DcmControlDTCSetting

[SWS_Dcm_91027] Definition of ModeDeclarationGroup DcmControlDTCSetting

Name	DcmControlDTCSetting	
Kind	ModeDeclarationGroup	
Category	EXPLICIT_ORDER	
Initial mode	DCM_ENABLEDTCSSETTING	
On transition value	255	
Modes	DCM_ENABLEDTCSSETTING	0
	DCM_DISABLEDTCSSETTING	1
Description	ModeDeclarationGroup representing the enable/disable state for DTC storage	

8.8.6.6 DcmSecurityAccess

[SWS_Dcm_91031] Definition of ModeDeclarationGroup DcmSecurityAccess

Name	DcmSecurityAccess	
Kind	ModeDeclarationGroup	
Category	EXPLICIT_ORDER	
Initial mode	DCM_SEC_LEV_LOCKED	
On transition value	255	
Modes	DCM_SEC_LEV_LOCKED	0
Description	ModeDeclarationGroup representing the different diagnostic security levels Further modes to be added: {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.SHORT-NAME)}	

8.8.6.7 DcmAuthenticationState

[SWS_Dcm_91067] Definition of ModeDeclarationGroup DcmAuthenticationState_{Connection}

Name	DcmAuthenticationState_{Connection}	
Kind	ModeDeclarationGroup	
Category	EXPLICIT_ORDER	
Initial mode	DCM_DEAUTHENTICATED	
On transition value	255	
Modes	DCM_DEAUTHENTICATED	0
	DCM_AUTHENTICATED	1
Description	Representing the authentication state of a diagnostic connection.	

8.8.7 Mode-Switch-Interfaces

8.8.7.1 Dcm_DiagnosticSessionControlModeSwitchInterface

[SWS_Dcm_91020] Definition of ModeSwitchInterface Dcm_DiagnosticSessionControlModeSwitchInterface [

Name	Dcm_DiagnosticSessionControlModeSwitchInterface	
Comment	A SW-C that wants to get informed about the current Diagnostic Session requires the Mode SwitchInterface Dcm_DiagnosticSessionControlModeSwitchInterface	
IsService	true	
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSession/DcmDspSessionRow.SHORT-NAME)}	
ModeGroup	diagnosticSession	DcmDiagnosticSessionControl

]

8.8.7.2 Dcm_EcuResetModeSwitchInterface

[SWS_Dcm_91022] Definition of ModeSwitchInterface Dcm_EcuResetModeSwitchInterface [

Name	Dcm_EcuResetModeSwitchInterface	
Comment	A SW-C that wants to get informed about an upcoming ECU Reset requires the ModeSwitch Interface Dcm_EcuResetModeSwitchInterface	
IsService	true	
Variation	–	
ModeGroup	ecuReset	DcmEcuReset

]

8.8.7.3 Dcm_ModeRapidPowerShutDownModeSwitchInterface

[SWS_Dcm_91024] Definition of ModeSwitchInterface Dcm_ModeRapidPowerShutDownModeSwitchInterface [

Name	Dcm_ModeRapidPowerShutDownModeSwitchInterface	
Comment	A SW-C that wants to get informed about the rapid power shut down mode requires the Mode SwitchInterface Dcm_ModeRapidPowerShutDownModeSwitchInterface	
IsService	true	
Variation	–	
ModeGroup	modeRapidPowerShutDown	DcmModeRapidPowerShutDown

]

8.8.7.4 Dcm_CommunicationControlModeSwitchInterface

[SWS_Dcm_91026] Definition of ModeSwitchInterface CommunicationControlModeSwitchInterface_{ComMChannelName} [

Name	CommunicationControlModeSwitchInterface_{ComMChannelName}	
Comment	A SW-C that wants to get informed about the communication control of a ComM channel requires the ModeSwitchInterface Dcm_CommunicationControlModeSwitchInterface	
IsService	true	
Variation	ComMChannelName = ({ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspComControl/DcmDspComControlAllChannel/DcmDspAllComMChannelRef->ComMChannel.SHORT-NAME)} {ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspComControl/DcmDspComControlSpecificChannel/DcmDspSpecificComMChannelRef->	
ModeGroup	communicationControl	DcmCommunicationControl

]

8.8.7.5 Dcm_ControlDTCSettingModeSwitchInterface

[SWS_Dcm_91028] Definition of ModeSwitchInterface Dcm_ControlDTCSettingModeSwitchInterface [

Name	Dcm_ControlDTCSettingModeSwitchInterface	
Comment	A SW-C that wants to get informed about the DTC Setting mode requires the ModeSwitchInterface Dcm_ControlDTCSettingModeSwitchInterface	
IsService	true	
Variation	–	
ModeGroup	controlDTCSetting	DcmControlDTCSetting

]

8.8.7.6 Dcm_SecurityAccessModeSwitchInterface

[SWS_Dcm_91032] Definition of ModeSwitchInterface Dcm_SecurityAccessModeSwitchInterface [

Name	Dcm_SecurityAccessModeSwitchInterface	
Comment	A SW-C that wants to get informed about the current Security Level requires the ModeSwitchInterface Dcm_SecurityAccessModeSwitchInterface	
IsService	true	
Variation	{ecuc(Dcm/DcmConfigSet/DcmDsp/DcmDspSecurity/DcmDspSecurityRow.SHORT-NAME)}	
ModeGroup	securityAccess	DcmSecurityAccess

]

8.8.7.7 Dcm_AuthenticationStateModeSwitchInterface

[SWS_Dcm_91074] Definition of ModeSwitchInterface Dcm_AuthenticationStateModeSwitchInterface_{Connection} [

Name	Dcm_AuthenticationStateModeSwitchInterface_{Connection}	
Comment	–	
IsService	true	
Variation	Connection = Dcm/DcmConfigSet/DcmDsl/DcmDslProtocol/DcmDslProtocolRow/DcmDslConnection/DcmDslMainConnection.Short-Name	
ModeGroup	authenticationState	DcmAuthenticationState

]

8.9 External diagnostic service processing

The following chapter applies only to external processed diagnostic services.

8.9.1 <Module>_<DiagnosticService>

[SWS_Dcm_00763] Definition of configurable interface <Module>_<DiagnosticService> [

Service Name	<Module>_<DiagnosticService>	
Syntax	<pre>Std_ReturnType <Module>_<DiagnosticService> (Dcm_ExtendedOpStatusType OpStatus, Dcm_MsgContextType* pMsgContext, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x7e	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL DCM_PENDING DCM_CANCEL DCM_FORCE_RCRRP_OK DCM_POS_RESPONSE_SENT DCM_POS_RESPONSE_FAILED DCM_NEG_RESPONSE_SENT DCM_NEG_RESPONSE_FAILED
Parameters (inout)	pMsgContext	Message-related information for one diagnostic protocol identifier. The pointers in pMsgContext shall point behind the SID.
Parameters (out)	ErrorCode	If the operation <Module>_<DiagnosticService> returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful E_NOT_OK: Request was not successful DCM_E_PENDING: Request is not yet finished DCM_E_FORCE_RCRRP: Application requests the transmission of a response Response Pending (NRC 0x78)





Description	Callout function. The Dcm shall call this callout function as soon as valid message is received on relevant DcmDslProtocolRxPduld on SID level. The usecase of multiple diagnostic protocols will be possible by using different arguments and the function shall be programmed in a way that it is reentrant. Caller is responsible for the lifetime of the argument pMsgContext. The name of the callout is defined within parameter DcmDsdSidTabFnc
Available via	Dcm_Externals.h

]

8.9.2 <Module>_<DiagnosticService>_<SubService>

[SWS_Dcm_00764] Definition of configurable interface <Module>_<DiagnosticService>_<SubService> [

Service Name	<Module>_<DiagnosticService>_<SubService>	
Syntax	<pre>Std_ReturnType <Module>_<DiagnosticService>_<SubService> (Dcm_ExtendedOpStatusType OpStatus, Dcm_MsgContextType* pMsgContext, Dcm_NegativeResponseCodeType* ErrorCode)</pre>	
Service ID [hex]	0x33	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	OpStatus	DCM_INITIAL DCM_PENDING DCM_CANCEL DCM_FORCE_RCRRP_OK DCM_POS_RESPONSE_SENT DCM_POS_RESPONSE_FAILED DCM_NEG_RESPONSE_SENT DCM_NEG_RESPONSE_FAILED
Parameters (inout)	pMsgContext	Message-related information for one diagnostic protocol identifier. The pointers in pMsgContext shall point behind the SID.
Parameters (out)	ErrorCode	If the operation <Module>_<DiagnosticService>_<SubService> returns value E_NOT_OK, the Dcm module shall send a negative response with NRC code equal to the parameter ErrorCode parameter value.
Return value	Std_ReturnType	E_OK: Request was successful E_NOT_OK: Request was not successful DCM_E_PENDING: Request is not yet finished DCM_E_FORCE_RCRRP: Application requests the transmission of a response Response Pending (NRC 0x78)
Description	Callout function. If a DcmDsdSubServiceFnc is configured for the received subservice, the Dcm shall call this callout function as soon as this subservice is requested. The usecase of multiple diagnostic protocols will be possible by using different arguments and the function shall be programmed in a way that it is reentrant. Caller is responsible for the lifetime of the argument pMsgContext. The name of the callout is defined within parameter DcmDsdSubServiceFnc.	
Available via	Dcm_Externals.h	

]

8.10 Internal interfaces (not normative)

The following interfaces are used in the Dcm SWS in order to improve the understanding of the Dcm module behavior. An implementation is not required to use these interfaces.

8.10.1 DslInternal_SetSecurityLevel

```
1 void
2 DslInternal_SetSecurityLevel(Dcm_SecLevelType SecurityLevel)
```

This function sets a new security level value in the [Dcm](#) module. NOTE: for the definition of the parameter, refer to [Dcm_GetSecurityLevel](#).

8.10.2 DslInternal_SetSesCtrlType

```
1 void
2 DslInternal_SetSesCtrlType(Dcm_SesCtrlType SesCtrlType)
```

This function sets a new session control type value in the [Dcm](#) module. NOTE: for the definition of the parameter, refer to the [Dcm_GetSesCtrlType](#).

8.10.3 DspInternal_DcmConfirmation

```
1 void
2 DspInternal_DcmConfirmation(Dcm_IdContextType idContext,
3 uint16 ConnectionId
4 Dcm_ConfirmationStatusType status)
```

This function confirms the successful transmission or a transmission error of a diagnostic service. This is the right time to perform any application state transitions.

This [API](#) is also called if the response to a diagnostic service is suppressed.

8.10.4 DsdInternal_StartPagedProcessing

```
1 void
2 DsdInternal_StartPagedProcessing(const Dcm_MsgContextType* pMsgContext)
```

With this API, the [DSP](#) submodule gives the complete response length to the [Dcm](#) module and starts paged-buffer handling. This [API](#) starts no transmission!

8.10.5 DspInternal_CancelPagedBufferProcessing

```
1 void
2 DspInternal_CancelPagedBufferProcessing()
```

[Dcm](#) informs [DSP](#), that processing of paged-buffer was cancelled due to errors. Upon this call, [DSP](#) is not allowed to process further on paged-buffer handling.

8.10.6 DsdInternal_ProcessPage

```
1 void
2 DsdInternal_ProcessPage(Dcm_MsgLenType FilledPageLen)
```

[DSP](#) requests transmission of filled page.

9 Sequence diagrams

9.1 Overview

For clarification, the following sequence diagrams don't represent the full communication mechanism between the `Dcm` module and the PduR module. This is to keep the sequence diagrams simple. Before the `Dcm_TpRxIndication` call, the PduR module will ask the `Dcm` module for a buffer by calling `Dcm_StartOfReception` and `Dcm_CopyRxData`. This exchange is not shown on the next sequence diagrams. After a `PduR_DcmTransmit()` request from the `Dcm` module to the PduR module, data exchanges with `Dcm_CopyTxData` service, are not shown in the sequence diagrams. The function `Xxx_StartProtocol()` shall be called with the very first diagnostic request.

9.2 DSL (Diagnostic Session Layer)

9.2.1 Process Busy behavior

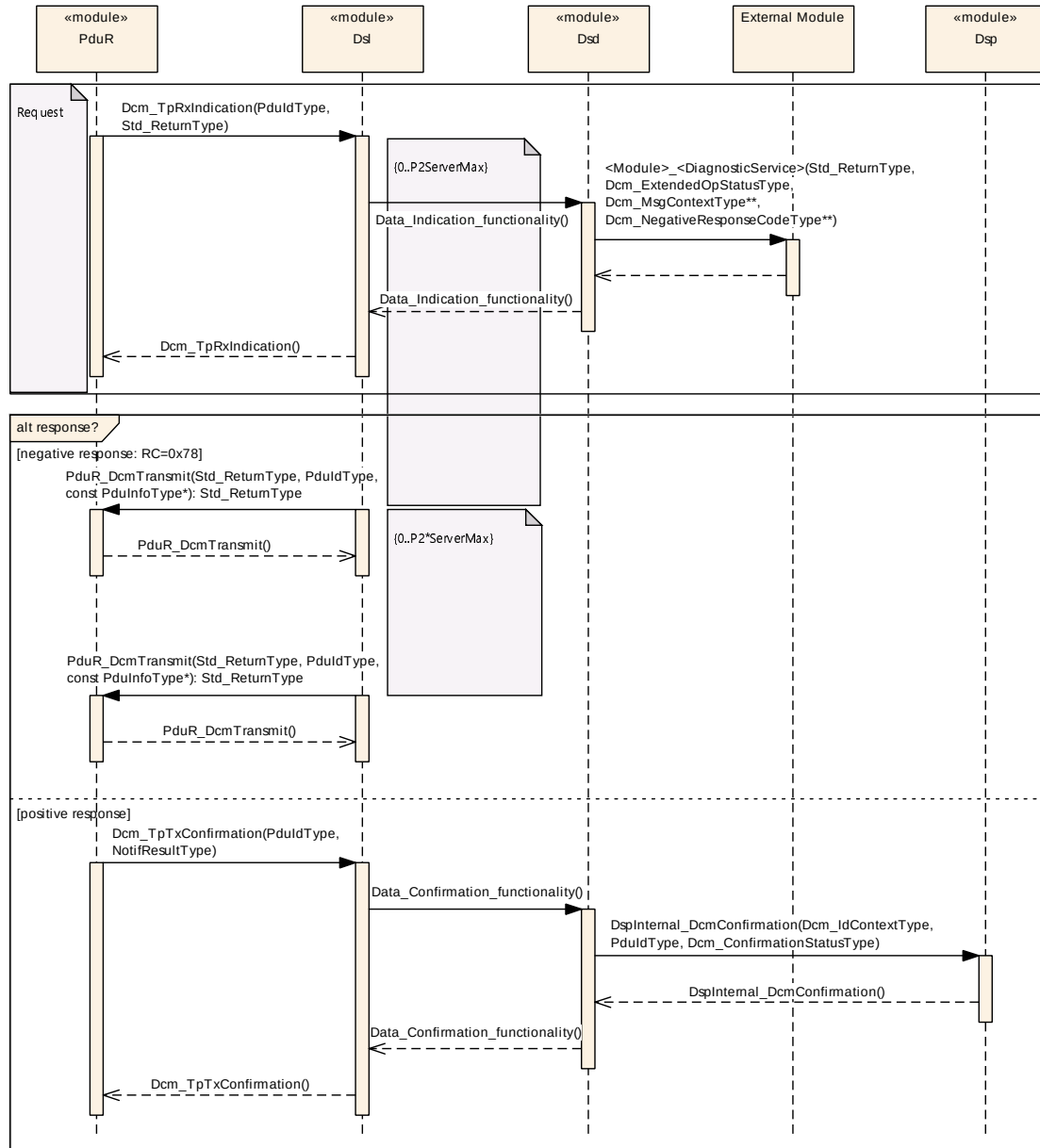


Figure 9.1: Process Busy Behavior

Internally, the **DSL** submodule calculates the time to response the tester. In the case that the external module processing the request doesn't close the request by returning `E_OK` or `E_NOT_OK` to `<Module>_<DiagnosticService>()` or `<Module>_<DiagnosticService>_<SubService>()` APIs call (in case of normal response handling) or `DsdInternal_ProcessPage()` (in case of paged-buffer handling) during the `P2ServerMax` and/or `P2*ServerMax`, the **DSL** submodule sends a negative response (requestCorrectlyReceived-ResponsePending) independently.

9.2.2 Update Diagnostic Session Control when timeout occurs

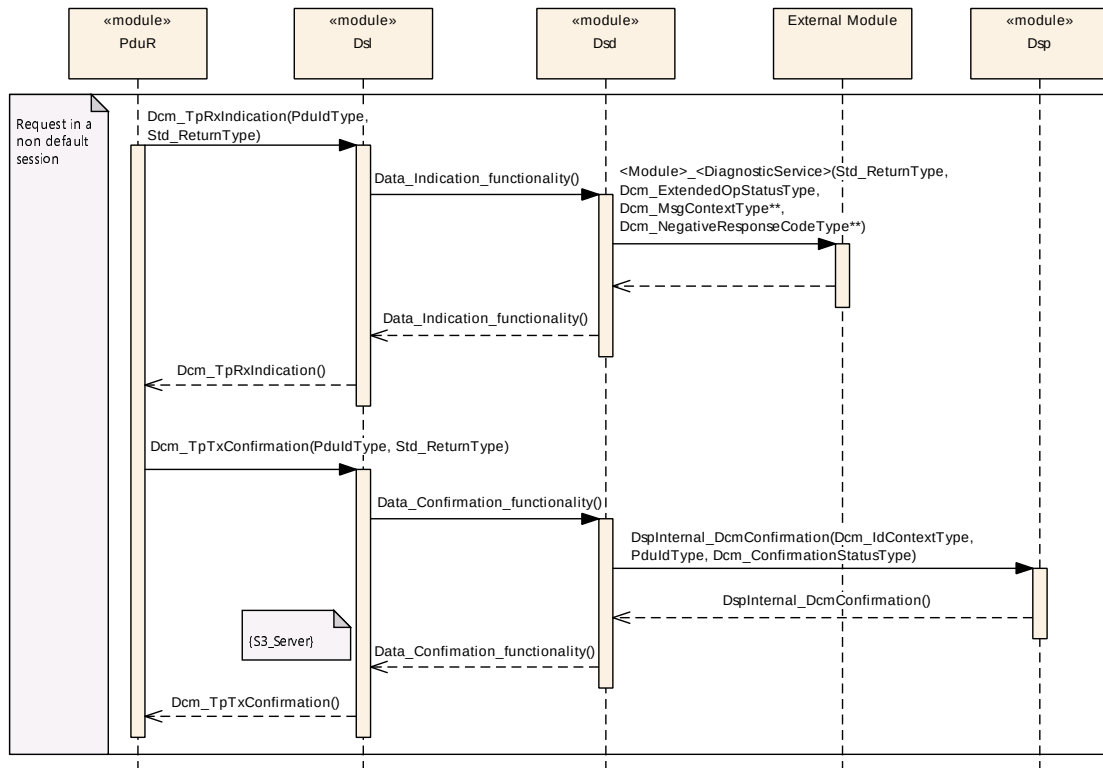


Figure 9.2: Update Diagnostic Session Control when timeout occurs

The **DSL** submodule resets session control value to default, if in a non-default session S3server timeout occurs. S3server timeout timer will be started with every data confirmation from the PduR module.

9.2.3 Process single response of ReadDataByPeriodicIdentifier

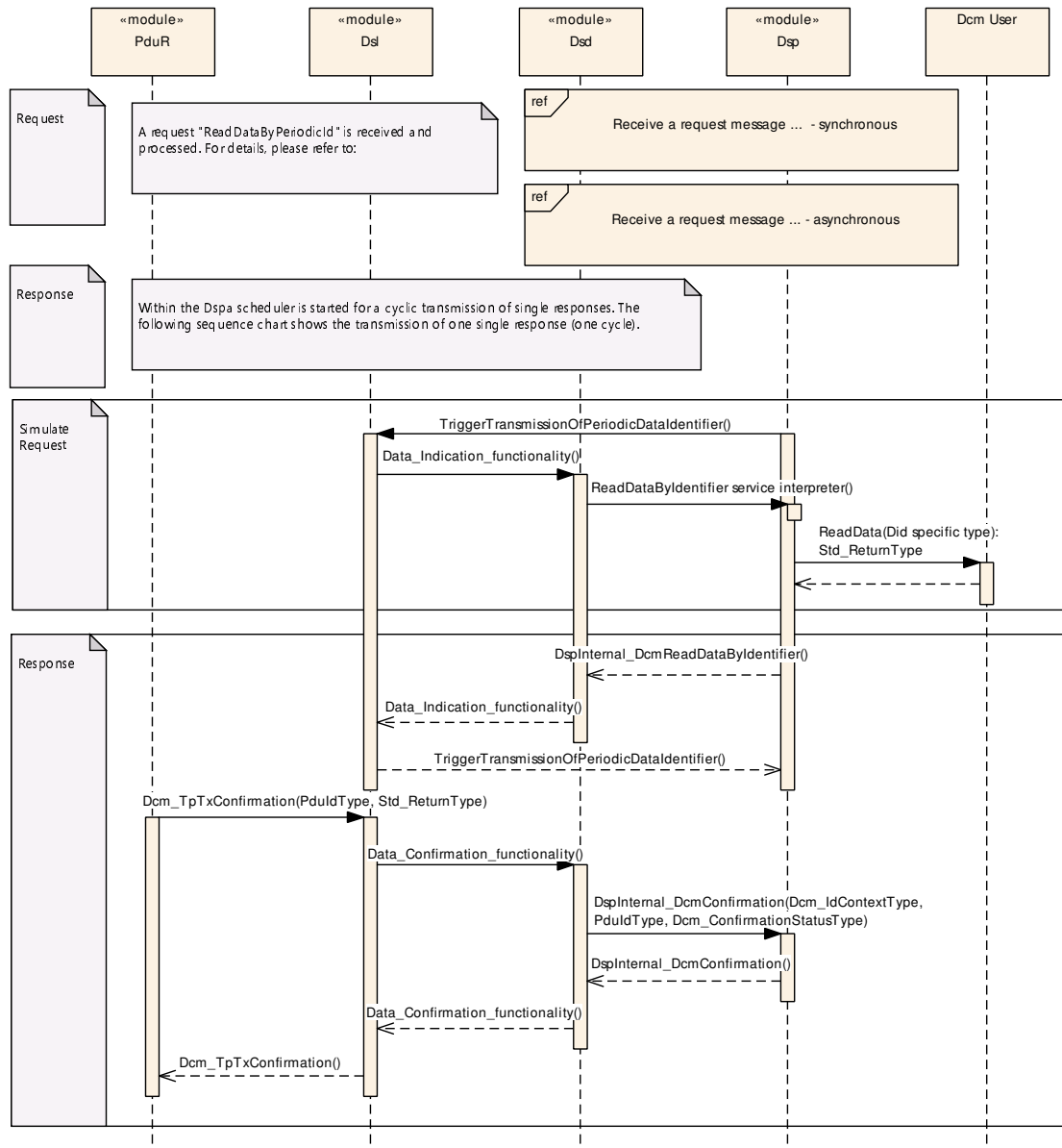


Figure 9.3: Process single response of ReadDataByPeriodicIdentifier

The **DSP** submodule requests sampling and transmission of Periodic Identifier data, when an event to Periodic Identifier occurs (i. e. a given time period is over). The **DSP** submodule initiates the sending of one periodic identifier calling the function `ResponseOnOneDataByPeriodicId()` provided by the **DSL** submodule.

Within this function the **DSL** submodule simulates a "ReadDataByIdentifier" request for the given PeriodicId. The High byte of the DataIdentifier shall be set to 0xF2 as specified in [19]) and the low byte is set to value of the PeriodicId.

The ReadData interfaces of the corresponding Datas of the **DID** are called to get the **DID** value. The **Dcm** module is not able to receive for the same periodic identifier

another event request from the **DSP** submodule, unless the confirmation of the current transmission is received.

9.2.4 Process single event-triggered response of ResponseOnEvent

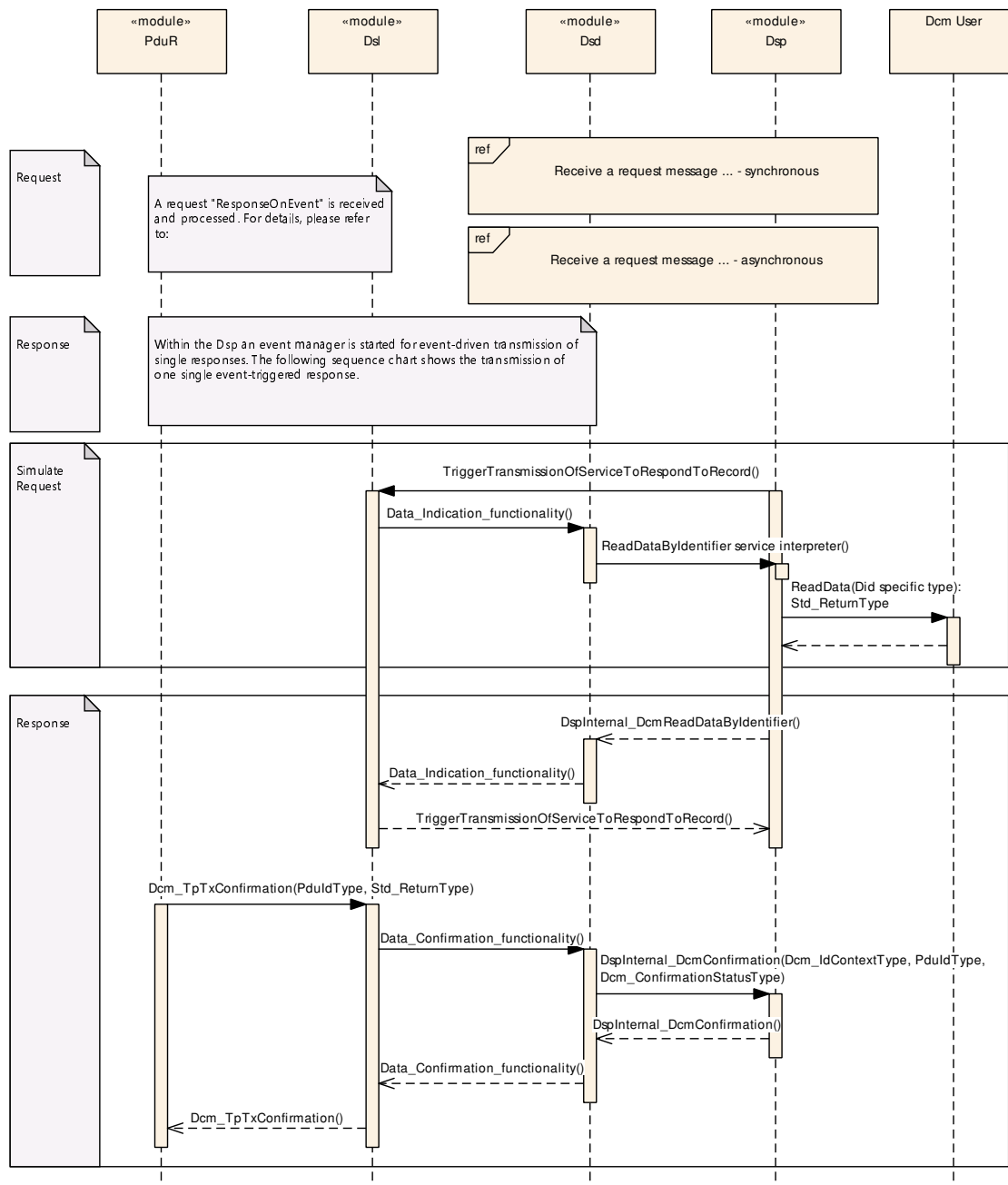


Figure 9.4: Process single event-triggered response of ResponseOnEvent

This sequence diagram shows an example for ResponseOnEvent. ResponseOnEvent is setup and started for onDTCStatusChange. Event changes are reported to the **Dcm** which will trigger a serviceToRespondTo.

9.2.5 Process concurrent requests

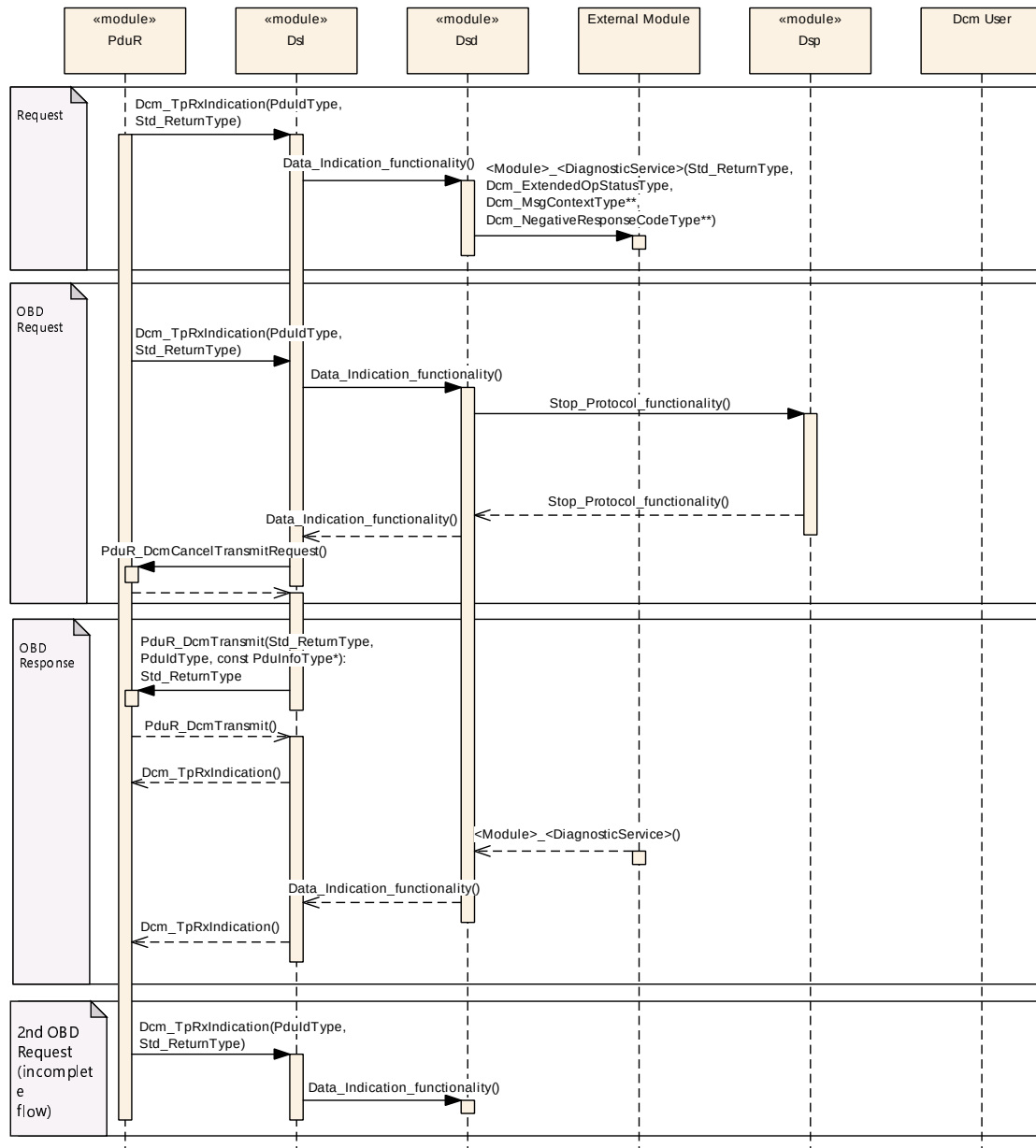


Figure 9.5: Process concurrent requests

On reception of **OBD** request in parallel to processing of a normal diagnostic request (e.g. enhanced diagnostic protocol, customer diagnostic protocol), running diagnostic request will be preempted. This is due to the configured higher priority of **OBD** protocol (see configuration parameter `DcmDslProtocolPriority`).

The following is processed on reception of 1st **OBD** request:

- The Application is informed of the protocol stop (done with `Xxx_StopProtocol()`) and resets to a stable state (e.g. switch of digital I/Os,...).

- Lower Layer is requested to cancel ongoing transmission on the same N-PDU (done with `PduR_DcmCancelTransmitRequest()`).
- If the `Dcm` is not able to switch fast enough from non `OBD` to `OBD` protocol, the `DSL` submodule responses with a negative response "BusyRepeatRequest" (NRC 0x21) to `OBD` tester. It is in the responsibility of the system designer to ensure that the legislative timings are satisfied.

As long as the external module processing the request is not finished (finish is indicated by returning `E_OK` or `E_NOT_OK` to `<Module>_<DiagnosticService>()/<Module>_<DiagnosticService>_<SubService>()` [API](#) call) or no timeout occurs, the `DSL` submodule responses with negative response "BusyRepeatRequest".

With receiving `E_OK` or `E_NOT_OK` from the external module to `<Module>_<DiagnosticService>()/<Module>_<DiagnosticService>_<SubService>()` [API](#) call, the `DSL` submodule will not transmit a response to old request. There will also not given any negative response to inform first tester about preemption of diagnostic request.

If the external module processing the request never returns `E_OK` or `E_NOT_OK` to `<Module>_<DiagnosticService>()/<Module>_<DiagnosticService>_<SubService>()` [API](#) call, the `DSL` submodule runs into timeout and switches directly to further processing of preempting protocol.

9.2.6 Interface to ComManager

9.2.6.1 Handling in Default Session

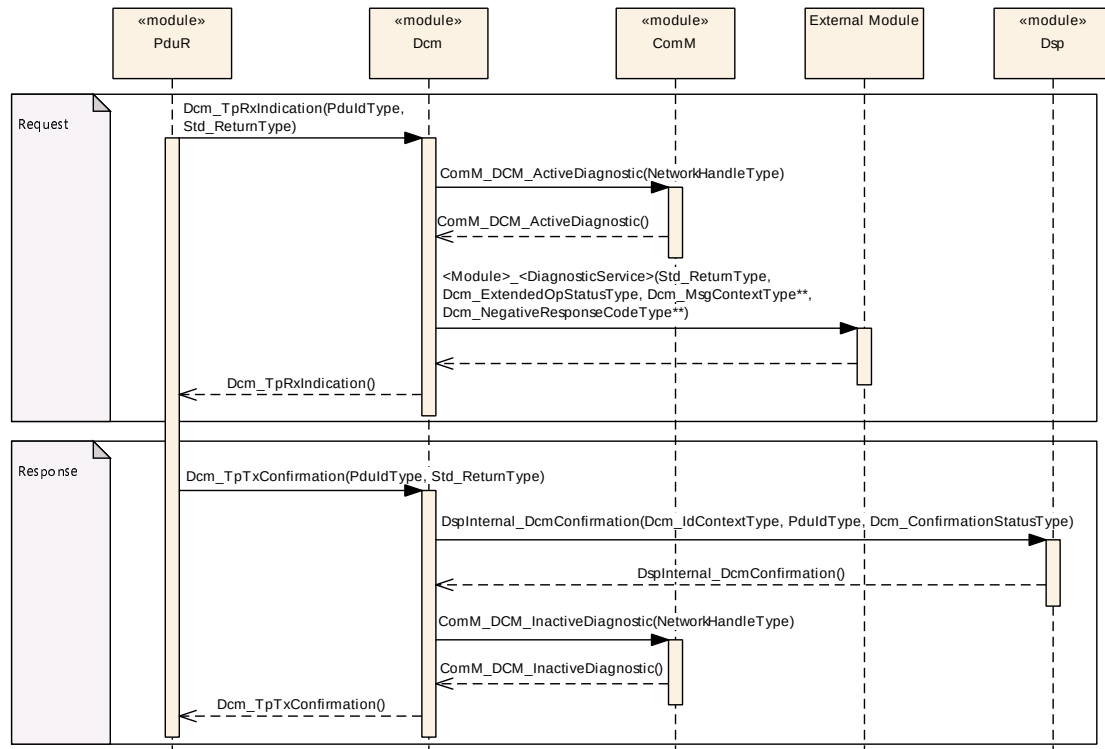


Figure 9.6: Handling in Default Session

9.2.6.2 Handling in Non-Default Session

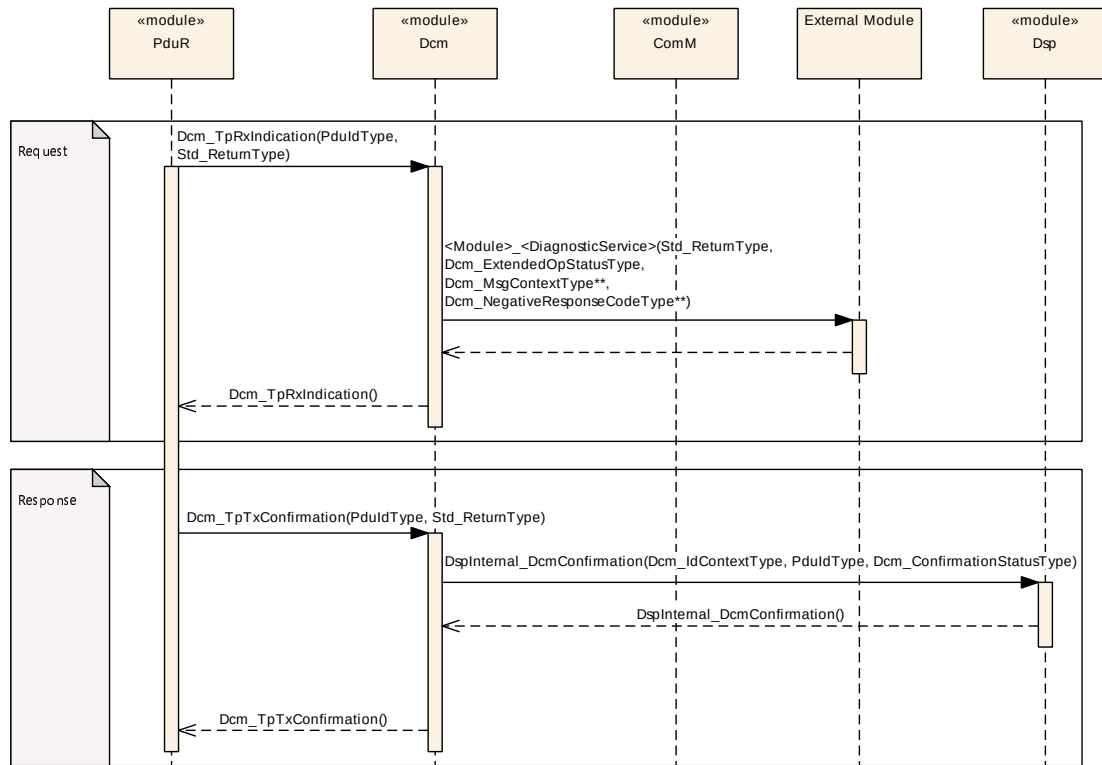


Figure 9.7: Handling in Non-Default Session

9.2.6.3 Session transitions

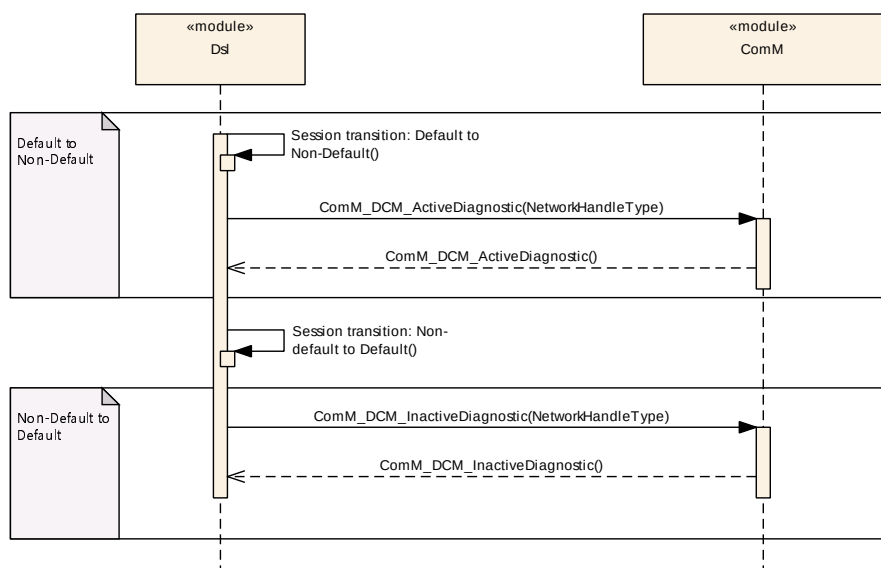


Figure 9.8: Session transitions

9.2.6.4 Communication States

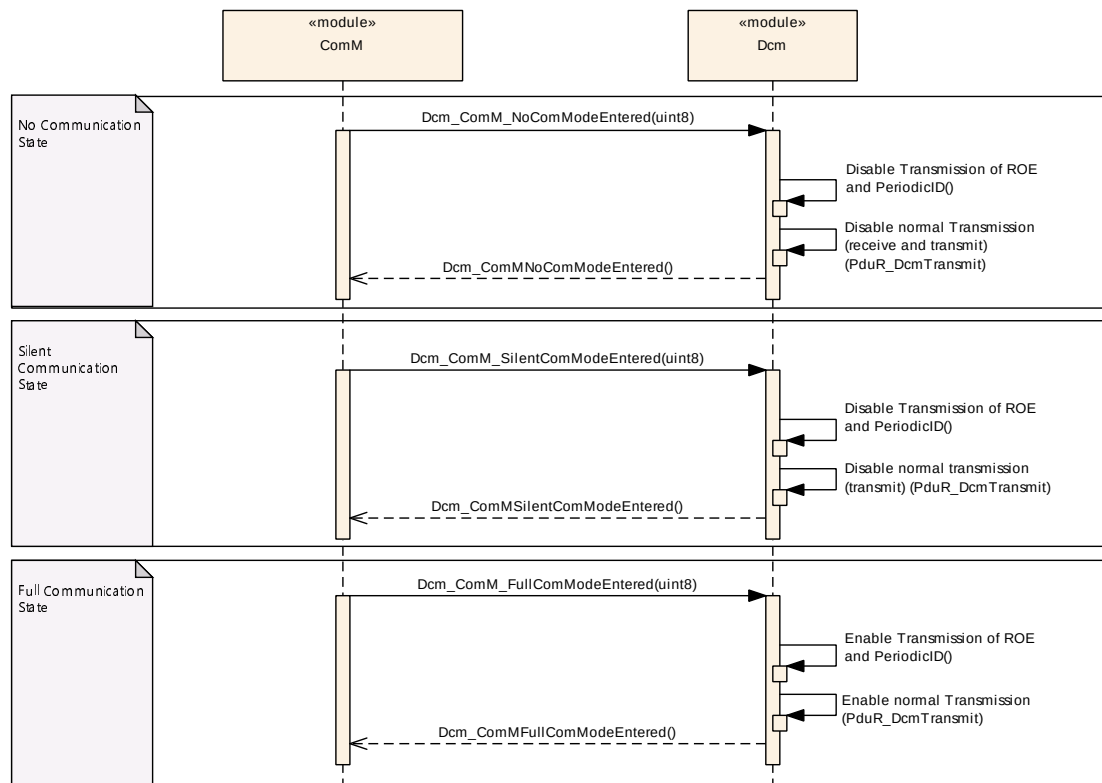


Figure 9.9: Communication States Overview

DSD (Diagnostic Service Dispatcher) Receive a request message and transmit a positive response message - synchronous transmission

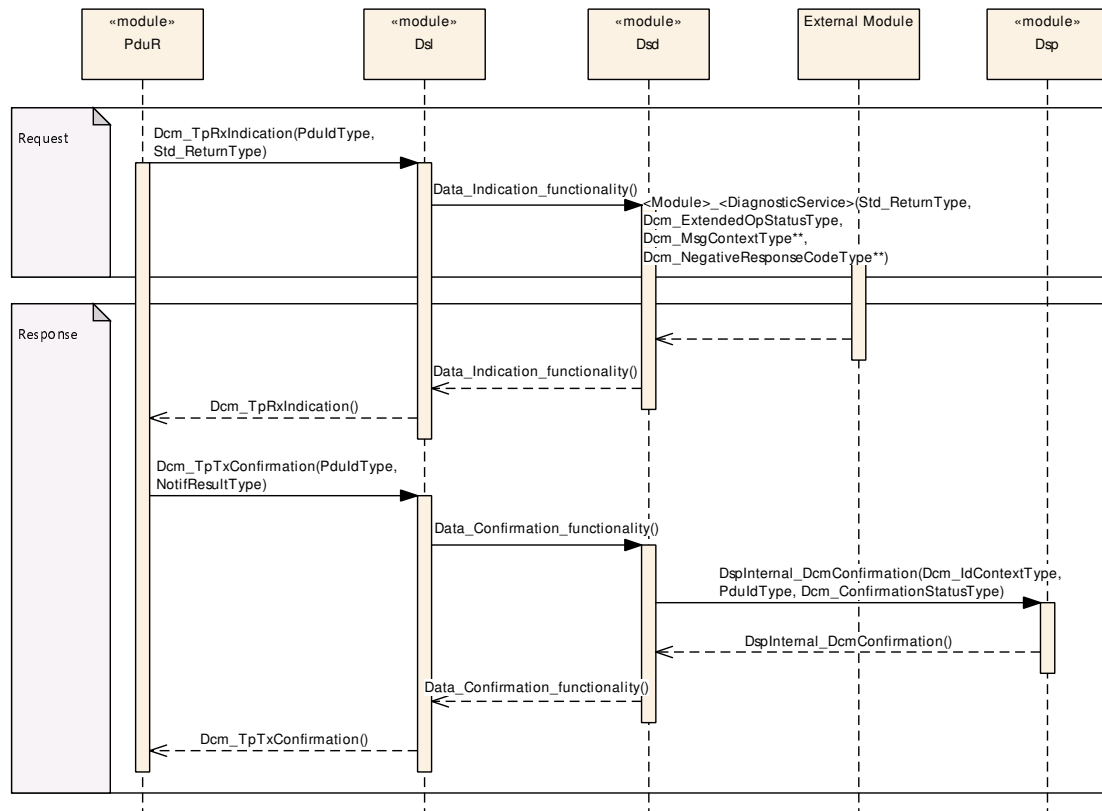


Figure 9.10: synchronous transmission

Receive a request message and transmit a positive response message - asynchronous transmission

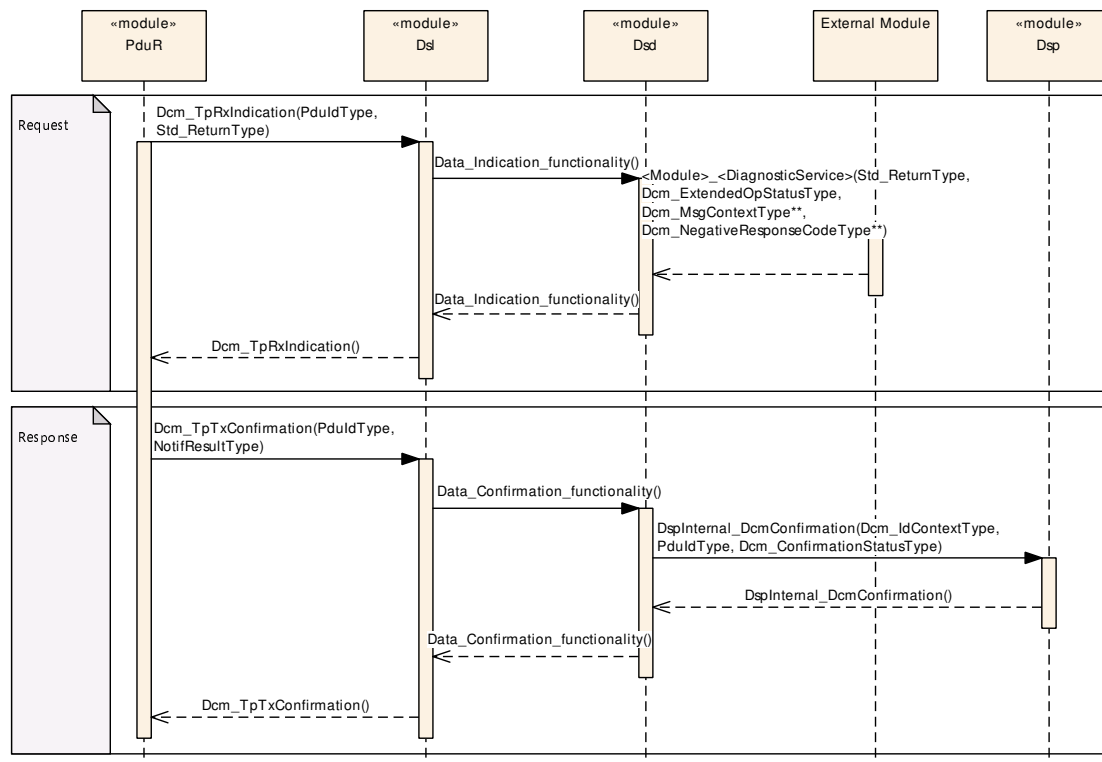


Figure 9.11: asynchronous transmission

Receive a request message and suppress a positive response

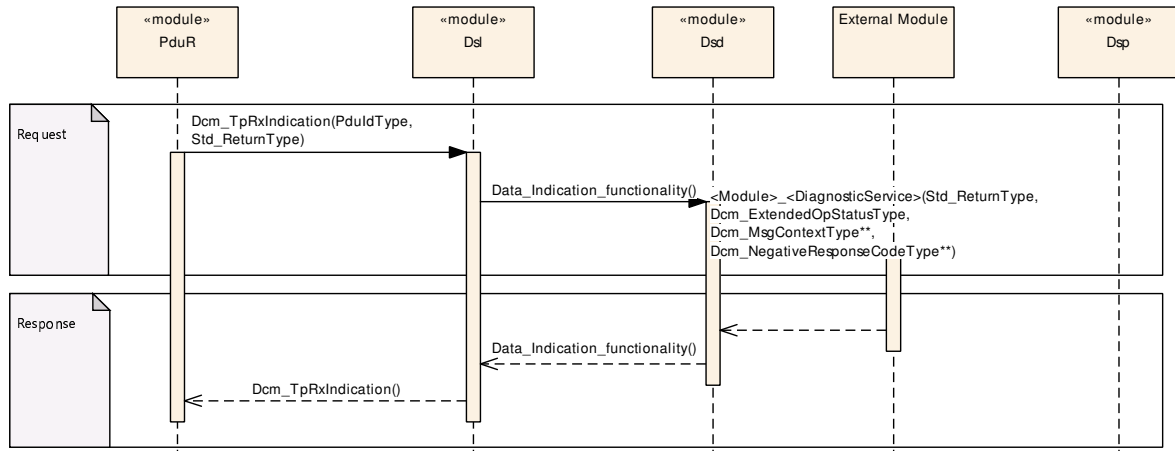


Figure 9.12: Mechanism of receiving a message and suppressing a positive response

9.2.7 Receive request message and transmit negative response message

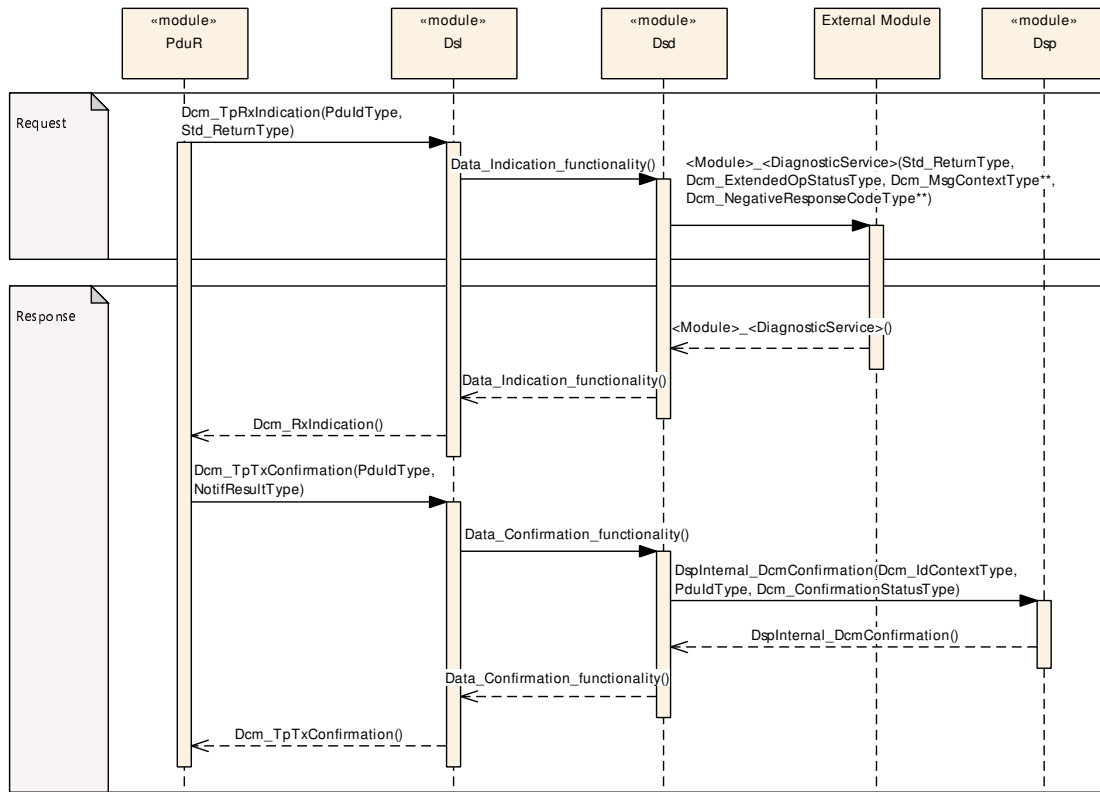


Figure 9.13: Mechanism of receiving a request message and transmitting a negative response message

9.2.8 Process Service Request with paged-buffer

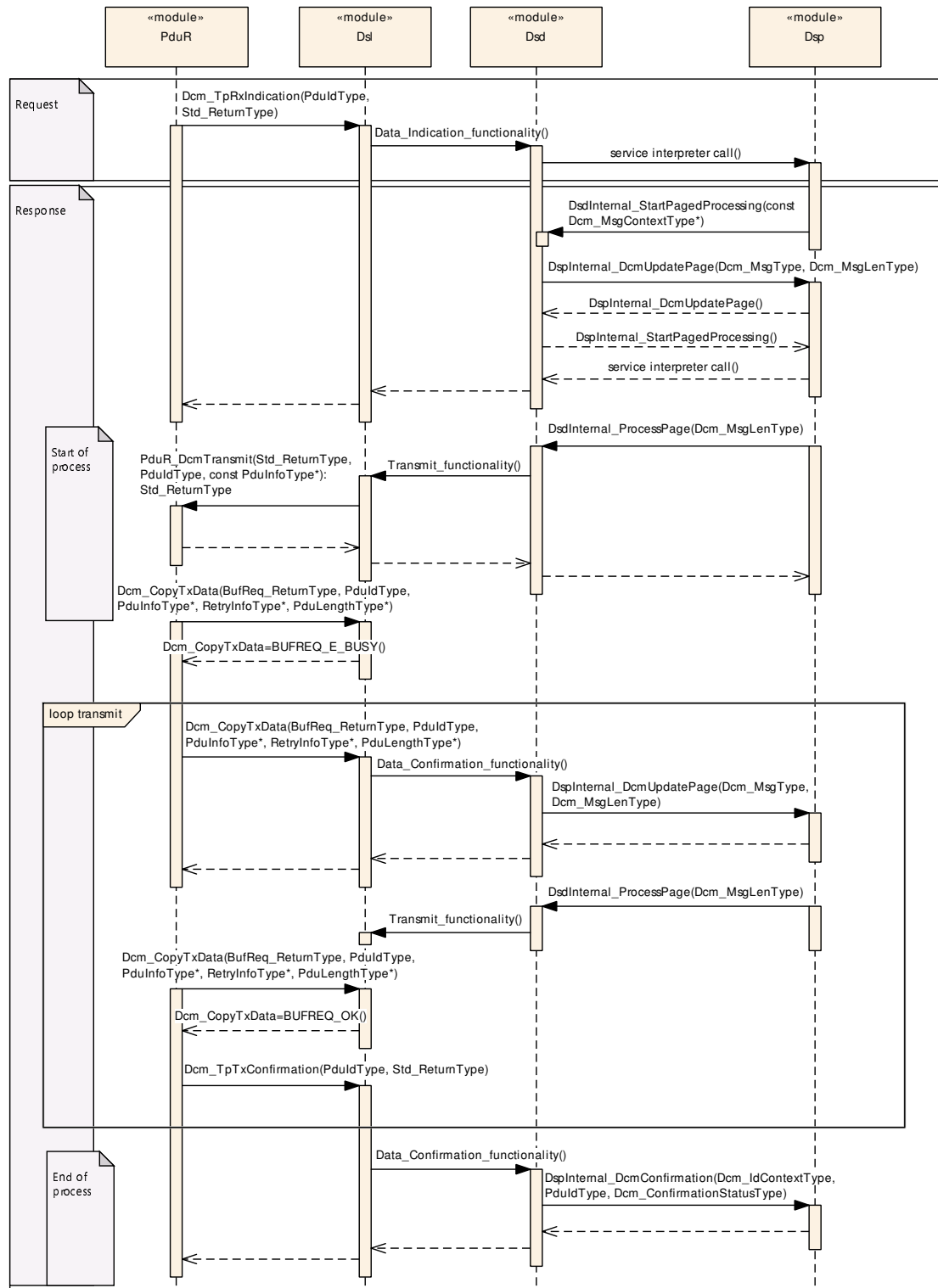


Figure 9.14: Process Service Request with paged-buffer

The following flow is processed in case no error occurs on the Application side:

Start of process:

- 4) DsdInternal_StartPagedProcessing(): With this API, the DSP submodule gives the complete response length to the Dcm module and starts paged-buffer handling. This API starts no transmission!
- 5) UpdatePage(): The Dcm module requests data to be transmitted.
- 6) DsdInternal_ProcessPage(): With this API, the DSP submodule requests transmission of the current page.
- 8) PduR_DcmTransmit(): The Dcm module requests transmission to the lower layers.
- 9) Dcm_CopyTxData: The buffer is filled and the Dcm module shall return "BUFREQ_OK"(10).

Start of the loop:

- 11) Dcm_CopyTxData: The PduR module requests the buffer but the buffer is not filled by the DSP submodule.
- 12 + 13) UpdatePage: The Dcm module requests the DSP submodule to fill the next page.
- 14) By returning "BUFREQ_E_BUSY", the Dcm module indicates that the buffer has to be filled by the DSP submodule.
- 15) DsdInternal_ProcessPage(): With this API, the DSP submodule requests transmission of the current page.
- 17) Then, on the next call of Dcm_CopyTxData the buffer is filled and the Dcm module shall return "BUFREQ_OK" (18).

LOOP: The flow 10 to 18 is repeated as long data can be sent.

End of the loop:

n-2 -> n) Dcm_TpTxConfirmation When all data is send, the PduR module indicates the sending with a confirmation, which is given to the DSP submodule.

The APIs 4, 5 and 6 are needed only for paged-buffer transmission.

Page buffer timeout handling:

The Dcm module reacts in the following described way, when the DSP submodule starts paged-buffer handling, but is not able to process further on filling the response data. E.g. there are problems to access data from an EEPROM device. When providing the Pagebuffer to the DSP submodule (13: UpdatePage()), and getting a negative Tx confirmation from underlying Transport Layers, the following error handling is carried out in the Dcm module:

- The Dcm module stops further processing of paged-buffer (item 15),
- The Dcm module requests the DSP submodule (14: DsdInternal_CancelPagedBufferProcessing()) to stop further processing of PagedBuffer.

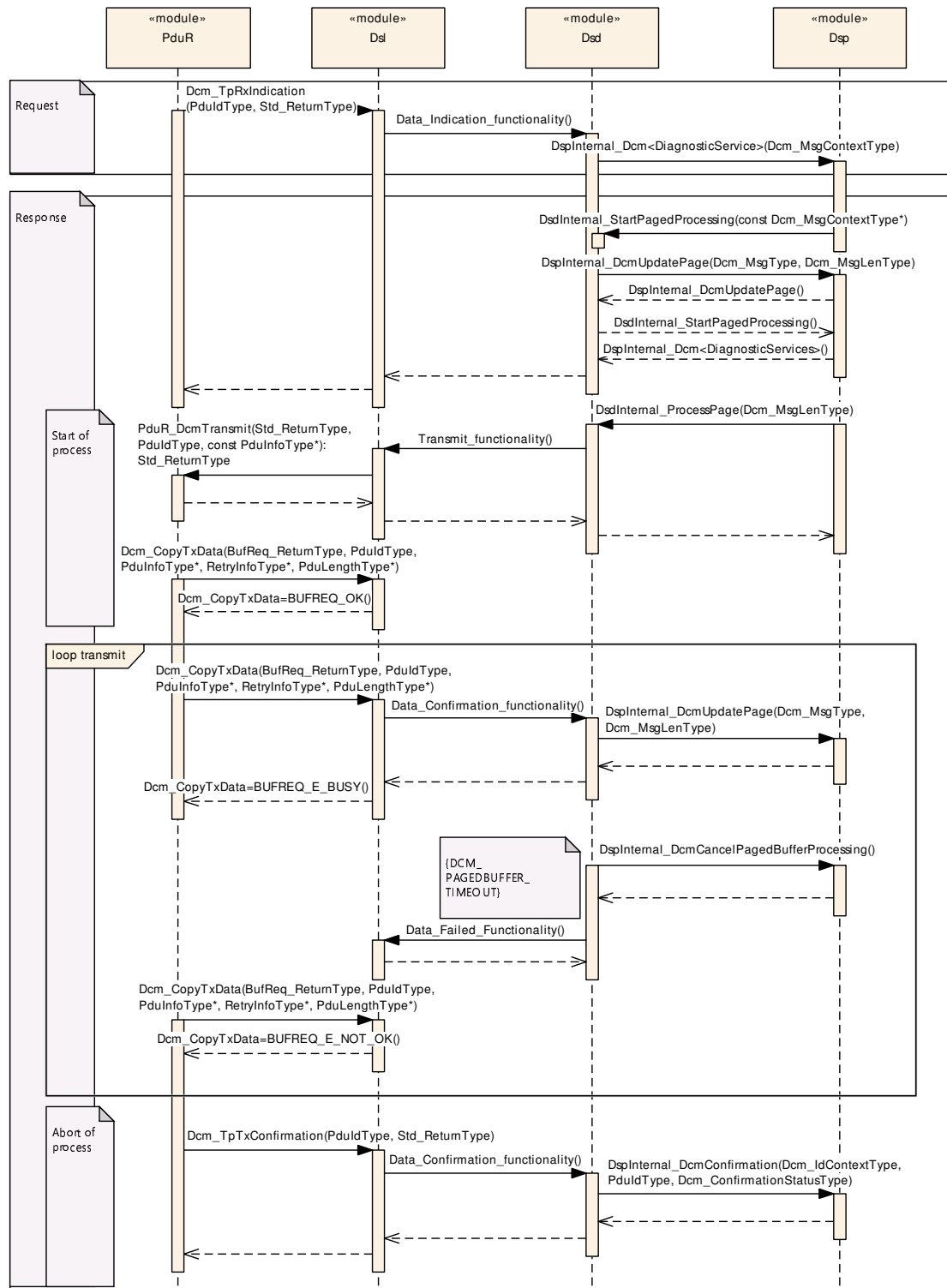


Figure 9.15: Process Service Request when paged-buffer Timeout occurs

9.2.9 Process copy data in reception

Please refer to Figure 9 "CanTp I-PDU reception" in [14, SWS PduR].

9.2.10 Process copy data in transmission

Please refer to Figure 14 "CanTp I-PDU transmission" in [14, SWS PduR].

9.3 DSP (Diagnostic Service Processing)

9.3.1 Interface DSP - DEM (service 0x19, 0x14, 0x85)

Please refer to Section 9 in [20, SWS Dem].

9.3.2 Interface special services

9.3.2.1 Process Diagnostic Session Control

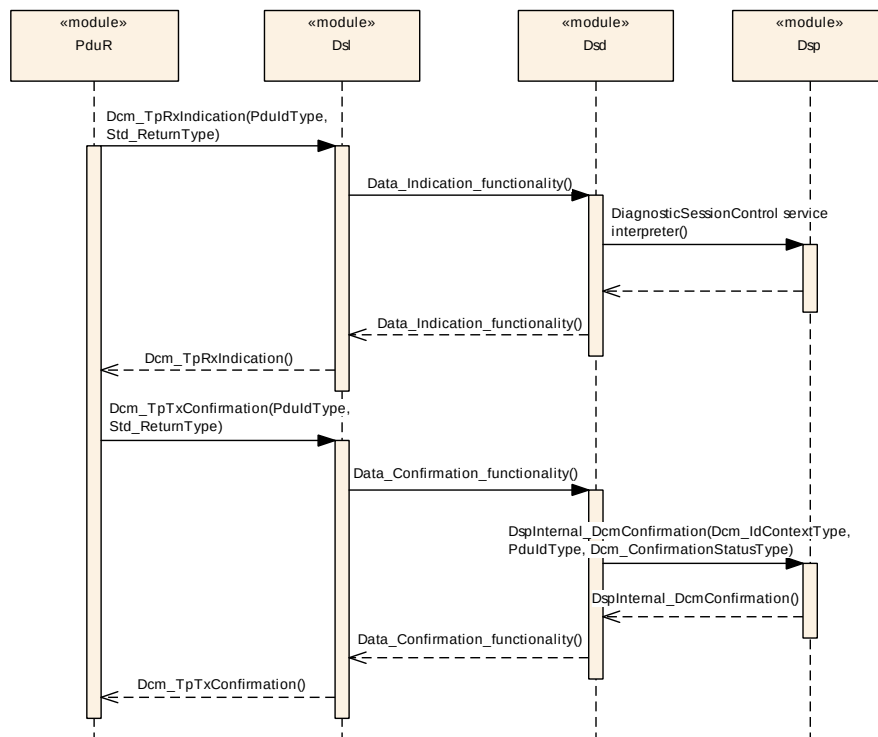


Figure 9.16: Process Diagnostic Session Control

Above sequence diagram shows processing of Diagnostic Session Control request from a tester. Note that the new diagnostic session and timing parameters only apply after the transmission confirmation of the server positive response

9.3.2.2 Process Tester Present

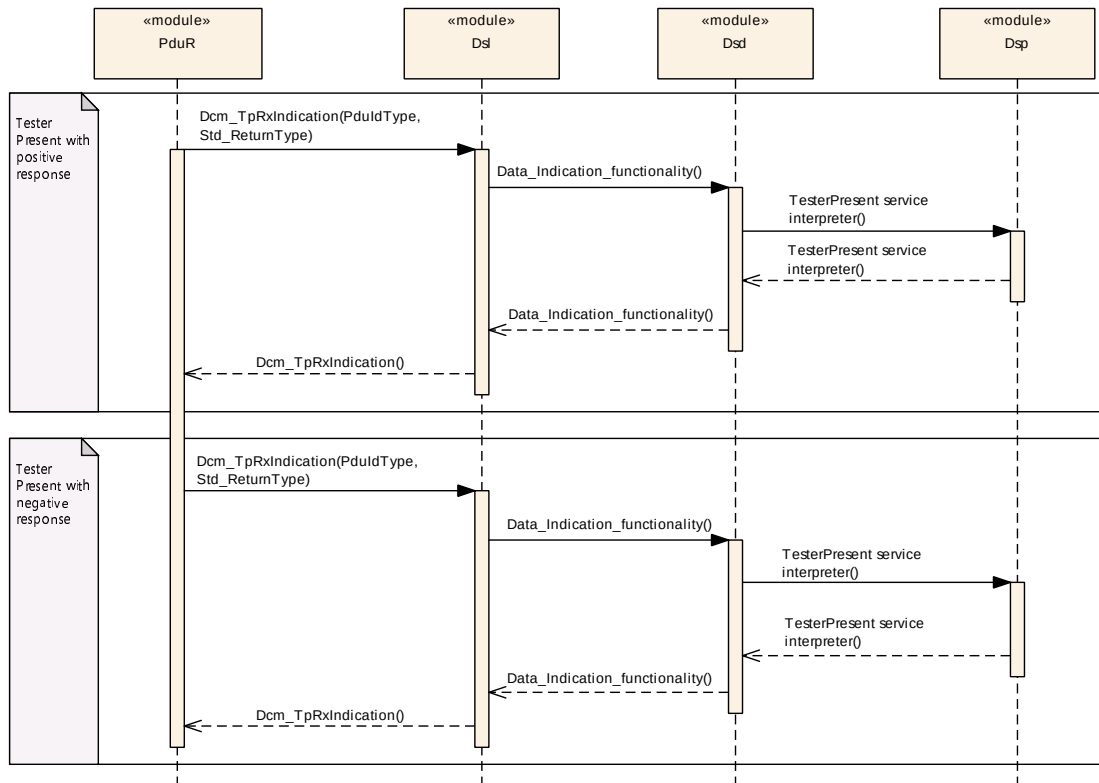


Figure 9.17: Process Tester Present

Above sequence diagram shows processing of TesterPresent commands, which are not of type functional addressed with subfunction 0x80. These TesterPresent commands are interpreted in the [DSL](#) submodule (more details can be found in Section [7.2.4.3 Concurrent TesterPresent keep alive logic](#)").

All the other TesterPresent commands are processed in the following way: On a command TesterPresent the [DSD](#) submodule calls the [DSP](#) submodule with the function TesterPresent(). The sequence chart also shows the case when an error occurs and a negative response is sent.

9.3.2.3 Process Security Access

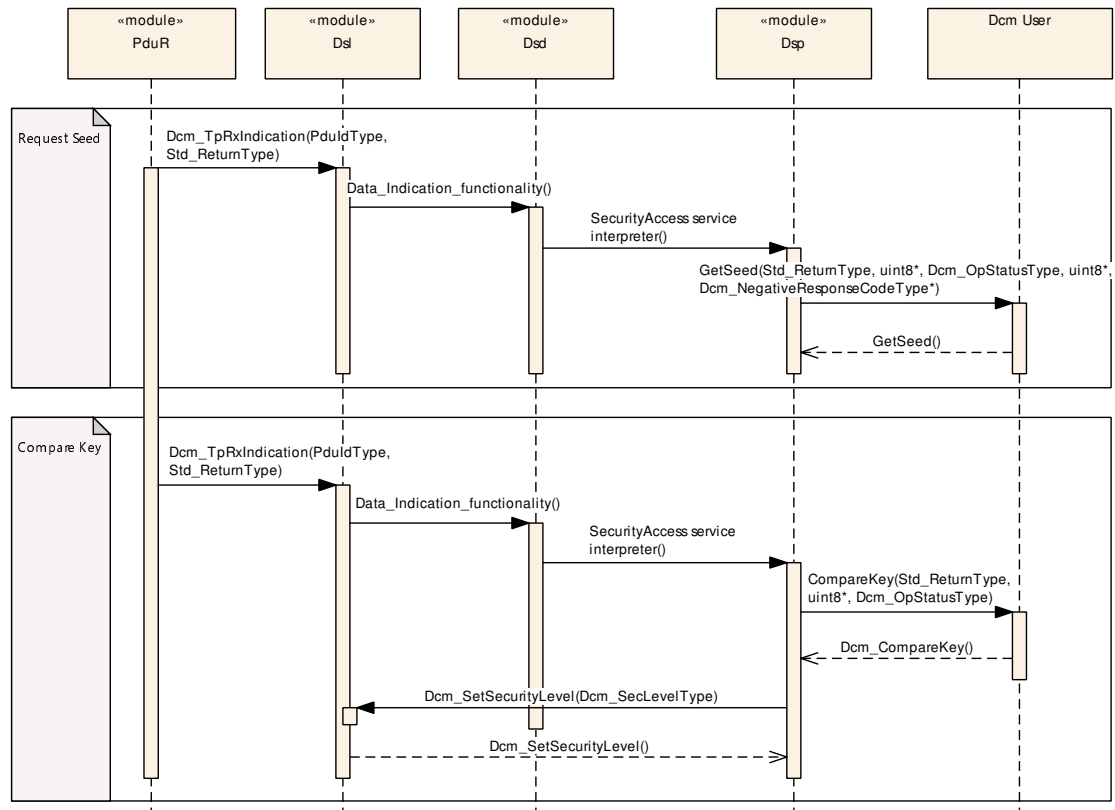


Figure 9.18: Process Security Access

To get the security access, the **DSD** submodule has to call the **DSP** submodule to get the seed value from the application. If no error is detected, the seed value is sent in the positive response.

In a second step, the **DSP** submodule gets the key calculated by the tester and requests the application to compare this key with the internal calculated key. If no error occurs, the new access type is set in the **DSL** submodule and a positive response is sent.

9.3.2.4 Process ResponseOnEvent OnDtcChange

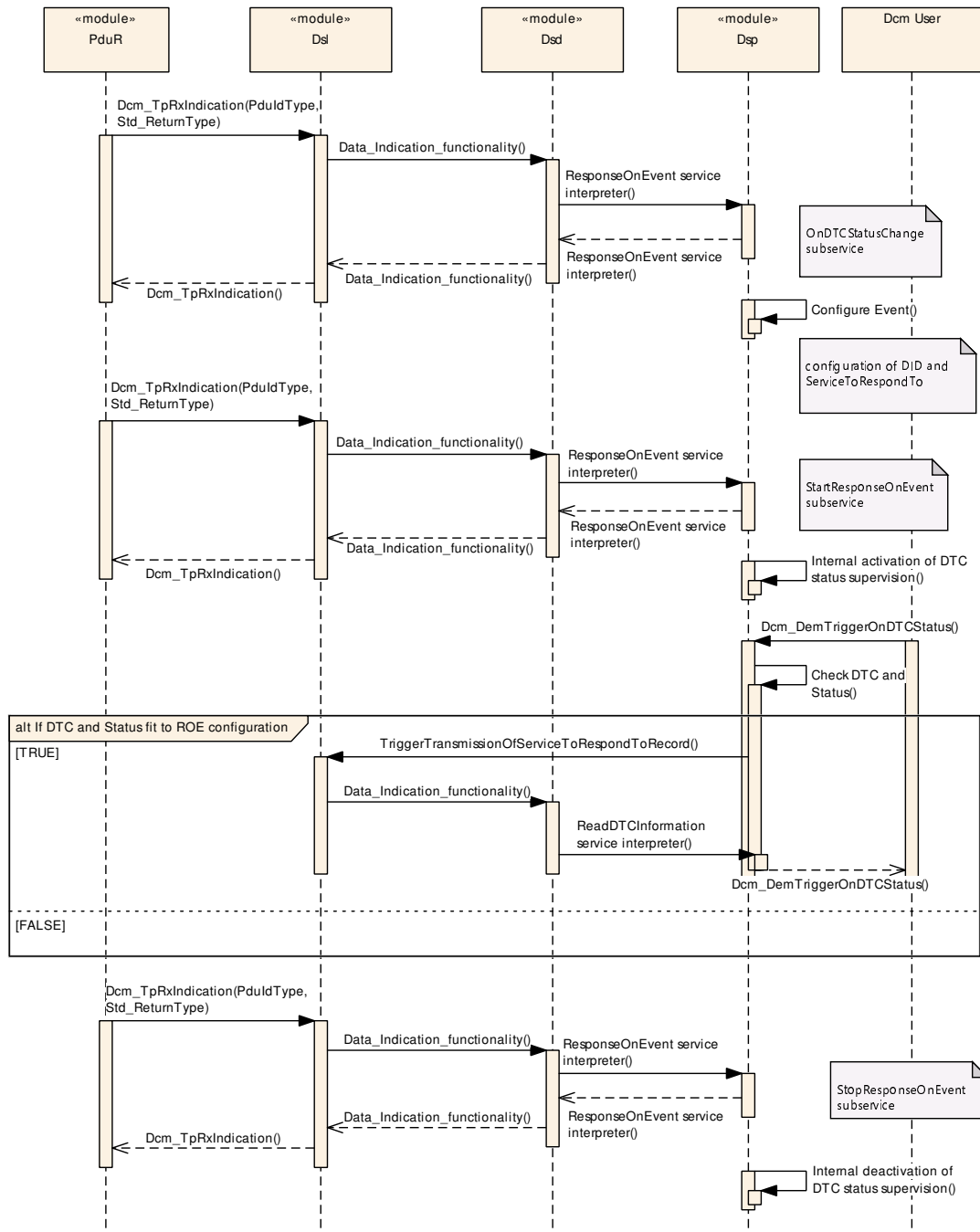


Figure 9.19: Process ResponseOnEvent OnDtcChange

Above sequence diagram shows processing of ResponseOnEvent service for sub-service OnDtcChange.

After configuration and activation of the event by the service ResponseOnEvent, the Dcm checks the status of the configured DTC on every call to interface Dcm_DemTriggerOnDTCStatus in order to identify if the event shall be trigger. This interface is

called by DEM for any **DTC** status change and independent of the activation/unactivation of ResponseOnEvent.

9.3.2.5 Process ResponseOnEvent OnChangeOfDataIdentifier

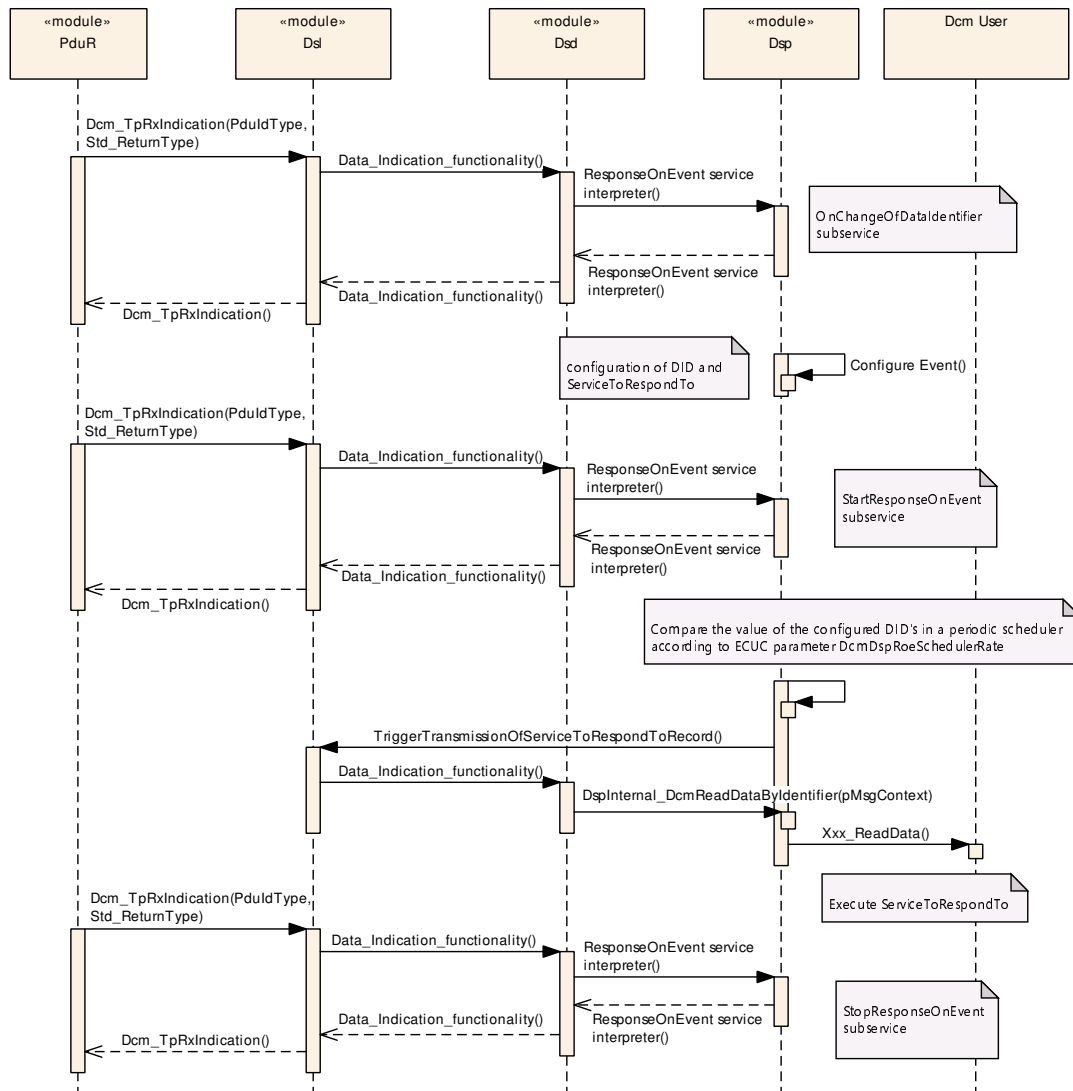


Figure 9.20: Process ResponseOnEvent OnChangeOfDataIdentifier

Above sequence diagram shows processing of ResponseOnEvent service for subservice OnChangeOfDataIdentifier. After configuration and activation of the event by the service ResponseOnEvent, **Dcm** compares values of all configured DID's every time ResponseOnEventSchedulerRate ellapses and triggers the event if the conditions are met. On occurrence of this Event, **Dcm** will execute the service configured in service-ToRespondToRecord.

9.3.2.6 Process Jump to Bootloader

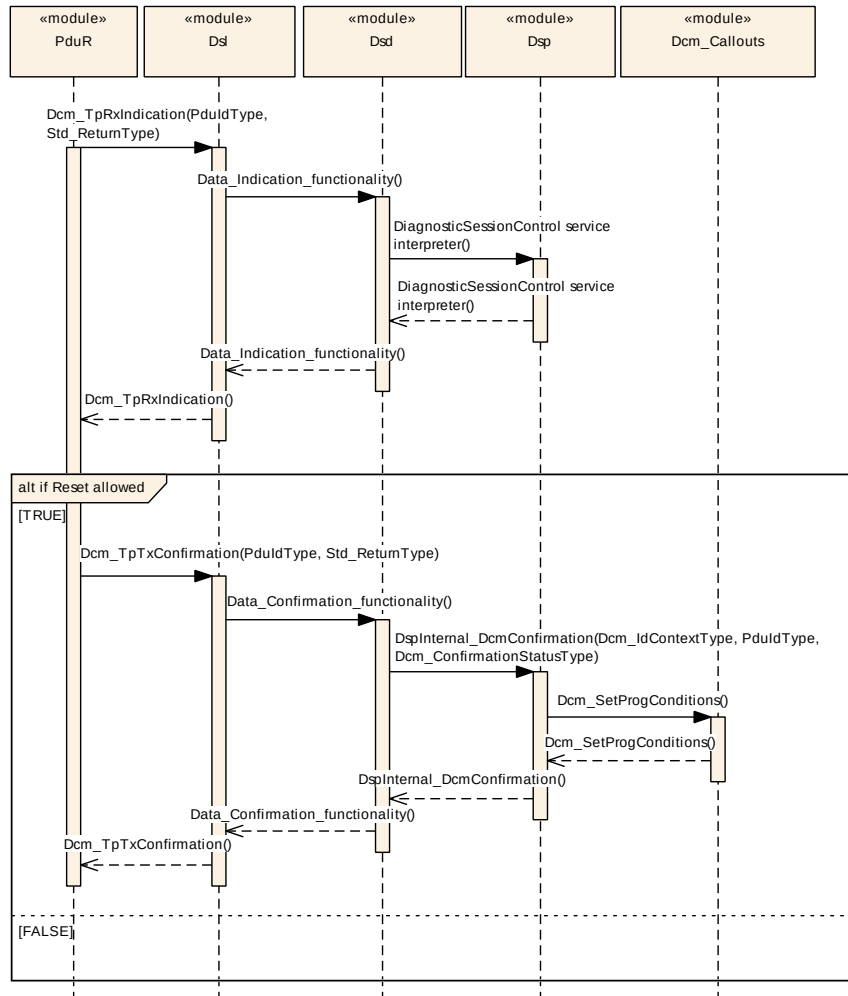


Figure 9.21: Process Jump to Bootloader

Above sequence diagram shows processing of a jump to bootloader on reception of DiagnosticSessionControl. On reception of DiagnosticSessionControl, the **Dcm** checks if the requested session is configured to trigger a jump to bootloader. In positive case, the **Dcm** start the jump to bootloader process:

- Transmission of **NRC 0x78** (ResponsePending)
- On confirmation of transmission of **NRC 0x78**, the **Dcm** calls the callout **DcmSetProgConditions** to store all information needed for the bootloader

9.3.2.7 Process IOControlRequest handling

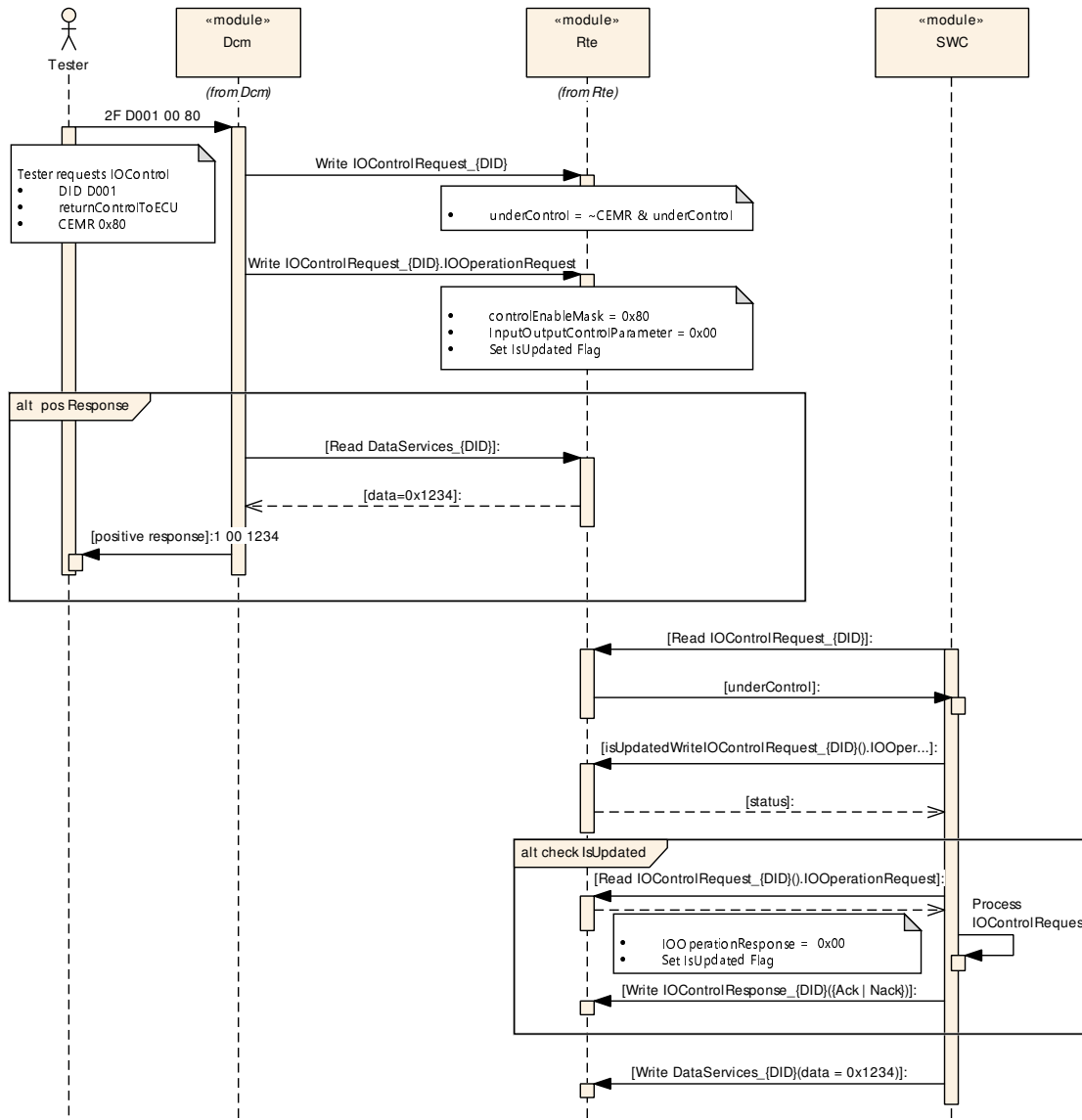


Figure 9.22: Process IOControlRequest handling with returnControlToECU as inputOutputControlParameter

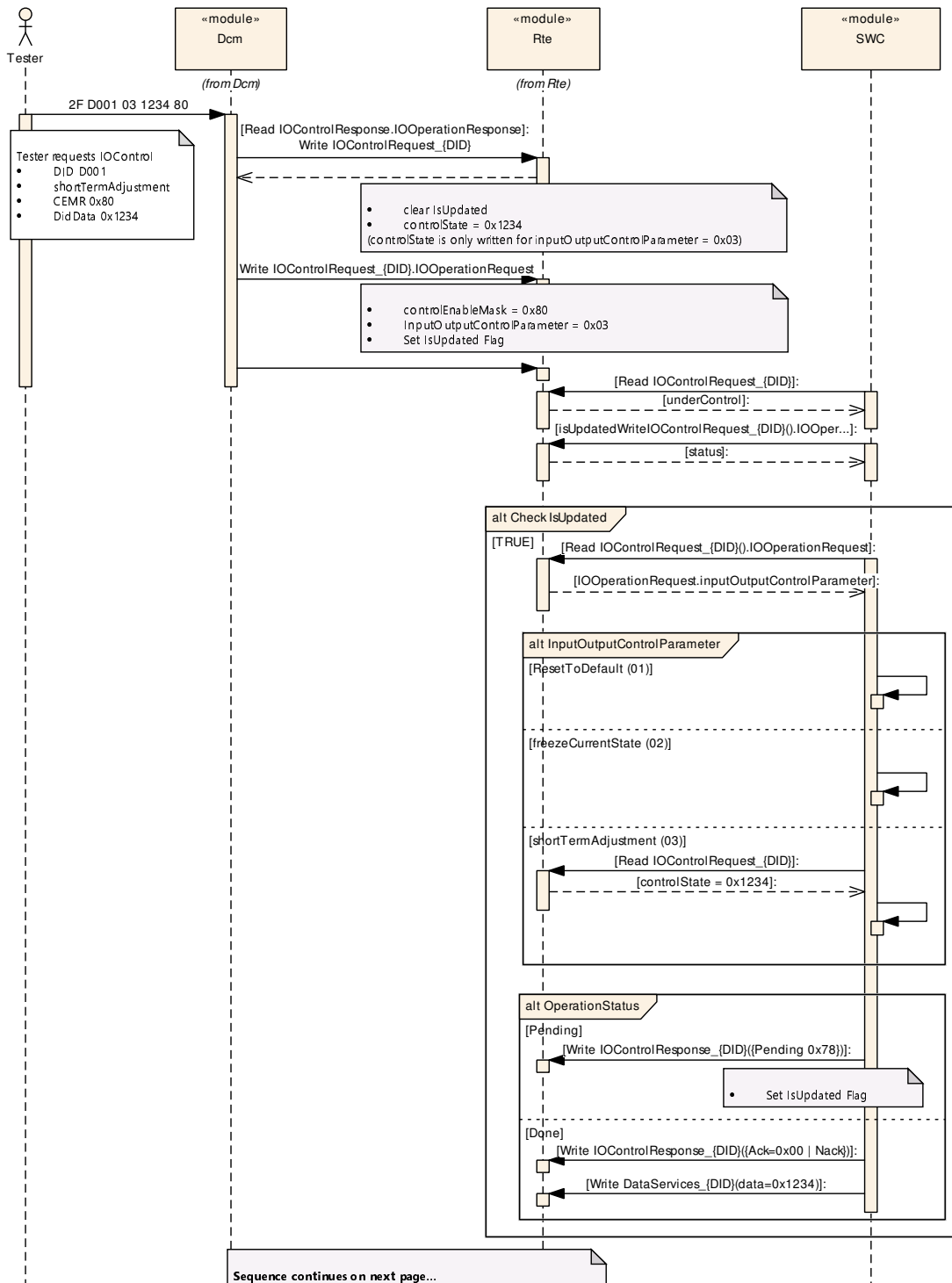


Figure 9.23: Process IOControlRequest handling with shortTermAdjustment as inputOutputControlParameter part 1

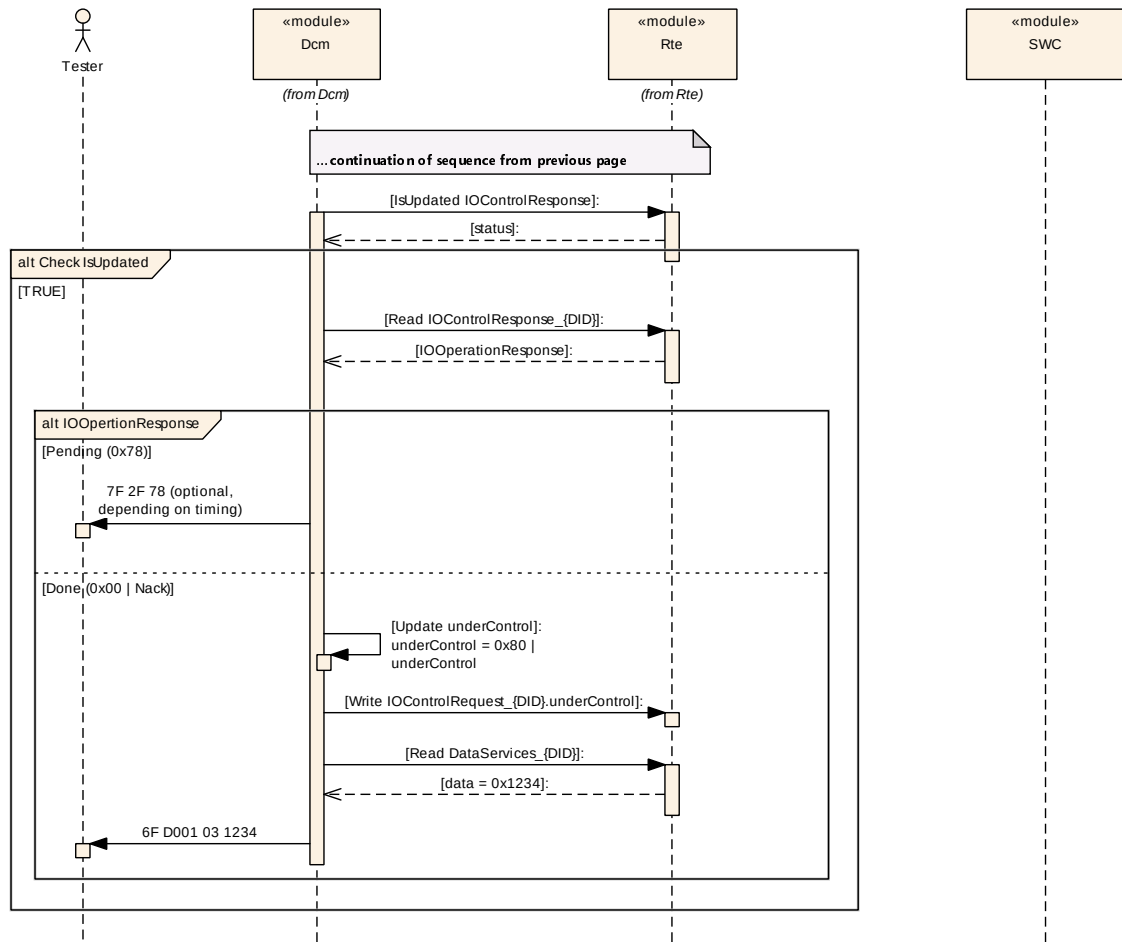


Figure 9.24: Process IOControlRequest handling with shortTermAdjustment as inputOutputControlParameter part 2

10 Configuration specification

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

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

Chapter 10.4 specifies published information of the module Dcm.

10.1 How to read this chapter

For details refer to [8] Chapter 10.1 *"Introduction to configuration specification"*.

10.2 Containers and configuration parameters

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

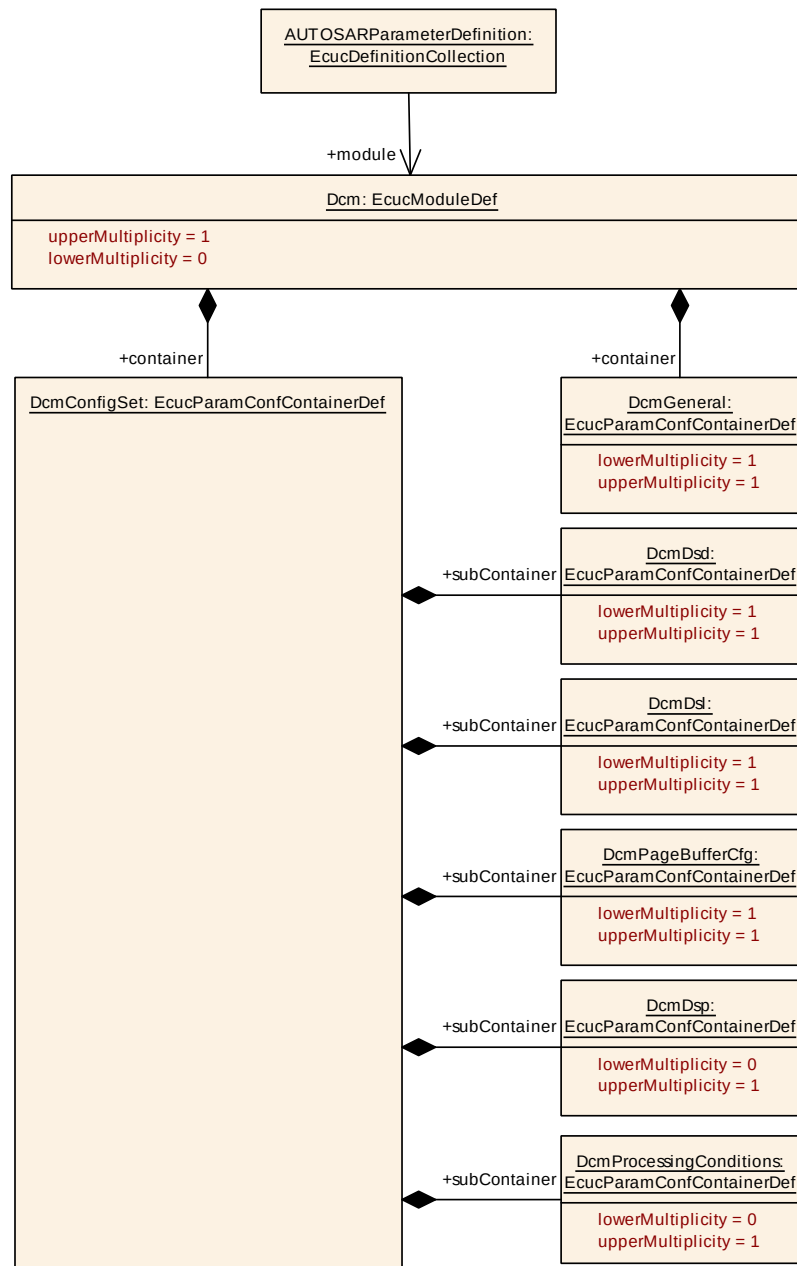


Figure 10.1: Configuration overview

10.2.1 Dcm

[ECUC_Dcm_01082] Definition of EcucModuleDef Dcm

Module Name	Dcm
Description	Configuration of the Dcm (Diagnostic Communications Manager) module.
Post-Build Variant Support	true
Supported Config Variants	VARIANT-LINK-TIME, VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

Included Containers		
Container Name	Multiplicity	Dependency
DcmConfigSet	1	This container contains the configuration parameters and sub containers of the DCM module supporting multiple configuration sets.
DcmGeneral	1	Contains general configuration parameters valid for the entire Dcm module.

]

10.2.2 DcmConfigSet

[ECUC_Dcm_00819] Definition of EcucParamConfContainerDef DcmConfigSet [

Container Name	DcmConfigSet
Parent Container	Dcm
Description	This container contains the configuration parameters and sub containers of the DCM module supporting multiple configuration sets.
Multiplicity	1
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDsd	1	These parameters configure the Diagnostic Service Dispatcher submodule.
DcmDsl	1	These parameters configure the Diagnostic Session Layer submodule.
DcmDsp	0..1	These parameters apply to Diagnostic Service Processing. There will always be one set of these parameters per Dcm. Please note: Although the multiplicity is set to 0..1. It can be expected that this container exists in any valid DCM configuration.
DcmPageBufferCfg	1	This container contains the configuration (parameters) for Page Buffer handling
DcmProcessingConditions	0..1	This container contains the configuration for mode arbitration functionality of the Dcm

]

10.2.2.1 DcmPageBufferCfg

[ECUC_Dcm_00775] Definition of EcucParamConfContainerDef DcmPageBuffer Cfg [

Container Name	DcmPageBufferCfg
Parent Container	DcmConfigSet
Description	This container contains the configuration (parameters) for Page Buffer handling
Multiplicity	1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmPagedBufferEnabled	1	[ECUC_Dcm_00776]

No Included Containers

]

[ECUC_Dcm_00776] Definition of EcucBooleanParamDef DcmPagedBufferEnabled [

Parameter Name	DcmPagedBufferEnabled		
Parent Container	DcmPageBufferCfg		
Description	Allow to enable or disable the Paged buffer mechanism. true = Paged buffer handling enabled false = Paged Buffer handling disabled		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

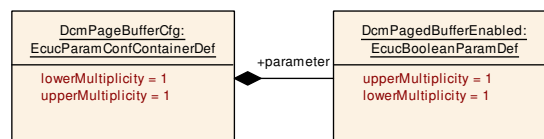


Figure 10.2: DcmPageBufferCfg configuration overview

10.2.2.2 DcmProcessingConditions

[ECUC_Dcm_00932] Definition of EcucParamConfContainerDef DcmProcessingConditions [

Container Name	DcmProcessingConditions
Parent Container	DcmConfigSet
Description	This container contains the configuration for mode arbitration functionality of the Dcm
Multiplicity	0..1
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmModeCondition	1..*	This container contains the configuration of a mode condition or an environmental conditions which can be used as argument in DcmModeRules. One DcmModeCondition shall contain either one DcmSwcModeRef or one DcmBswModeRef or one DcmSwcSRDataElementRef. Please note that the Dcm acts as well as mode manager. Therefore the references DcmSwcModeRef or one DcmBswModeRef. might point to provided ModeDeclarationGroup Prototypes of the Dcm itself as well as to provided ModeDeclarationGroupPrototypes of other Bsw Modules or software components. In case of a configured DcmSwcModeRef or DcmBswModeRef only the DcmConditionType DCM_EQUALS or DCM_EQUALS_NOT are applicable. In case of DcmSwcSRDataElementRef all literals of DcmConditionType are possible.
DcmModeRule	1..*	This container contains the configuration of a mode rule which represents a logical expression with DcmModeConditions or other DcmModeRules as arguments. All arguments are processed with the operator defined by DcmLogicalOperator, for instance: Argument_A AND Argument_B AND Argument_C

]

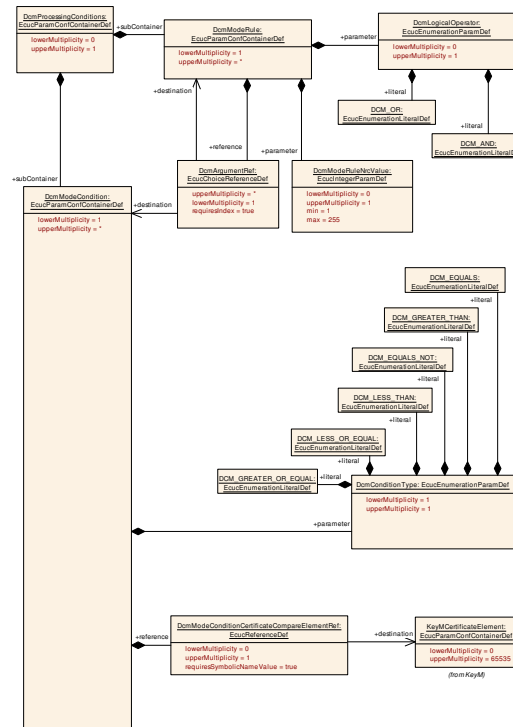


Figure 10.3: DcmDspProcessingConditions1 configuration overview

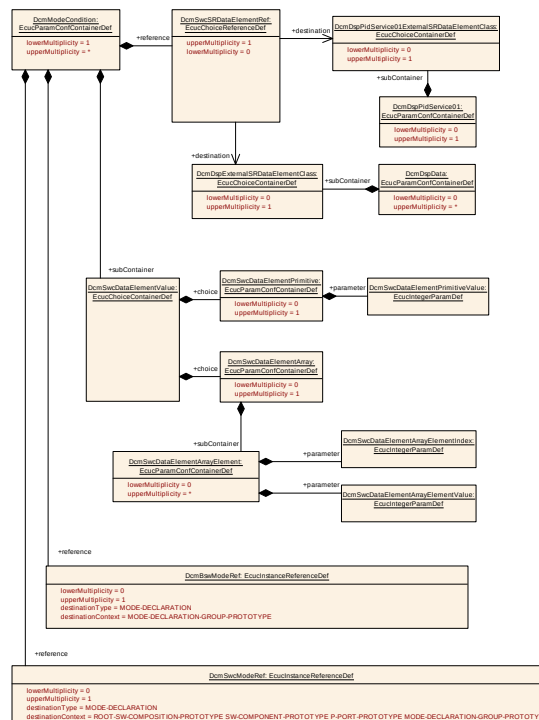


Figure 10.4: DcmDspProcessingConditions2 configuration overview

10.2.3 DcmDsd

10.2.3.1 DcmDsd

[ECUC_Dcm_00688] Definition of EcucParamConfContainerDef DcmDsd [

Container Name	DcmDsd
Parent Container	DcmConfigSet
Description	These parameters configure the Diagnostic Service Dispatcher submodule.
Multiplicity	1
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDsdServiceRequestManufacturerNotification	0..*	The name of this container is used to define the name of the R-Port through which the DCM accesses the interface ServiceRequestNotification. The R-Port is named ServiceRequestManufacturerNotification_{Name} where {Name} is the name of the container DcmDsdServiceRequestManufacturerNotification. The lowerMultiplicity is 0: If container DcmDsdServiceRequestManufacturerNotification does not exist the Indication API is not available.
DcmDsdServiceRequestSupplierNotification	0..*	The name of this container is used to define the name of the R-Port through which the DCM accesses the interface ServiceRequestNotification. The R-Port is named ServiceRequestSupplierNotification_<SWC> where <SWC> is the name of the container DcmDsdServiceRequestSupplierNotification. The lowerMultiplicity is 0: If the container DcmDsdRequestSupplierNotification does not exist the Indication API is not available.
DcmDsdServiceTable	1..256	This container contains the configuration (DSD parameters) for a Service Identifier Table. Note: It is allowed to add OBD services to a DcmDsdServiceTable related to a UDS Protocol. But it is not allowed to add UDS services to a DcmDsdServiceTable related to an OBD Protocol.

]

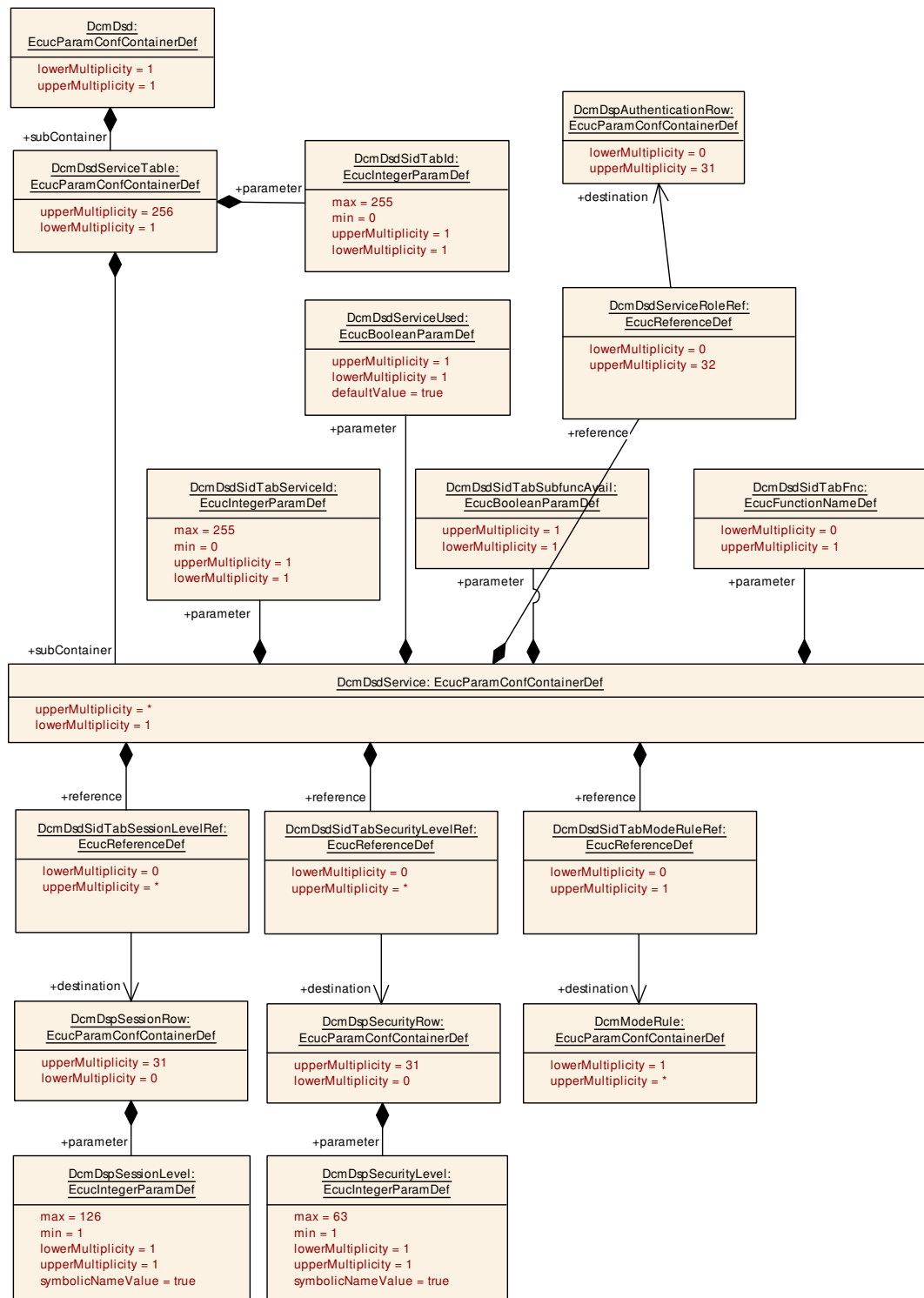


Figure 10.5: DcmDsd configuration overview

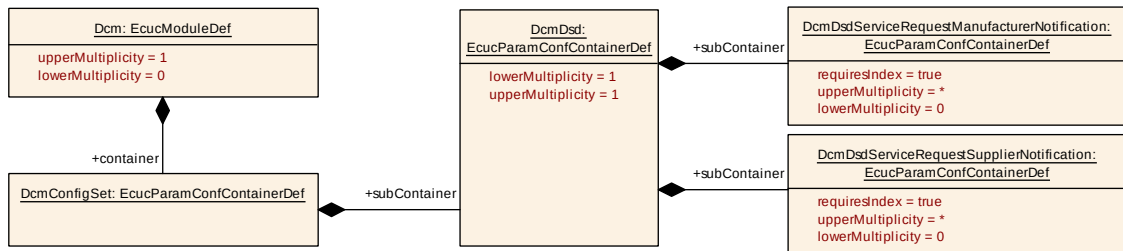


Figure 10.6: DcmDsdIndication configuration overview

10.2.3.2 DcmDsdService

[ECUC_Dcm_00689] Definition of EcucParamConfContainerDef DcmDsdService

Container Name	DcmDsdService
Parent Container	DcmDsdServiceTable
Description	This container contains the configuration (DSD parameters) for a Service.
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDsdServiceUsed	1	[ECUC_Dcm_01044]
DcmDsdSidTabFnc	0..1	[ECUC_Dcm_00777]
DcmDsdSidTabServiceId	1	[ECUC_Dcm_00735]
DcmDsdSidTabSubfuncAvail	1	[ECUC_Dcm_00737]
DcmDsdServiceRoleRef	0..32	[ECUC_Dcm_01139]
DcmDsdSidTabModeRuleRef	0..1	[ECUC_Dcm_00918]
DcmDsdSidTabSecurityLevelRef	0..*	[ECUC_Dcm_00733]
DcmDsdSidTabSessionLevelRef	0..*	[ECUC_Dcm_00734]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDsdSubService	0..*	This container contains the configuration (DSD parameters) for a subservice of a service. Only those services may have subservices, which have the DcmDsdSidTabSubfuncAvail configured as TRUE.

[ECUC_Dcm_01044] Definition of EcucBooleanParamDef DcmDsdServiceUsed [

Parameter Name	DcmDsdServiceUsed		
Parent Container	DcmDsdService		
Description	Allows to activate or deactivate the usage of a Service. This parameter can be used for multi-purpose ECUs. true - service is available false - service is not available		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

]

[ECUC_Dcm_00777] Definition of EcucFunctionNameDef DcmDsdSidTabFnc [

Parameter Name	DcmDsdSidTabFnc		
Parent Container	DcmDsdService		
Description	Callback function of the ECU Supplier specific component for the particular service. The function's prototype is as described for <Module>_<DiagnosticService>. If this parameter is not configured, the service is handled Dcm-internally.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00735] Definition of EcucIntegerParamDef DcmDsdSidTabServiceId [

Parameter Name	DcmDsdSidTabServiceId		
Parent Container	DcmDsdService		
Description	Identifier of the service. The possible service identifiers are defined in ISO 14229-1 and ISO 15031-5.		
Multiplicity	1		
Type	EcucIntegerParamDef		





Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00737] Definition of EcucBooleanParamDef DcmDsdSidTabSubfuncAvail [

Parameter Name	DcmDsdSidTabSubfuncAvail		
Parent Container	DcmDsdService		
Description	Information about whether the service has subfunctions or not. This parameter is used for the handling of the "suppressPosRspMsgIndicationBit" as defined in ISO 14229-1, which can be used as a reference for the configuration. true - service has subfunctions, suppressPosRspMsgIndicationBit is available false - service has no subfunctions, suppressPosRspMsgIndicationBit is not available		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01139] Definition of EcucReferenceDef DcmDsdServiceRoleRef [

Parameter Name	DcmDsdServiceRoleRef		
Parent Container	DcmDsdService		
Description	Reference to DcmDspAuthenticationRow that defines a role in that the service is allowed to be executed.		
Multiplicity	0..32		
Type	Reference to DcmDspAuthenticationRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00918] Definition of EcucReferenceDef DcmDsdSidTabModeRuleRef [

Parameter Name	DcmDsdSidTabModeRuleRef		
Parent Container	DcmDsdService		
Description	Reference to a DcmDspModeRule which controls the execution of the service. If there is no reference configured, no mode rule check shall be performed.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00733] Definition of EcucReferenceDef DcmDsdSidTabSecurityLevelRef [

Parameter Name	DcmDsdSidTabSecurityLevelRef		
Parent Container	DcmDsdService		
Description	Reference to a Security Level in which the service is allowed to be executed. Multiple references are allowed for a service. Please refer to ISO 14229-1, ISO 15031-5 and chapter "Verification of the Service Security Access levels." If there is no reference configured, no service security verification shall be performed.		
Multiplicity	0..*		
Type	Reference to DcmDspSecurityRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00734] Definition of EcucReferenceDef DcmDsdSidTabSessionLevelRef [

Parameter Name	DcmDsdSidTabSessionLevelRef		
Parent Container	DcmDsdService		
Description	Reference to a Session Level in which the service is allowed to be executed. Multiple references are allowed for a service. Please refer to ISO 14229-1, ISO 15031-5 and chapter "Verification of the Diagnostic Session". If there is no reference configured, no diagnostic session verification shall be performed.		
Multiplicity	0..*		
Type	Reference to DcmDspSessionRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

10.2.3.3 DcmDsdServiceRequestManufacturerNotification

[ECUC_Dcm_00681] Definition of EcucParamConfContainerDef DcmDsdServiceRequestManufacturerNotification [

Container Name	DcmDsdServiceRequestManufacturerNotification		
Parent Container	DcmDsd		
Description	The name of this container is used to define the name of the R-Port through which the DCM accesses the interface ServiceRequestNotification. The R-Port is named ServiceRequestManufacturerNotification_{Name} where {Name} is the name of the container DcmDsdServiceRequestManufacturerNotification. The lowerMultiplicity is 0: If container DcmDsdServiceRequestManufacturerNotification does not exist the Indication API is not available. Attributes: requiresIndex=true		
Multiplicity	0..*		
Configuration Parameters			

No Included Parameters

No Included Containers

]

10.2.3.4 DcmDsdServiceRequestSupplierNotification

[ECUC_Dcm_00816] Definition of EcucParamConfContainerDef DcmDsdServiceRequestSupplierNotification [

Container Name	DcmDsdServiceRequestSupplierNotification
Parent Container	DcmDsd
Description	The name of this container is used to define the name of the R-Port through which the DCM accesses the interface ServiceRequestNotification. The R-Port is named ServiceRequestSupplierNotification_<SWC> where <SWC> is the name of the container DcmDsdServiceRequestSupplierNotification. The lowerMultiplicity is 0: If the container DcmDsdRequestSupplierNotification does not exist the Indication API is not available. Attributes: requiresIndex=true
Multiplicity	0..*
Configuration Parameters	

No Included Parameters

No Included Containers

]

10.2.3.5 DcmDsdServiceTable

[ECUC_Dcm_00732] Definition of EcucParamConfContainerDef DcmDsdServiceTable [

Container Name	DcmDsdServiceTable
Parent Container	DcmDsd
Description	This container contains the configuration (DSD parameters) for a Service Identifier Table. Note: It is allowed to add OBD services to a DcmDsdServiceTable related to a UDS Protocol. But it is not allowed to add UDS services to a DcmDsdServiceTable related to an OBD Protocol.
Multiplicity	1..256
Configuration Parameters	

Included Parameters

Parameter Name	Multiplicity	ECUC ID
DcmDsdSidTabId	1	[ECUC_Dcm_00736]

Included Containers

Container Name	Multiplicity	Dependency
DcmDsdService	1..*	This container contains the configuration (DSD parameters) for a Service.

]

[ECUC_Dcm_00736] Definition of EcucIntegerParamDef DcmDsdSidTabId [

Parameter Name	DcmDsdSidTabId		
Parent Container	DcmDsdServiceTable		
Description	Due to using possibly more service tables, the unique DcmDsdSidTabId can be used to identify them.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

Note : The [Dcm](#) internal interaction with the [DSP](#) is implementation specific and therefore not explicitly configured.

10.2.3.6 DcmDsdSubService

[ECUC_Dcm_00802] Definition of EcucParamConfContainerDef DcmDsdSubService [

Container Name	DcmDsdSubService		
Parent Container	DcmDsdService		
Description	This container contains the configuration (DSD parameters) for a subservice of a service. Only those services may have subservices, which have the DcmDsdSidTabSubfuncAvail configured as TRUE.		
Multiplicity	0..*		
Configuration Parameters			

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDsdSubServiceFnc	0..1	[ECUC_Dcm_00942]
DcmDsdSubServiceId	1	[ECUC_Dcm_00803]
DcmDsdSubServiceUsed	1	[ECUC_Dcm_01047]
DcmDsdSubServiceModeRuleRef	0..1	[ECUC_Dcm_00924]
DcmDsdSubServiceRoleRef	0..32	[ECUC_Dcm_01140]
DcmDsdSubServiceSecurityLevelRef	0..*	[ECUC_Dcm_00812]
DcmDsdSubServiceSessionLevelRef	0..*	[ECUC_Dcm_00804]

No Included Containers

]

[ECUC_Dcm_00942] Definition of EcucFunctionNameDef DcmDsdSubService Fnc

Parameter Name	DcmDsdSubServiceFnc		
Parent Container	DcmDsdSubService		
Description	Callback function of the ECU Supplier specific component for the particular service. The function's prototype is as described for <Module>_<DiagnosticService>_<SubService>. If this parameter is not configured, the subservice is handled Dcm-internally.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_00803] Definition of EcucIntegerParamDef DcmDsdSubServiceId

Parameter Name	DcmDsdSubServiceId		
Parent Container	DcmDsdSubService		
Description	Identifier of the subservice. The possible subservice identifiers are defined in ISO 14229-1 and ISO 15031-5.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 127		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_01047] Definition of EcucBooleanParamDef DcmDsdSubServiceUsed

Parameter Name	DcmDsdSubServiceUsed		
Parent Container	DcmDsdSubService		
Description	Allows to activate or deactivate the usage of a Subservice. This parameter can be used for multi-purpose ECUs. true - subservice is available false - subservice is not available.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

[ECUC_Dcm_00924] Definition of EcucReferenceDef DcmDsdSubServiceModeRuleRef

Parameter Name	DcmDsdSubServiceModeRuleRef		
Parent Container	DcmDsdSubService		
Description	Reference to a DcmDspModeRule which controls the execution of the subservice. If there is no reference configured, no mode rule check shall be performed.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_01140] Definition of EcucReferenceDef DcmDsdSubServiceRoleRef

Parameter Name	DcmDsdSubServiceRoleRef		
Parent Container	DcmDsdSubService		
Description	Reference to DcmDspAuthenticationRow that defines a role in that the service with this subfunction is allowed to be executed.		
Multiplicity	0..32		
Type	Reference to DcmDspAuthenticationRow		
Post-Build Variant Multiplicity	false		





Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00812] Definition of EcucReferenceDef DcmDsdSubServiceSecurityLevelRef

Parameter Name	DcmDsdSubServiceSecurityLevelRef		
Parent Container	DcmDsdSubService		
Description	Reference to a Security Level in which the subservice is allowed to be executed. Multiple references are allowed for a subservice. Please refer to ISO 14229-1, ISO 15031-5 and chapter "Verification of the Service Security Access levels." If there is no reference configured, no subservice security verification shall be performed.		
Multiplicity	0..*		
Type	Reference to DcmDspSecurityRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00804] Definition of EcucReferenceDef DcmDsdSubServiceSessionLevelRef

Parameter Name	DcmDsdSubServiceSessionLevelRef		
Parent Container	DcmDsdSubService		
Description	Reference to a Session Level in which the subservice is allowed to be executed. Multiple references are allowed for a subservice. Please refer to ISO 14229-1, ISO 15031-5 and chapter "Verification of the Diagnostic Session". If there is no reference configured, no diagnostic session verification shall be performed.		
Multiplicity	0..*		
Type	Reference to DcmDspSessionRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD





Value Configuration Class	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

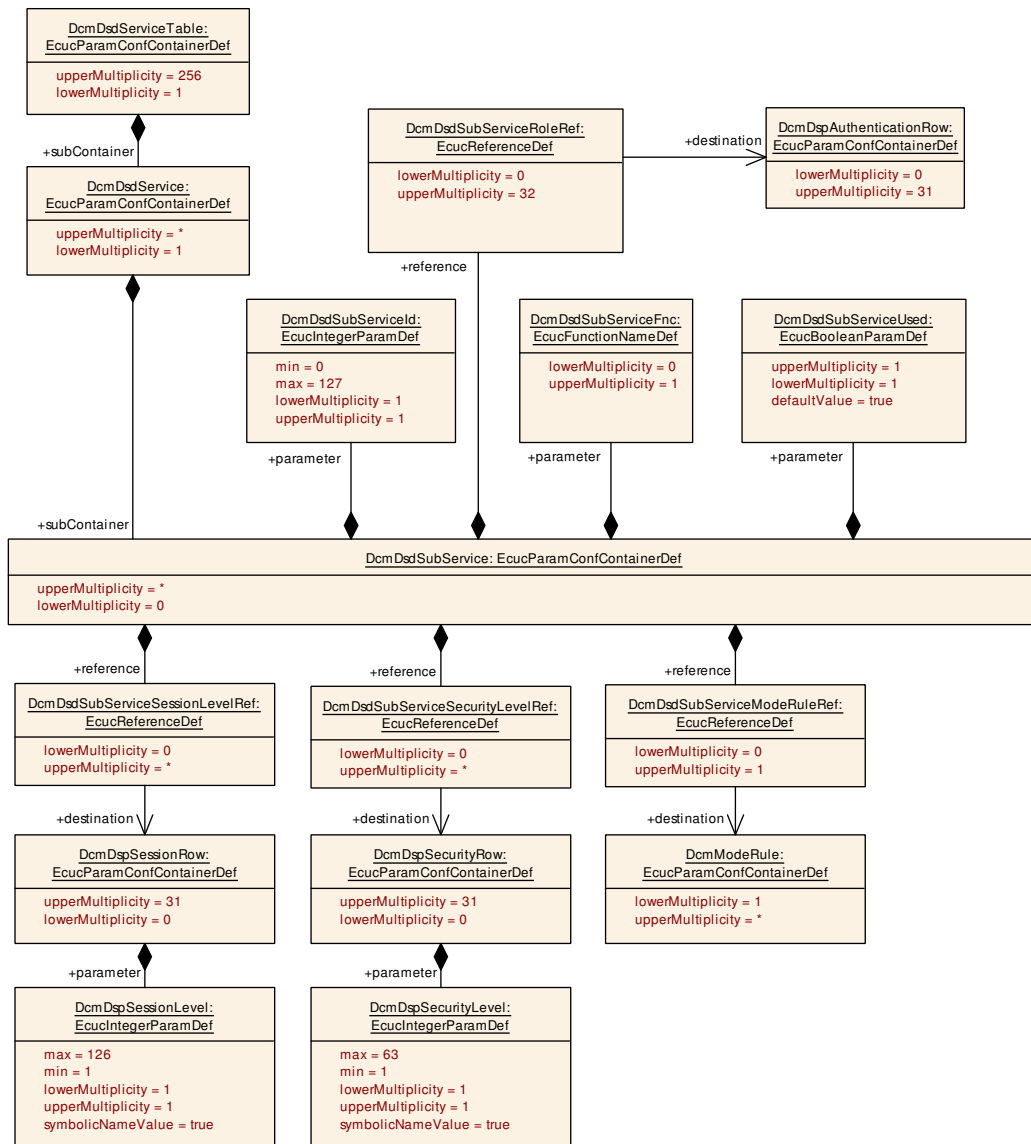


Figure 10.7: DcmDsdSubService configuration overview

10.2.4 DcmDsl

10.2.4.1 DcmDsl

[ECUC_Dcm_00690] Definition of EcucParamConfContainerDef DcmDsl [

Container Name	DcmDsl
Parent Container	DcmConfigSet
Description	These parameters configure the Diagnostic Session Layer submodule.
Multiplicity	1
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDslBuffer	1..256	This container contains the configuration of a diagnostic buffer.
DcmDslDiagResp	1	This container contains the configuration of the automatic requestCorrectlyReceivedResponsePending response management in the Dcm.
DcmDslProtocol	1	This container contains the configurations of the diagnostic protocols used in Dcm.

]

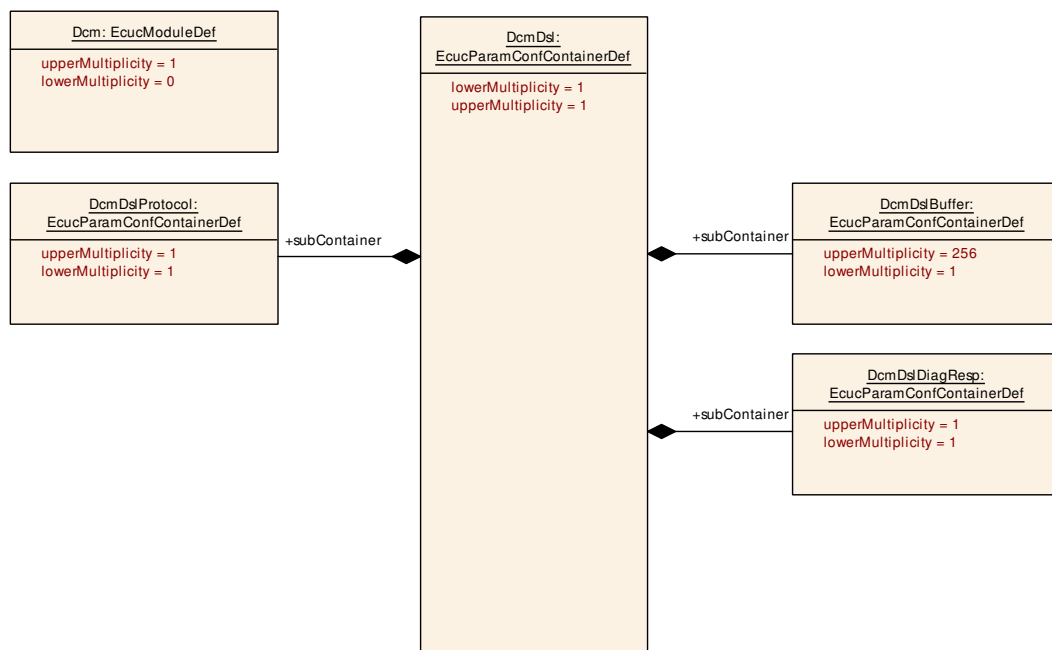


Figure 10.8: DcmDsl configuration overview

10.2.4.2 DcmDslBuffer

[ECUC_Dcm_00739] Definition of EcucParamConfContainerDef DcmDslBuffer [

Container Name	DcmDslBuffer
Parent Container	DcmDsl
Description	This container contains the configuration of a diagnostic buffer.
Multiplicity	1..256
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDslBufferSize	1	[ECUC_Dcm_00738]

No Included Containers

]

[[ECUC_Dcm_00738](#)] Definition of EcucIntegerParamDef DcmDslBufferSize [

Parameter Name	DcmDslBufferSize		
Parent Container	DcmDslBuffer		
Description	Size of the diagnostic buffer in bytes. For a linear buffer the size shall be as large as the longest diagnostic message (request or response). For a paged buffer the size has impacts on the application performance.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	8 .. 4294967294		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

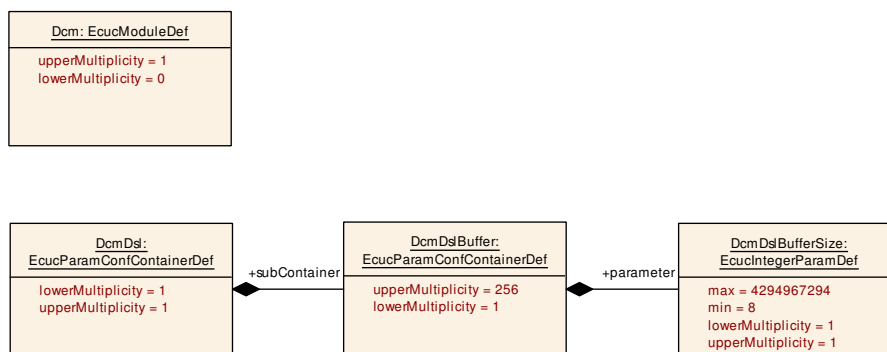


Figure 10.9: DcmDslBuffer configuration overview

10.2.4.3 DcmDslDiagResp

[ECUC_Dcm_00691] Definition of EcucParamConfContainerDef DcmDslDiagResp

Container Name	DcmDslDiagResp
Parent Container	DcmDsl
Description	This container contains the configuration of the automatic requestCorrectlyReceivedResponsePending response management in the Dcm.
Multiplicity	1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDslDiagRespMaxNumRespPend	0..1	[ECUC_Dcm_00693]
DcmDslDiagRespOnSecondDeclinedRequest	1	[ECUC_Dcm_00914]

No Included Containers

]

[ECUC_Dcm_00693] Definition of EcucIntegerParamDef DcmDslDiagRespMaxNumRespPend

Parameter Name	DcmDslDiagRespMaxNumRespPend		
Parent Container	DcmDslDiagResp		
Description	Maximum number of negative responses with response code 0x78 (requestCorrectlyReceivedResponsePending) allowed for a request. If Dcm reaches this limit, an automatic 0x10 (generalReject) final response will be trasmitted and the service processing will be cancelled. Value 0x00 means that no NRC 0x78 response will be transmitted.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00914] Definition of EcucBooleanParamDef DcmDslDiagRespOnSecondDeclinedRequest

Parameter Name	DcmDslDiagRespOnSecondDeclinedRequest		
Parent Container	DcmDslDiagResp		
Description	Defines the reaction upon a second request (ClientB) that can not be processed (e.g. due to priority assessment). TRUE: when the second request (Client B) can not be processed, it shall be answered with NRC21 BusyRepeatRequest. FALSE: when the second request (Client B) can not be processed, it shall not be responded.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

10.2.4.4 DcmDslProtocol

[ECUC_Dcm_00694] Definition of EcucParamConfContainerDef DcmDslProtocol

Container Name	DcmDslProtocol	
Parent Container	DcmDsl	
Description	This container contains the configurations of the diagnostic protocols used in Dcm.	
Multiplicity	1	
Configuration Parameters		
No Included Parameters		
Included Containers		
Container Name	Multiplicity	Dependency
DcmDslProtocolRow	1..*	This container contains the configuration of one particular diagnostic protocol used in Dcm.

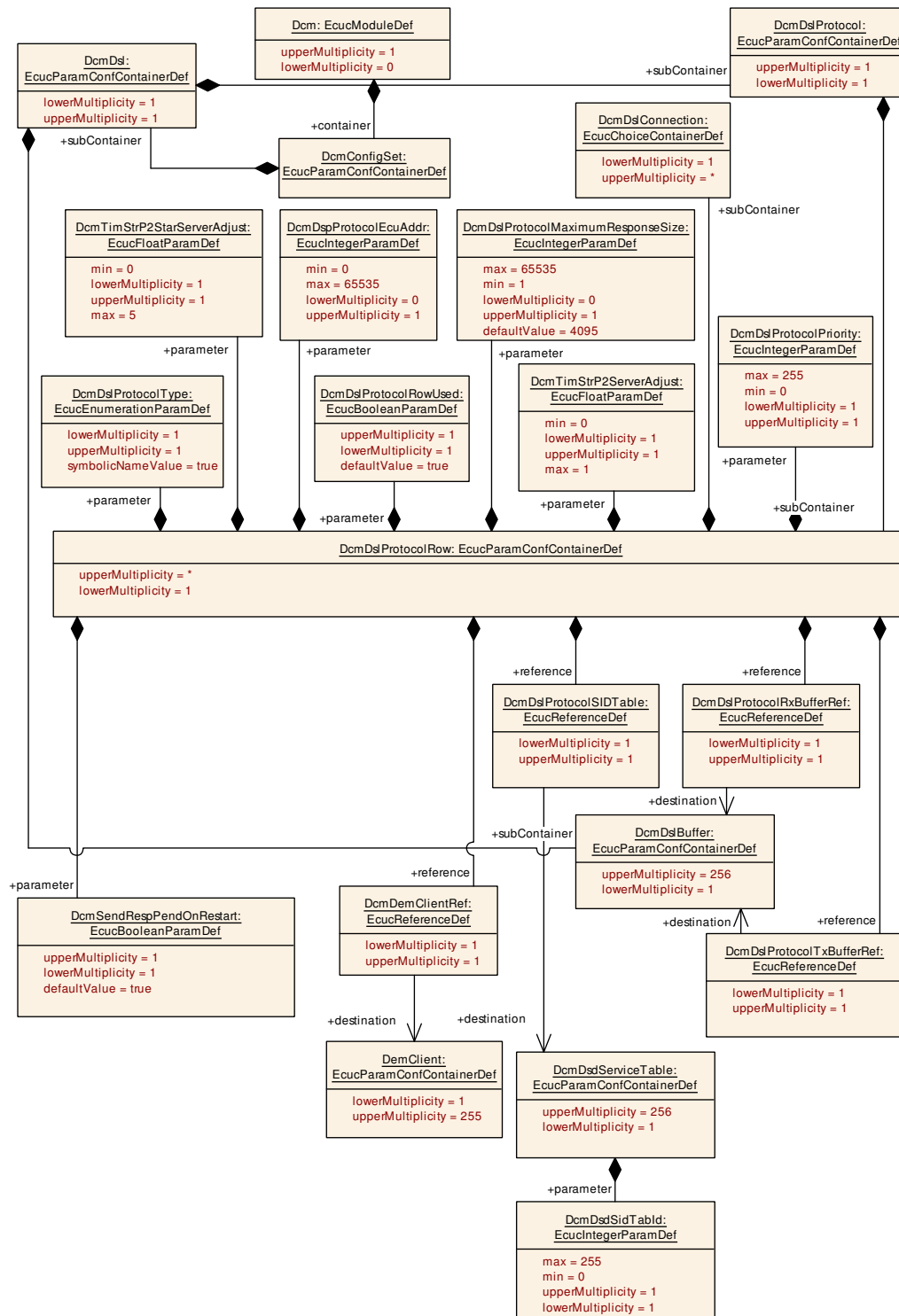


Figure 10.10: DcmDslProtocol configuration overview

10.2.4.5 DcmDslProtocolRow

[ECUC_Dcm_00695] Definition of EcucParamConfContainerDef DcmDslProtocolRow

Container Name	DcmDslProtocolRow
Parent Container	DcmDslProtocol
Description	This container contains the configuration of one particular diagnostic protocol used in Dcm.
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDslProtocolMaximumResponseSize	0..1	[ECUC_Dcm_01020]
DcmDslProtocolPriority	1	[ECUC_Dcm_00699]
DcmDslProtocolRowUsed	1	[ECUC_Dcm_01043]
DcmDslProtocolType	1	[ECUC_Dcm_01110]
DcmDspProtocolEcuAddr	0..1	[ECUC_Dcm_01081]
DcmSendRespPendOnRestart	1	[ECUC_Dcm_01114]
DcmTimStrP2ServerAdjust	1	[ECUC_Dcm_00729]
DcmTimStrP2StarServerAdjust	1	[ECUC_Dcm_00728]
DcmDemClientRef	1	[ECUC_Dcm_01083]
DcmDslProtocolRxBufferRef	1	[ECUC_Dcm_00701]
DcmDslProtocolSIDTable	1	[ECUC_Dcm_00702]
DcmDslProtocolTxBufferRef	1	[ECUC_Dcm_00704]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDslConnection	1..*	This container contains the configuration of a communication channel for one particular protocol. Note that it is allowed to communicate with multiple testers, therefore multiple connections may be configured for a protocol.

]

[ECUC_Dcm_01020] Definition of EcucIntegerParamDef DcmDslProtocolMaximumResponseSize

Parameter Name	DcmDslProtocolMaximumResponseSize	
Parent Container	DcmDslProtocolRow	
Description	This parameter is mandatory and defines the maximum length of the response message in case DcmPagedBufferEnabled == TRUE	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	1 .. 65535	
Default value	4095	
Post-Build Variant Multiplicity	false	
Post-Build Variant Value	false	





Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00699] Definition of EcucIntegerParamDef DcmDslProtocolPriority

[

Parameter Name	DcmDslProtocolPriority		
Parent Container	DcmDslProtocolRow		
Description	Protocol priority used during protocol preemption. A higher priority protocol may preempt a lower priority protocol. Lower numeric values represent higher protocol priority: 0 - Highest protocol priority 255 - Lowest protocol priority		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01043] Definition of EcucBooleanParamDef DcmDslProtocolRow Used

[

Parameter Name	DcmDslProtocolRowUsed		
Parent Container	DcmDslProtocolRow		
Description	Allows to activate or deactivate the usage of a Protocol. This parameter can be used for multi-purpose ECUs. true - protocol is available false - protocol is not available.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

]

[ECUC_Dcm_01110] Definition of EcucEnumerationParamDef DcmDslProtocolType

Parameter Name	DcmDslProtocolType		
Parent Container	DcmDslProtocolRow		
Description	The diagnostic protocol type for the DCM DSL protocol that is being configured. Implementation Type: Dcm_ProtocolType		
Multiplicity	1		
Type	EcucEnumerationParamDef (Symbolic Name generated for this parameter)		
Range	DCM_OBD_ON_CAN	OBD on CAN (ISO15765-4; ISO15031-5)	
	DCM_OBD_ON_FLEXRAY	–	
	DCM_OBD_ON_IP	–	
	DCM_PERIODICTRANS_ON_CAN	–	
	DCM_PERIODICTRANS_ON_FLEXRAY	–	
	DCM_PERIODICTRANS_ON_IP	–	
	DCM_ROE_ON_CAN	–	
	DCM_ROE_ON_FLEXRAY	–	
	DCM_ROE_ON_IP	–	
	DCM_SUPPLIER_1	Reserved for SW supplier specific	
	DCM_SUPPLIER_10	Reserved for SW supplier specific	
	DCM_SUPPLIER_11	Reserved for SW supplier specific	
	DCM_SUPPLIER_12	Reserved for SW supplier specific	
	DCM_SUPPLIER_13	Reserved for SW supplier specific	
	DCM_SUPPLIER_14	Reserved for SW supplier specific	
	DCM_SUPPLIER_15	Reserved for SW supplier specific	
	DCM_SUPPLIER_2	Reserved for SW supplier specific	
	DCM_SUPPLIER_3	Reserved for SW supplier specific	
	DCM_SUPPLIER_4	Reserved for SW supplier specific	
	DCM_SUPPLIER_5	Reserved for SW supplier specific	
	DCM_SUPPLIER_6	Reserved for SW supplier specific	
	DCM_SUPPLIER_7	Reserved for SW supplier specific	
	DCM_SUPPLIER_8	Reserved for SW supplier specific	
	DCM_SUPPLIER_9	Reserved for SW supplier specific	
	DCM_UDS_ON_CAN	UDS on CAN (ISO15765-3; ISO14229-1)	
	DCM_UDS_ON_FLEXRAY	UDS on FlexRay (Manufacturer specific; ISO14229-1)	
	DCM_UDS_ON_IP	–	
	DCM_UDS_ON_LIN	–	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01081] Definition of EcucIntegerParamDef DcmDspProtocolEcuAddr

Parameter Name	DcmDspProtocolEcuAddr		
Parent Container	DcmDslProtocolRow		
Description	Ecu source address used for diagnostic communication. This parameter is required for Generic Connections.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_01114] Definition of EcucBooleanParamDef DcmSendRespPendOnRestart

Parameter Name	DcmSendRespPendOnRestart		
Parent Container	DcmDslProtocolRow		
Description	If set to TRUE, the Dcm will send a NRC 0x78 before a transition to bootloader or performing an ECU reset. If set to False, no 0x78 is send in this case.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_00729] Definition of EcucFloatParamDef DcmTimStrP2ServerAdjust

Parameter Name	DcmTimStrP2ServerAdjust		
Parent Container	DcmDslProtocolRow		
Description	This parameter is used to guarantee that the diagnostic response is available on the bus before reaching P2 by adjusting the current DcmDspSessionP2ServerMax. This parameter mainly represents the software architecture dependent communication delay between the time the transmission is initiated by DCM and the time when the message is actually transmitted to the bus. The parameter value is defined in seconds and must be a multiple of DcmTaskTime.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 1]		





Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00728] Definition of EcucFloatParamDef DcmTimStrP2StarServer Adjust [

Parameter Name	DcmTimStrP2StarServerAdjust		
Parent Container	DcmDslProtocolRow		
Description	This parameter is used to guarantee that the diagnostic response is available on the bus before reaching P2Star by adjusting the current DcmDspSessionP2StarServerMax. This parameter mainly represents the software architecture dependent communication delay between the time the transmission is initiated by DCM and the time when the message is actually transmitted to the bus. The parameter value is defined in seconds and must be a multiple of DcmTaskTime.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 5]		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01083] Definition of EcucReferenceDef DcmDemClientRef [

Parameter Name	DcmDemClientRef		
Parent Container	DcmDslProtocolRow		
Description	Reference to DemClient in Dem configuration. Used by the Dem to distinguish different client calls.		
Multiplicity	1		
Type	Reference to DemClient		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	





Dependency	

]

[ECUC_Dcm_00701] Definition of EcucReferenceDef DcmDslProtocolRxBuffer Ref [

Parameter Name	DcmDslProtocolRxBufferRef		
Parent Container	DcmDslProtocolRow		
Description	Reference to a configured diagnostic buffer that is used for diagnostic request reception for the protocol.		
Multiplicity	1		
Type	Reference to DcmDslBuffer		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00702] Definition of EcucReferenceDef DcmDslProtocolSIDTable [

Parameter Name	DcmDslProtocolSIDTable		
Parent Container	DcmDslProtocolRow		
Description	Reference to a service table that is used for diagnostic request processing for this protocol.		
Multiplicity	1		
Type	Reference to DcmDsdServiceTable		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00704] Definition of EcucReferenceDef DcmDslProtocolTxBuffer Ref [

Parameter Name	DcmDslProtocolTxBufferRef		
Parent Container	DcmDslProtocolRow		
Description	Reference to a configured diagnostic buffer that is used for diagnostic response transmission for the protocol.		
Multiplicity	1		
Type	Reference to DcmDslBuffer		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD





	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

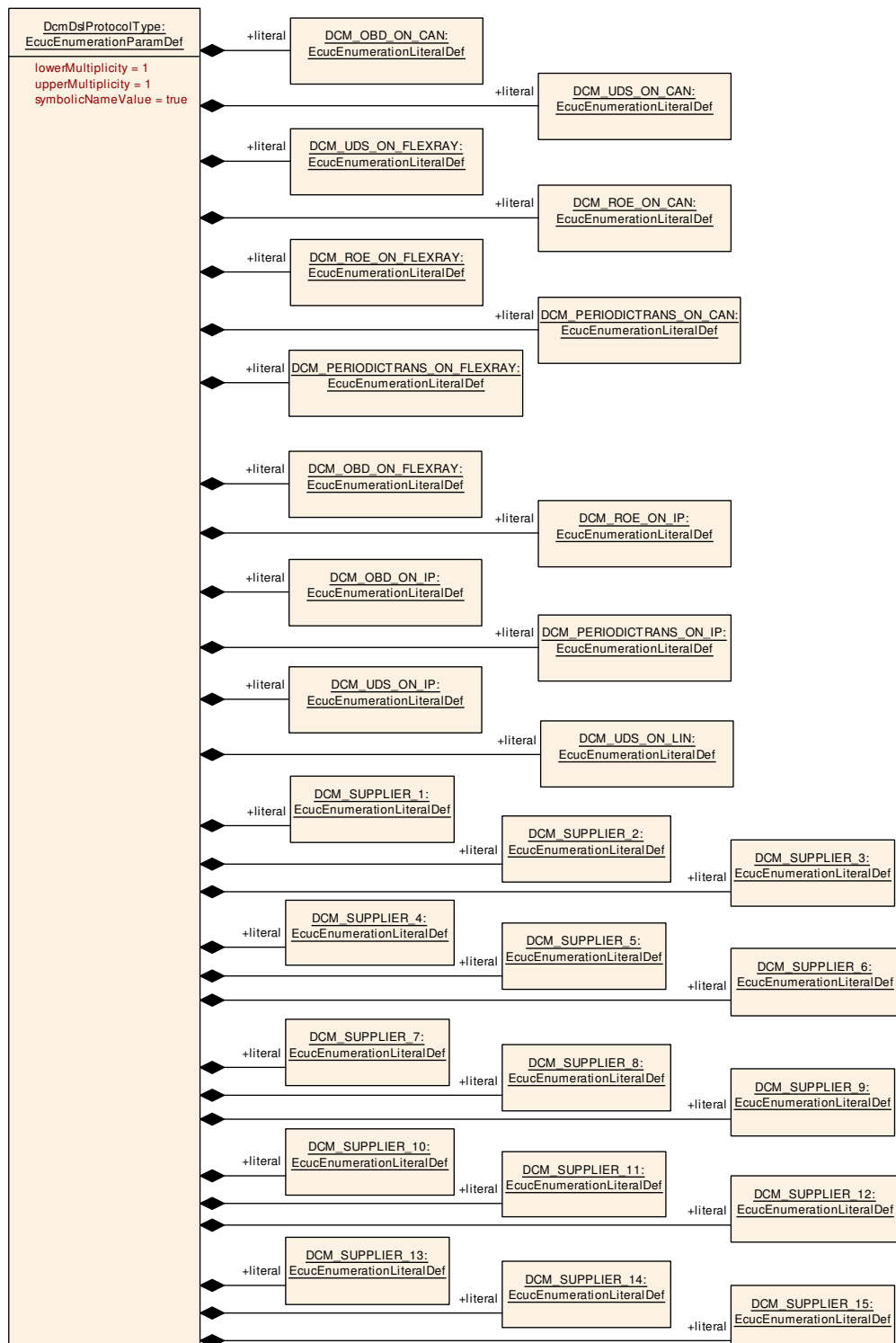


Figure 10.11: DcmDslProtocolType configuration overview

10.2.4.6 DcmDslConnection

[ECUC_Dcm_00705] Definition of EcucChoiceContainerDef DcmDslConnection

Choice Container Name	DcmDslConnection
Parent Container	DcmDslProtocolRow
Description	This container contains the configuration of a communication channel for one particular protocol. Note that it is allowed to communicate with multiple testers, therefore multiple connections may be configured for a protocol.
Multiplicity	1..*

No Included Parameters

Container Choices		
Container Name	Multiplicity	Dependency
DcmDslMainConnection	0..1	This container contains the configuration for a main connection of a diagnostic protocol. Additionally it may contain references to ROE and Periodic connections if the protocol type or protocol transmission type needs them.
DcmDslPeriodicTransmission	0..1	This container contains the configuration of a periodic transmission connection.

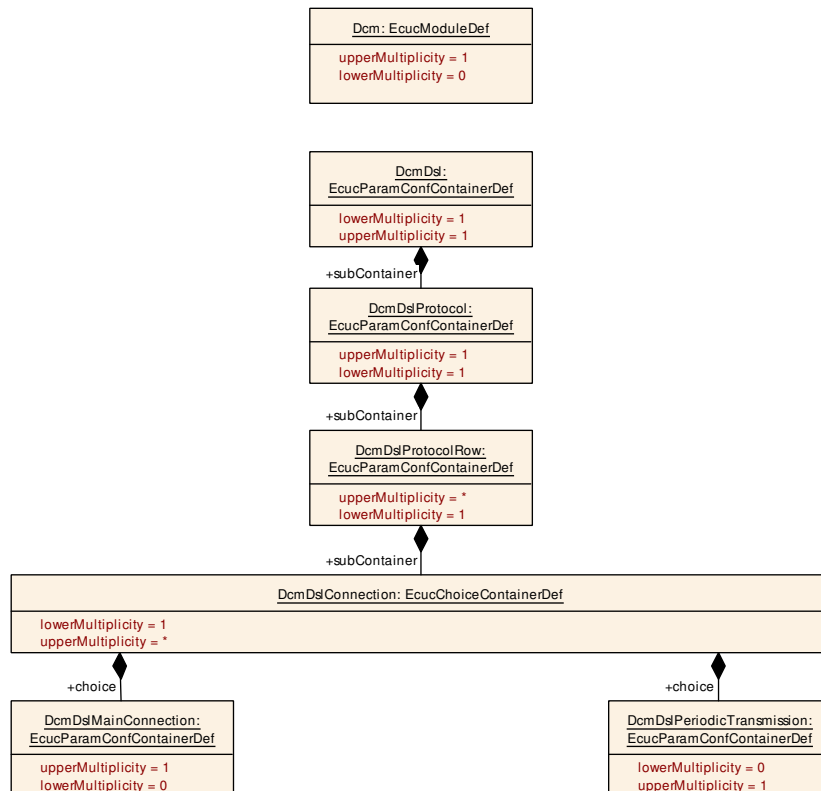


Figure 10.12: DcmDslConnectionTable configuration overview

10.2.4.7 DcmDslMainConnection

[ECUC_Dcm_00706] Definition of EcucParamConfContainerDef DcmDslMainConnection

Container Name	DcmDslMainConnection
Parent Container	DcmDslConnection
Description	This container contains the configuration for a main connection of a diagnostic protocol. Additionally it may contain references to ROE and Periodic connections if the protocol type or protocol transmission type needs them.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDslProtocolRxConnectionId	1	[ECUC_Dcm_00826]
DcmDslProtocolRxTesterSourceAddr	0..1	[ECUC_Dcm_01115]
DcmDslPeriodicTransmissionConRef	0..1	[ECUC_Dcm_00707]
DcmDslProtocolComMChannelRef	1	[ECUC_Dcm_00952]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDslProtocolRx	1..*	This container contains the configuration parameters of a reception channel in a diagnostic connection. The PDU referenced by this reception channel can consume meta data items of type SOURCE_ADDRESS_16 and TARGET_ADDRESS_16.
DcmDslProtocolTx	0..1	This container contains the configuration parameters of a transmission channel in a diagnostic connection. The PDU referenced by this transmission channel can produce meta data items of type TARGET_ADDRESS_16 and SOURCE_ADDRESS_16.

[ECUC_Dcm_00826] Definition of EcucIntegerParamDef DcmDslProtocolRxConnectionId

Parameter Name	DcmDslProtocolRxConnectionId		
Parent Container	DcmDslMainConnection		
Description	Unique identifier of the tester which uses this connection for diagnostic communication.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants





	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01115] Definition of EcucIntegerParamDef DcmDslProtocolRx TesterSourceAddr

Parameter Name	DcmDslProtocolRxTesterSourceAddr		
Parent Container	DcmDslMainConnection		
Description	Tester source address uniquely describes a client and will be used e.g within the jump to Bootloader interfaces. This parameter is not required for generic connections (Dcm Pdus with MetaDataLength >= 1).		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00707] Definition of EcucReferenceDef DcmDslPeriodicTransmissionConRef

Parameter Name	DcmDslPeriodicTransmissionConRef		
Parent Container	DcmDslMainConnection		
Description	Reference to a periodic transmission connection which is used for the processing of periodic transmission events.		
Multiplicity	0..1		
Type	Reference to DcmDslPeriodicTransmission		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	





Dependency	

」

[ECUC_Dcm_00952] Definition of EcucReferenceDef DcmDslProtocolComMChannelRef 「

Parameter Name	DcmDslProtocolComMChannelRef		
Parent Container	DcmDslMainConnection		
Description	Reference to the ComMChannel on which the DcmDslProtocolRxPdu is received and the DcmDslProtocolTxPdu is transmitted.		
Multiplicity	1		
Type	Symbolic name reference to ComMChannel		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

」

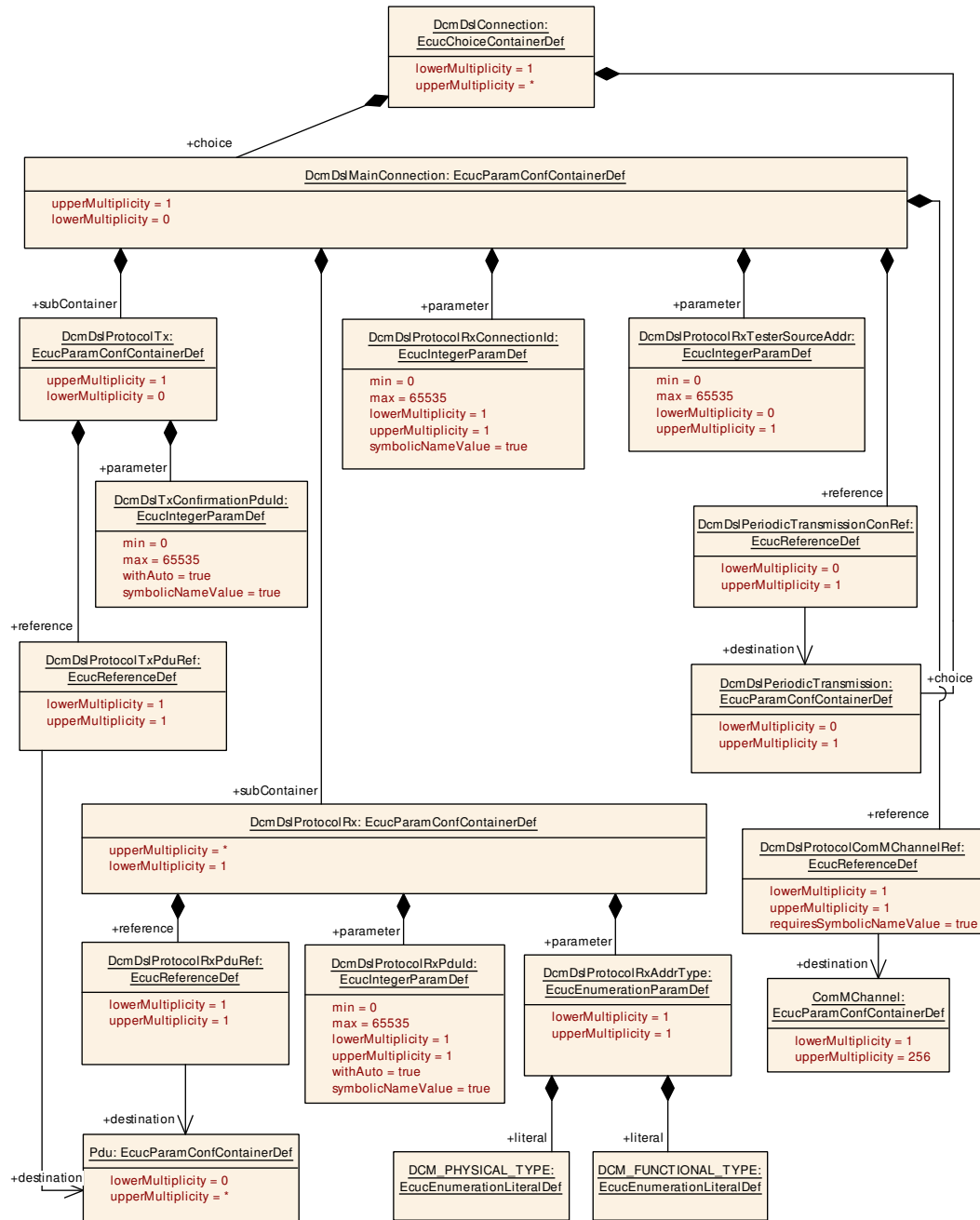


Figure 10.13: DcmDslMainConnection configuration overview

10.2.4.8 DcmDslProtocolRx

[ECUC_Dcm_00709] Definition of EcucParamConfContainerDef DcmDslProtocolRx

Container Name	DcmDslProtocolRx
Parent Container	DcmDslMainConnection
Description	This container contains the configuration parameters of a reception channel in a diagnostic connection. The PDU referenced by this reception channel can consume meta data items of type SOURCE_ADDRESS_16 and TARGET_ADDRESS_16.
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDslProtocolRxAddrType	1	[ECUC_Dcm_00710]
DcmDslProtocolRxPduId	1	[ECUC_Dcm_00687]
DcmDslProtocolRxPduRef	1	[ECUC_Dcm_00770]

No Included Containers

]

[ECUC_Dcm_00710] Definition of EcucEnumerationParamDef DcmDslProtocolRxAddrType

Parameter Name	DcmDslProtocolRxAddrType		
Parent Container	DcmDslProtocolRx		
Description	Selects the addressing type of the reception channel. Physical addressing is used for 1:1 communication, functional addressing is used for 1:N communication. For details refer to ISO 14229-1.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DCM_FUNCTIONAL_TYPE	FUNCTIONAL = 1 to n communication	
	DCM_PHYSICAL_TYPE	PHYSICAL = 1 to 1 communications using physical addressing	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_00687] Definition of EcucIntegerParamDef DcmDslProtocolRxPduId

Parameter Name	DcmDslProtocolRxPduId		
Parent Container	DcmDslProtocolRx		
Description	Identifier of the PDU that is used for this reception channel.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		





Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency	withAuto = true		

]

[ECUC_Dcm_00770] Definition of EcucReferenceDef DcmDslProtocolRxPduRef

[

Parameter Name	DcmDslProtocolRxPduRef		
Parent Container	DcmDslProtocolRx		
Description	Reference to a Pdu in EcuC that is used for this reception channel.		
Multiplicity	1		
Type	Reference to Pdu		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

10.2.4.9 DcmDslProtocolTx

[ECUC_Dcm_00711] Definition of EcucParamConfContainerDef DcmDslProtocolTx

[

Container Name	DcmDslProtocolTx		
Parent Container	DcmDslMainConnection		
Description	This container contains the configuration parameters of a transmission channel in a diagnostic connection. The PDU referenced by this transmission channel can produce meta data items of type TARGET_ADDRESS_16 and SOURCE_ADDRESS_16.		
Multiplicity	0..1		
Configuration Parameters			

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDslTxConfirmationPduId	1	[ECUC_Dcm_00864]
DcmDslProtocolTxPduRef	1	[ECUC_Dcm_00772]

No Included Containers

]

[ECUC_Dcm_00864] Definition of EcucIntegerParamDef DcmDslTxConfirmationPduId

Parameter Name	DcmDslTxConfirmationPduId		
Parent Container	DcmDslProtocolTx		
Description	Identifier of the PDU that is used by the lower level module for transmission confirmation of responses on this channel.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency	withAuto = true		

[ECUC_Dcm_00772] Definition of EcucReferenceDef DcmDslProtocolTxPduRef

Parameter Name	DcmDslProtocolTxPduRef		
Parent Container	DcmDslProtocolTx		
Description	Reference to a Pdu in EcuC that is used for this transmission channel.		
Multiplicity	1		
Type	Reference to Pdu		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

10.2.4.10 DcmDslPeriodicTransmission

[ECUC_Dcm_00741] Definition of EcucParamConfContainerDef DcmDslPeriodicTransmission

Container Name	DcmDslPeriodicTransmission
Parent Container	DcmDslConnection
Description	This container contains the configuration of a periodic transmission connection.
Multiplicity	0..1
Configuration Parameters	
No Included Parameters	

Included Containers		
Container Name	Multiplicity	Dependency
DcmDslPeriodicConnection	0..255	This container contains the configuration of a transmission channel for a periodic transmission connection. The PDU referenced by this transmission channel can produce meta data items of type TARGET_ADDRESS_16 and SOURCE_ADDRESS_16.

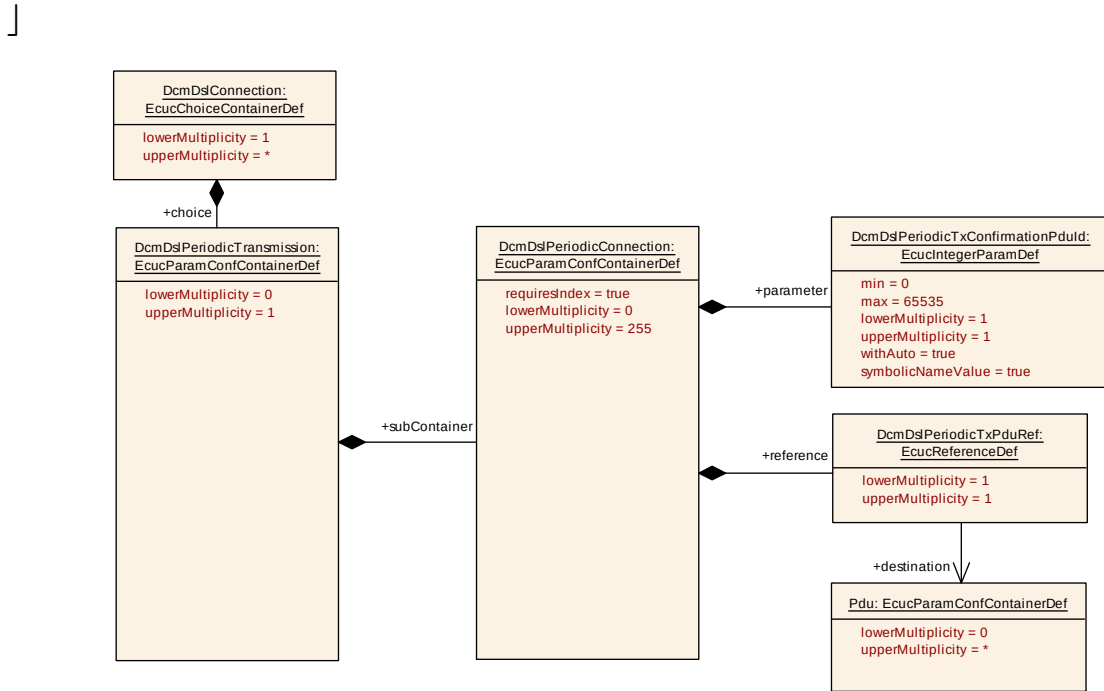


Figure 10.14: DcmDslPeriodicTransmission configuration overview

10.2.4.11 DcmDslPeriodicConnection

[ECUC_Dcm_00897] Definition of EcucParamConfContainerDef DcmDslPeriodicConnection

Container Name	DcmDslPeriodicConnection
Parent Container	DcmDslPeriodicTransmission
Description	This container contains the configuration of a transmission channel for a periodic transmission connection. The PDU referenced by this transmission channel can produce meta data items of type TARGET_ADDRESS_16 and SOURCE_ADDRESS_16. Attributes: requiresIndex=true
Multiplicity	0..255
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDslPeriodicTxConfirmationPdu	1	[ECUC_Dcm_00862]
DcmDslPeriodicTxPduRef	1	[ECUC_Dcm_00742]

No Included Containers

]

[ECUC_Dcm_00862] Definition of EcucIntegerParamDef DcmDslPeriodicTxConfirmationPduld [

Parameter Name	DcmDslPeriodicTxConfirmationPduld		
Parent Container	DcmDslPeriodicConnection		
Description	Identifier of the PDU that is used by the lower level module for transmission confirmation of responses on this channel.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency	withAuto = true		

]

[ECUC_Dcm_00742] Definition of EcucReferenceDef DcmDslPeriodicTxPduRef [

Parameter Name	DcmDslPeriodicTxPduRef		
Parent Container	DcmDslPeriodicConnection		
Description	Reference to a Pdu in EcuC that is used for this periodic transmission channel.		
Multiplicity	1		
Type	Reference to Pdu		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

10.2.5 DcmDsp

[ECUC_Dcm_00712] Definition of EcucParamConfContainerDef DcmDsp [

Container Name	DcmDsp
Parent Container	DcmConfigSet
Description	These parameters apply to Diagnostic Service Processing. There will always be one set of these parameters per Dcm. Please note: Although the multiplicity is set to 0..1. It can be expected that this container exists in any valid DCM configuration.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDataDefaultEndianness	1	[ECUC_Dcm_00987]
DcmDspDDIDCheckPerSourceDID	0..1	[ECUC_Dcm_00966]
DcmDspEnableObdMirror	1	[ECUC_Dcm_01061]
DcmDspMaxDidToRead	0..1	[ECUC_Dcm_00638]
DcmDspMaxPeriodicDidToRead	0..1	[ECUC_Dcm_00956]
DcmDspPowerDownTime	0..1	[ECUC_Dcm_00818]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspAuthentication	0..1	This container contains the configuration for the UDS service Authentication (0x29).
DcmDspClearDTC	0..1	This container contains the configuration for the Clear DTC service.
DcmDspComControl	0..1	Provides the configuration of the CommunicationControl mechanism.
DcmDspCommonAuthorization	0..*	This container contains the configuration (parameters) for the common Authorization being equal for several services / sub-services.
DcmDspControlDTCSetting	0..1	Provide the configuration of the ControlDTCSetting mechanism.
DcmDspData	0..*	This container contains the configuration (parameters) of a Data belonging to a DID
DcmDspDataInfo	0..*	This container contains the configuration (parameters) of one Data.
DcmDspDid	0..*	This container contains the configuration (parameters) of the DID.
DcmDspDidInfo	0..*	This container contains the configuration (parameters) of the DID's Info
DcmDspDidRange	0..*	This container defines the DID Range
DcmDspEcuReset	0..1	This container contains the configuration for DcmDspEcuReset service
DcmDspMemory	0..1	This container contains the configuration of the memory access.
DcmDspMemoryTransfer	0..1	This container contains the configuration of the memory transfer.
DcmDspPeriodicTransmission	0..1	This container contains the configuration (parameters) for Periodic Transmission Scheduler.
DcmDspPid	0..*	This container defines the availability of a PID to the DCM.
DcmDspReadDTCInformation	0..1	Provides further configuration for the UDS service Read DTCInformation processing.
DcmDspRequestControl	0..*	This container contains the configuration (parameters) of the "Request control of on-board system, test or component" service (Service \$08). The DCM will request the control using an R-Port requiring a PortInterface RequestControlServices_{Tid}. The R-Port is named RequestControlServices_{Tid} where {Tid} is the name of the container DcmDspRequestControl.
DcmDspRequestFileTransfer	0..1	This container contains the configuration for RequestFile Transfer. This container only exists if RequestFileTransfer is configured.
DcmDspRoe	0..1	Provide the configuration of the ResponseOnEvent mechanism.
DcmDspRoutine	0..*	This container contains the configuration (parameters) for Routines
DcmDspSecurity	1	This container contains the configuration (DSP parameter) for security level configuration (per security level) Description This container contains Rows of DcmDspSecurityRow





Included Containers		
Container Name	Multiplicity	Dependency
DcmDspSession	1	Parent container holding single rows to configure particular sessions
DcmDspVehInfo	0..*	This container contains the configuration (parameters) for one single VehicleInfoType of service \$09

]

[ECUC_Dcm_00987] Definition of EcucEnumerationParamDef DcmDspDataDefaultEndianness [

Parameter Name	DcmDspDataDefaultEndianness		
Parent Container	DcmDsp		
Description	Defines the default endianness belonging to a DID, RID or PID if the corresponding data does not define an endianness.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	BIG_ENDIAN	Most significant byte shall be stored at the lowest address.	
	LITTLE_ENDIAN	Most significant byte shall be stored at the highest address	
	OPAQUE	Opaque data endianness	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_00966] Definition of EcucBooleanParamDef DcmDspDDDIDcheckPerSourceDID [

Parameter Name	DcmDspDDDIDcheckPerSourceDID		
Parent Container	DcmDsp		
Description	Defines the check for session, security and mode dependencies per source DIDs with a ReadDataByIdentifier (0x22). true: Dcm module shall check the session, security and mode dependencies per source DIDs with a ReadDataByIdentifier (0x22) with DID in the range 0xF200 to 0xF3FF false: Dcm module shall not check the session, security and mode dependencies per source DIDs with a ReadDataByIdentifier (0x22) with DID in the range 0xF200 to 0xF3FF		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME





	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01061] Definition of EcucBooleanParamDef DcmDspEnableObdMirror

Parameter Name	DcmDspEnableObdMirror		
Parent Container	DcmDsp		
Description	DcmDspEnableObdMirror defines whether a DID inside the OBD range (F400-F4FF) and the OBD InfoType range (F800-F8FF) shall get the DID value as defined for OBD on reception of the UDS Service ReadDataByIdentifier (0x22), or not. It also defines whether a RID inside the OBD range (E000-E0FF) shall handle the RID as defined for OBD on reception of the UDS Service RoutineControl (0x31), or not.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00638] Definition of EcucIntegerParamDef DcmDspMaxDidToRead

Parameter Name	DcmDspMaxDidToRead		
Parent Container	DcmDsp		
Description	Indicates the maximum number of allowed DIDs in a single 'ReadDataByIdentifier' request for non-OBd DID requests (at least one DID outside the OBD range of 0x F400-0xF8FF).		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	—		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD





	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00956] Definition of EcucIntegerParamDef DcmDspMaxPeriodicDidToRead

Parameter Name	DcmDspMaxPeriodicDidToRead		
Parent Container	DcmDsp		
Description	Indicates the maximum allowed periodicDIDs which can be read in a single "ReadData ByPeriodicIdentifier" request.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00818] Definition of EcucIntegerParamDef DcmDspPowerDownTime

Parameter Name	DcmDspPowerDownTime		
Parent Container	DcmDsp		
Description	This parameter indicates to the client the minimum time of the stand-by sequence the server will remain in the power-down sequence. The resolution of this parameter is one second per count. The following values are valid: 00 - FE hex: 0 - 254 s powerDownTime; FF hex: indicates a failure or time not available. This value needs to be defined by the integrator according to the ECU capabilities. This parameter has to be available if the service EcuReset, sub-service enableRapidPower ShutDown is configured.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		



△

Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

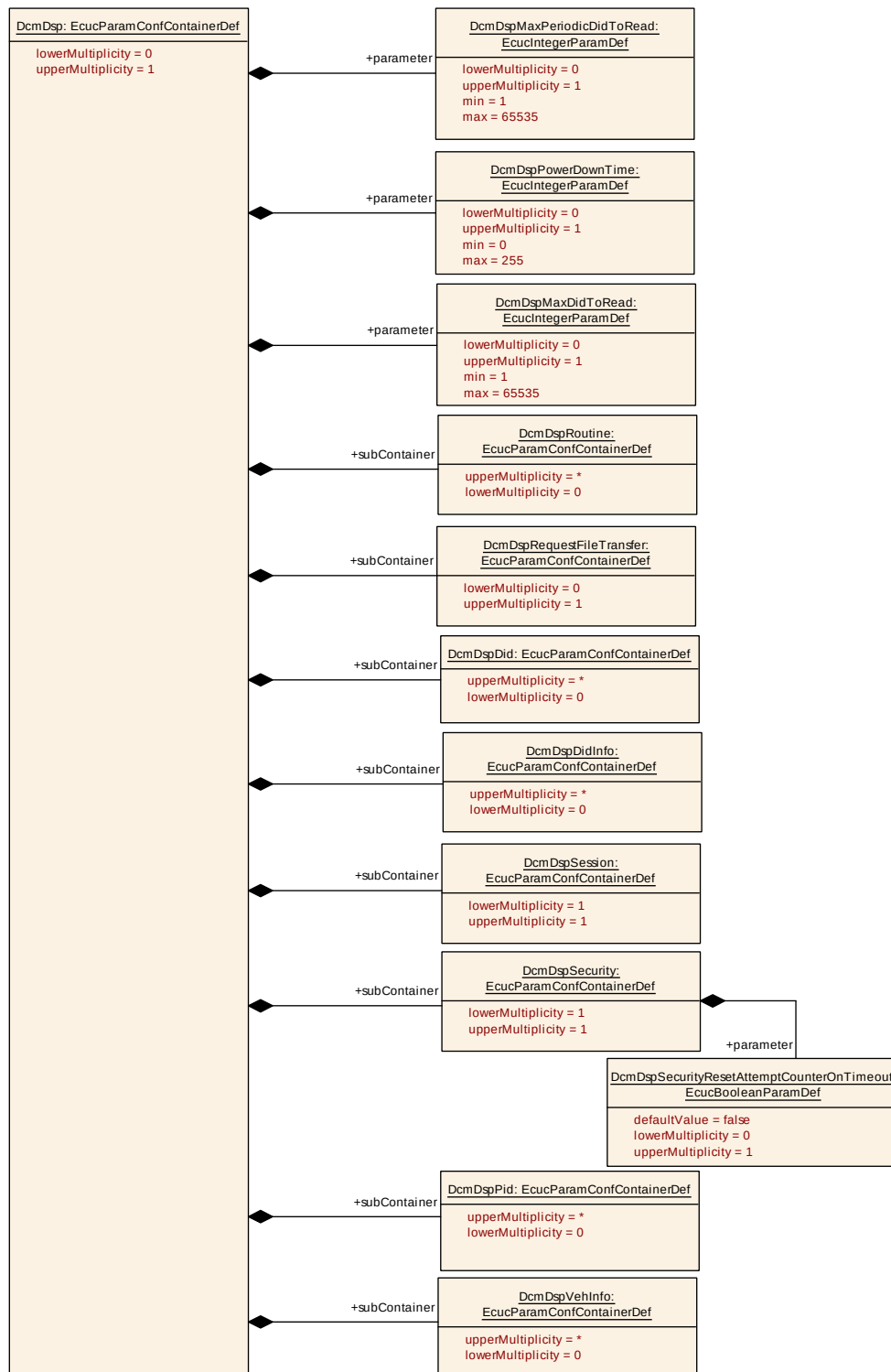


Figure 10.15: DcmDsp configuration overview Part1

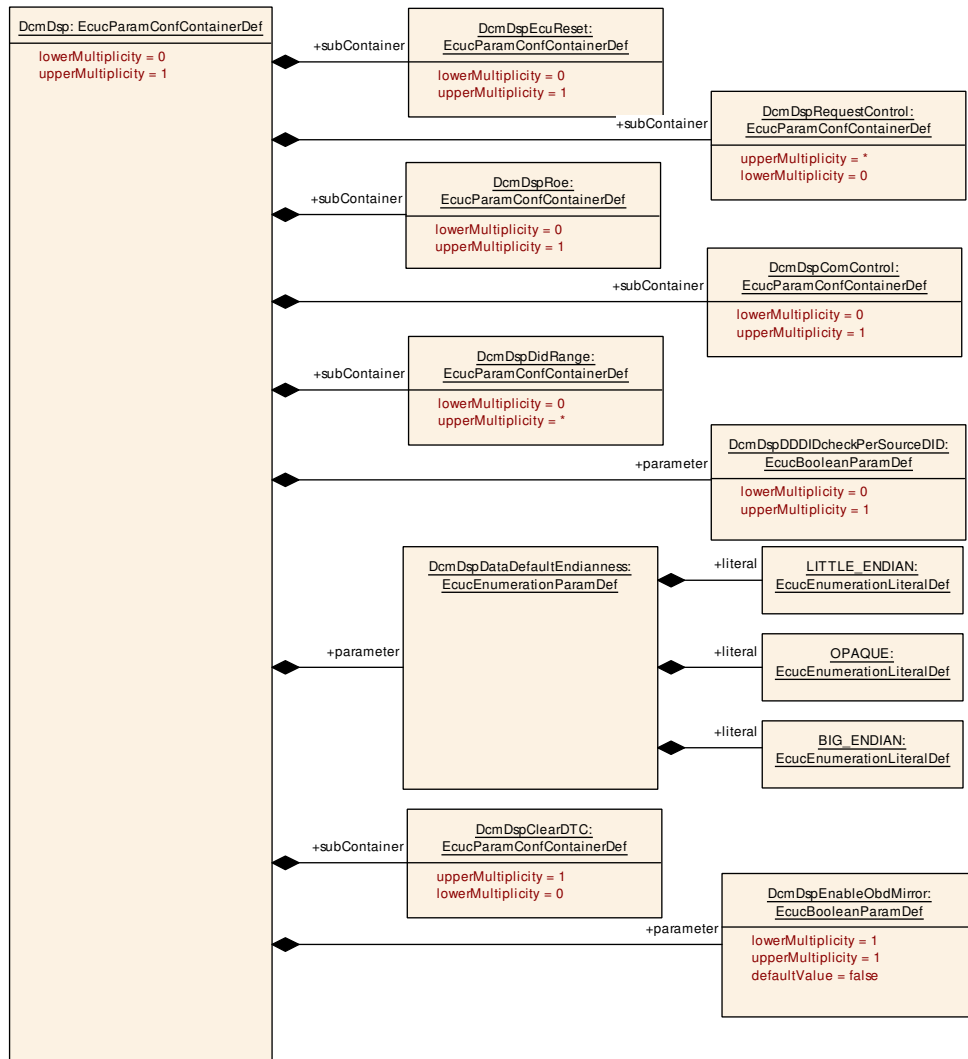


Figure 10.16: DcmDsp configuration overview Part2

10.2.5.1 DcmDspReadDTCInformation

[ECUC_Dcm_01147] Definition of EcucParamConfContainerDef DcmDspReadDTCInformation [

Container Name	DcmDspReadDTCInformation
Parent Container	DcmDsp
Description	Provides further configuration for the UDS service ReadDTCInformation processing.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspReadDTCInformationSupportedObdUdsDtcSeparation	0..1	[ECUC_Dcm_01214]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspReadDTCInformationUserDefinedFaultMemory	0..255	This container contains the configuration for user defined fault memories in DcmDspReadDTCInformation.

]

[ECUC_Dcm_01214] Definition of EcucBooleanParamDef DcmDspReadDTCInformationSupportedObdUdsDtcSeparation [

Parameter Name	DcmDspReadDTCInformationSupportedObdUdsDtcSeparation		
Parent Container	DcmDspReadDTCInformation		
Description	If this parameter is set to true, the OBD UDS DTC separation is enabled. If this parameter is not configured or set to false, the OBD UDS DTC separation is disabled.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

10.2.5.2 DcmDspReadDTCInformationUserDefinedFaultMemory

[ECUC_Dcm_01148] Definition of EcucParamConfContainerDef DcmDspReadDTCInformationUserDefinedFaultMemory [

Container Name	DcmDspReadDTCInformationUserDefinedFaultMemory
Parent Container	DcmDspReadDTCInformation
Description	This container contains the configuration for user defined fault memories in DcmDspReadDTCInformation.
Multiplicity	0..255
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspReadDTCInformationUserDefinedFaultMemoryId	1	[ECUC_Dcm_01149]
DcmDspReadDTCInformationUserDefinedFaultMemoryRoleRef	0..32	[ECUC_Dcm_01150]

No Included Containers

]

[ECUC_Dcm_01149] Definition of EcucIntegerParamDef DcmDspReadDTCInformationUserDefinedFaultMemoryId [

Parameter Name	DcmDspReadDTCInformationUserDefinedFaultMemoryId		
Parent Container	DcmDspReadDTCInformationUserDefinedFaultMemory		
Description	Identifier used by external tester to identify the User defined event Memory.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01150] Definition of EcucReferenceDef DcmDspReadDTCInformationUserDefinedFaultMemoryRoleRef [

Parameter Name	DcmDspReadDTCInformationUserDefinedFaultMemoryRoleRef		
Parent Container	DcmDspReadDTCInformationUserDefinedFaultMemory		
Description	Reference to DcmDspAuthenticationRow that defines a role in that this user defined memory can be accessed.		
Multiplicity	0..32		
Type	Reference to DcmDspAuthenticationRow		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

]

10.2.5.3 DcmDspAuthentication

[ECUC_Dcm_01151] Definition of EcucParamConfContainerDef DcmDspAuthentication [

Container Name	DcmDspAuthentication
Parent Container	DcmDsp
Description	This container contains the configuration for the UDS service Authentication (0x29).
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspAuthenticationDefaultSessionTimeOut	0..1	[ECUC_Dcm_01161]
DcmDspAuthenticationGeneralNRC	0..1	[ECUC_Dcm_01159]
DcmDspAuthenticationMileageOffsetLimit	0..1	[ECUC_Dcm_01277]
DcmDspAuthenticationRoleSize	1	[ECUC_Dcm_01152]
DcmDspAuthenticationWhiteListDIDMaxSize	0..1	[ECUC_Dcm_01155]
DcmDspAuthenticationWhiteListMemorySelectionMaxSize	0..1	[ECUC_Dcm_01157]
DcmDspAuthenticationWhiteListRIDMaxSize	0..1	[ECUC_Dcm_01156]
DcmDspAuthenticationWhiteListServicesMaxSize	0..1	[ECUC_Dcm_01154]
DcmDspAuthenticationDeauthenticatedRoleRef	1..32	[ECUC_Dcm_01153]
DcmDspAuthenticationGeneralNRCModeRuleRef	0..1	[ECUC_Dcm_01158]
DcmDspAuthenticationMileageInformation	0..1	[ECUC_Dcm_01276]
DcmDspAuthenticationPersistStateModeRuleRef	0..1	[ECUC_Dcm_01160]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspAuthenticationConnection	0..*	This container contains the authentication configuration use for a dsl connection.
DcmDspAuthenticationRow	0..31	This container contains the definition of a single role. Each role is a bit in the authentication role bitfield.
DcmDspAuthenticationTransmit Certificate	0..*	This container contains the authentication configuration for the subfunction trasnmitCertificate.

]

[[ECUC_Dcm_01161](#)] Definition of EcucFloatParamDef DcmDspAuthenticationDefaultSessionTimeOut [

Parameter Name	DcmDspAuthenticationDefaultSessionTimeOut		
Parent Container	DcmDspAuthentication		
Description	Defines the number of seconds after which the Dcm makes a transition to deauthenticated state, in case of no active communication.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[0 .. INF[		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	





Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01159] Definition of EcucIntegerParamDef DcmDspAuthenticationGeneralNRC

Parameter Name	DcmDspAuthenticationGeneralNRC		
Parent Container	DcmDspAuthentication		
Description	Defines the NRC that shall be send as replacement of all ISO 14229-1 defined NRCs in case of invalid certificate or content.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01277] Definition of EcucIntegerParamDef DcmDspAuthenticationMileageOffsetLimit

Parameter Name	DcmDspAuthenticationMileageOffsetLimit		
Parent Container	DcmDspAuthentication		
Description	Defines the maximum distance that can be travelled in authenticated state.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01152] Definition of EcucIntegerParamDef DcmDspAuthenticationRoleSize

Parameter Name	DcmDspAuthenticationRoleSize		
Parent Container	DcmDspAuthentication		
Description	Defines the size in bytes for the role element within a certificate.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01155] Definition of EcucIntegerParamDef DcmDspAuthenticationWhiteListDIDMaxSize

Parameter Name	DcmDspAuthenticationWhiteListDIDMaxSize		
Parent Container	DcmDspAuthentication		
Description	Defines the maximum size in bytes for the white list element within a certificate.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01157] Definition of EcucIntegerParamDef DcmDspAuthenticationWhiteListMemorySelectionMaxSize

Parameter Name	DcmDspAuthenticationWhiteListMemorySelectionMaxSize		
Parent Container	DcmDspAuthentication		
Description	Defines the maximum size in bytes for the white list element within a certificate.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		





Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01156] Definition of EcucIntegerParamDef DcmDspAuthenticationWhiteListRIDMaxSize [

Parameter Name	DcmDspAuthenticationWhiteListRIDMaxSize		
Parent Container	DcmDspAuthentication		
Description	Defines the maximum size in bytes for the white list element within a certificate.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01154] Definition of EcucIntegerParamDef DcmDspAuthenticationWhiteListServicesMaxSize [

Parameter Name	DcmDspAuthenticationWhiteListServicesMaxSize		
Parent Container	DcmDspAuthentication		
Description	Defines the maximum size in bytes for the white list element within a certificate.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01153] Definition of EcucReferenceDef DcmDspAuthenticationDeauthenticatedRoleRef [

Parameter Name	DcmDspAuthenticationDeauthenticatedRoleRef		
Parent Container	DcmDspAuthentication		
Description	Reference to DcmDspAuthenticationRow that defines a role in that is used as deauthenticated role.		
Multiplicity	1..32		
Type	Reference to DcmDspAuthenticationRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01158] Definition of EcucReferenceDef DcmDspAuthenticationGeneralNRCModeRuleRef [

Parameter Name	DcmDspAuthenticationGeneralNRCModeRuleRef		
Parent Container	DcmDspAuthentication		
Description	Mode rule that defines if the general NRC shall be send for all failures due to invalid certificate or content.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01276] Definition of EcucReferenceDef DcmDspAuthenticationMileageInformation [

Parameter Name	DcmDspAuthenticationMileageInformation		
Parent Container	DcmDspAuthentication		
Description	Input variable for the distance information (odometer value), which is used to supervise the distance offset limit for the authenticated state. The variable is a 4 byte unsigned integer		
Multiplicity	0..1		
Type	Reference to DcmDspData		
Post-Build Variant Multiplicity	false		





Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01160] Definition of EcucReferenceDef DcmDspAuthenticationPersistStateModeRuleRef [

Parameter Name	DcmDspAuthenticationPersistStateModeRuleRef		
Parent Container	DcmDspAuthentication		
Description	References a mode rule that defines if the authentication state shall be persisted in non-volatile memory.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.3.1 DcmDspAuthenticationTransmitCertificate

[ECUC_Dcm_01241] Definition of EcucParamConfContainerDef DcmDspAuthenticationTransmitCertificate [

Container Name	DcmDspAuthenticationTransmitCertificate		
Parent Container	DcmDspAuthentication		
Description	This container contains the authentication configuration for the subfunction transmit Certificate.		
Multiplicity	0..*		
Post-Build Variant Multiplicity	false		
Configuration Parameters			

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspAuthenticationTransmitCertificateEvaluationId	1	[ECUC_Dcm_01242]
DcmDspAuthenticationTransmitCertificateFunction	0..1	[ECUC_Dcm_01244]
DcmDspAuthenticationTransmitCertificateRef	1	[ECUC_Dcm_01243]

No Included Containers

]

[ECUC_Dcm_01242] Definition of EcucIntegerParamDef DcmDspAuthenticationTransmitCertificateEvaluationId [

Parameter Name	DcmDspAuthenticationTransmitCertificateEvaluationId		
Parent Container	DcmDspAuthenticationTransmitCertificate		
Description	Certificate evaluation id.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01244] Definition of EcucEnumerationParamDef DcmDspAuthenticationTransmitCertificateFunction [

Parameter Name	DcmDspAuthenticationTransmitCertificateFunction		
Parent Container	DcmDspAuthenticationTransmitCertificate		
Description	Provides a fixed semantics to this certificate that will lead to predefined actions in the Dcm.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	SECURE_CODING	This certificate is use write authenticated data.	
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01243] Definition of EcucReferenceDef DcmDspAuthenticationTransmitCertificateRef

Parameter Name	DcmDspAuthenticationTransmitCertificateRef		
Parent Container	DcmDspAuthenticationTransmitCertificate		
Description	Reference to a KeyMCertificate used to handle the certificate.		
Multiplicity	1		
Type	Symbolic name reference to KeyMCertificate		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.4 DcmDspAuthenticationRow

[ECUC_Dcm_01211] Definition of EcucParamConfContainerDef DcmDspAuthenticationRow

Container Name	DcmDspAuthenticationRow		
Parent Container	DcmDspAuthentication		
Description	This container contains the definition of a single role. Each role is a bit in the authentication role bitfield.		
Multiplicity	0..31		
Configuration Parameters			

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspAuthenticationRoleBitPosition	1	[ECUC_Dcm_01212]

No Included Containers

]

[ECUC_Dcm_01212] Definition of EcucIntegerParamDef DcmDspAuthenticationRoleBitPosition

Parameter Name	DcmDspAuthenticationRoleBitPosition	
Parent Container	DcmDspAuthenticationRow	
Description	Defines the bit number that represents the role in the authentication bit field. The bit mask is aligned with the logical bit calculations: $2^{\text{DcmDspAuthenticationRoleBitPosition}}$.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 31	
Default value	–	
Dependency		

]

10.2.5.5 DcmDspAuthenticationConnection

[ECUC_Dcm_01162] Definition of EcucParamConfContainerDef DcmDspAuthenticationConnection

Container Name	DcmDspAuthenticationConnection	
Parent Container	DcmDspAuthentication	
Description	This container contains the authentication configuration use for a dsl connection.	
Multiplicity	0..*	
Configuration Parameters		

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspAuthenticationEcuChallengeLength	1	[ECUC_Dcm_01186]
DcmDspAuthenticationClientChallengeSignJobRef	1	[ECUC_Dcm_01174]
DcmDspAuthenticationConnectionCertificateRef	1	[ECUC_Dcm_01164]
DcmDspAuthenticationConnectionMainConnectionRef	1	[ECUC_Dcm_01163]
DcmDspAuthenticationECUCertificateRef	0..1	[ECUC_Dcm_01177]
DcmDspAuthenticationRandomJobRef	1	[ECUC_Dcm_01173]
DcmDspAuthenticationRoleElementRef	1	[ECUC_Dcm_01167]
DcmDspAuthenticationVerifyProofOfOwnershipClientJobRef	1	[ECUC_Dcm_01175]
DcmDspAuthenticationWhiteListDIDElementRef	0..1	[ECUC_Dcm_01169]
DcmDspAuthenticationWhiteListMemorySelectionElementRef	0..1	[ECUC_Dcm_01171]
DcmDspAuthenticationWhiteListRIDElementRef	0..1	[ECUC_Dcm_01170]
DcmDspAuthenticationWhiteListServicesElementRef	0..1	[ECUC_Dcm_01168]

No Included Containers

]

[ECUC_Dcm_01186] Definition of EcucIntegerParamDef DcmDspAuthenticationEcuChallengeLength

Parameter Name	DcmDspAuthenticationEcuChallengeLength		
Parent Container	DcmDspAuthenticationConnection		
Description	Length in byte of the ECU challenge		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01174] Definition of EcucReferenceDef DcmDspAuthenticationClientChallengeSignJobRef

Parameter Name	DcmDspAuthenticationClientChallengeSignJobRef		
Parent Container	DcmDspAuthenticationConnection		
Description	Reference to a job used to sign the client challenge.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01164] Definition of EcucReferenceDef DcmDspAuthenticationConnectionCertificateRef

Parameter Name	DcmDspAuthenticationConnectionCertificateRef		
Parent Container	DcmDspAuthenticationConnection		
Description	Reference to a KeyMCertificate used to store the certificate within the KeyM.		
Multiplicity	1		
Type	Symbolic name reference to KeyMCertificate		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants





Value Configuration Class	Link time	–	
	Post-build time	–	
	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01163] Definition of EcucReferenceDef DcmDspAuthentication ConnectionMainConnectionRef [

Parameter Name	DcmDspAuthenticationConnectionMainConnectionRef		
Parent Container	DcmDspAuthenticationConnection		
Description	Reference to the dsl diagnostic connection that uses this authentication configuration.		
Multiplicity	1		
Type	Reference to DcmDslMainConnection		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01177] Definition of EcucReferenceDef DcmDspAuthentication ECUCertificateRef [

Parameter Name	DcmDspAuthenticationECUCertificateRef		
Parent Container	DcmDspAuthenticationConnection		
Description	Reference to a KeyMCertificate with the server certificate for bi-directional authentication.		
Multiplicity	0..1		
Type	Symbolic name reference to KeyMCertificate		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01173] Definition of EcucReferenceDef DcmDspAuthenticationRandomJobRef

Parameter Name	DcmDspAuthenticationRandomJobRef		
Parent Container	DcmDspAuthenticationConnection		
Description	Reference to a certificate parse job used to parse the authentication certificate.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_01167] Definition of EcucReferenceDef DcmDspAuthenticationRoleElementRef

Parameter Name	DcmDspAuthenticationRoleElementRef		
Parent Container	DcmDspAuthenticationConnection		
Description	Reference to a certificate data element with the role in the certificate.		
Multiplicity	1		
Type	Symbolic name reference to KeyMCCertificateElement		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_01175] Definition of EcucReferenceDef DcmDspAuthenticationVerifyProofOfOwnershipClientJobRef

Parameter Name	DcmDspAuthenticationVerifyProofOfOwnershipClientJobRef		
Parent Container	DcmDspAuthenticationConnection		
Description	Reference to a CsmJob used to verify the proof of ownership client in the Csm.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		





Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01169] Definition of EcucReferenceDef DcmDspAuthenticationWhiteListDIDElementRef [

Parameter Name	DcmDspAuthenticationWhiteListDIDElementRef		
Parent Container	DcmDspAuthenticationConnection		
Description	Reference to a certificate data element with the white list for data identifiers in the certificate.		
Multiplicity	0..1		
Type	Symbolic name reference to KeyMCertificateElement		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01171] Definition of EcucReferenceDef DcmDspAuthenticationWhiteListMemorySelectionElementRef [

Parameter Name	DcmDspAuthenticationWhiteListMemorySelectionElementRef		
Parent Container	DcmDspAuthenticationConnection		
Description	Reference to a certificate data element with the white list for user defined memory selection in the certificate.		
Multiplicity	0..1		
Type	Symbolic name reference to KeyMCertificateElement		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	





Dependency	

]

[ECUC_Dcm_01170] Definition of EcucReferenceDef DcmDspAuthenticationWhiteListRIDElementRef [

Parameter Name	DcmDspAuthenticationWhiteListRIDElementRef		
Parent Container	DcmDspAuthenticationConnection		
Description	Reference to a certificate data element with the white list for routine identifiers in the certificate.		
Multiplicity	0..1		
Type	Symbolic name reference to KeyMCertificateElement		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01168] Definition of EcucReferenceDef DcmDspAuthenticationWhiteListServicesElementRef [

Parameter Name	DcmDspAuthenticationWhiteListServicesElementRef		
Parent Container	DcmDspAuthenticationConnection		
Description	Reference to a certificate data element with the white list in the certificate.		
Multiplicity	0..1		
Type	Symbolic name reference to KeyMCertificateElement		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.6 Communication Control

10.2.5.6.1 DcmDspComControl

[ECUC_Dcm_00900] Definition of EcucParamConfContainerDef DcmDspComControl [

Container Name	DcmDspComControl
Parent Container	DcmDsp
Description	Provides the configuration of the CommunicationControl mechanism.
Multiplicity	0..1
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspComControlAllChannel	0..*	Collection of ComM channels which shall be controlled if all networks are addressed.
DcmDspComControlSetting	0..1	Provide the configuration of the Communication control.
DcmDspComControlSpecificChannel	0..*	Assigns subnet number to ComM channel which will be controlled.
DcmDspComControlSubNode	0..65535	This container gives information about the node identification number and the ComM channel used to address a sub-network.

]

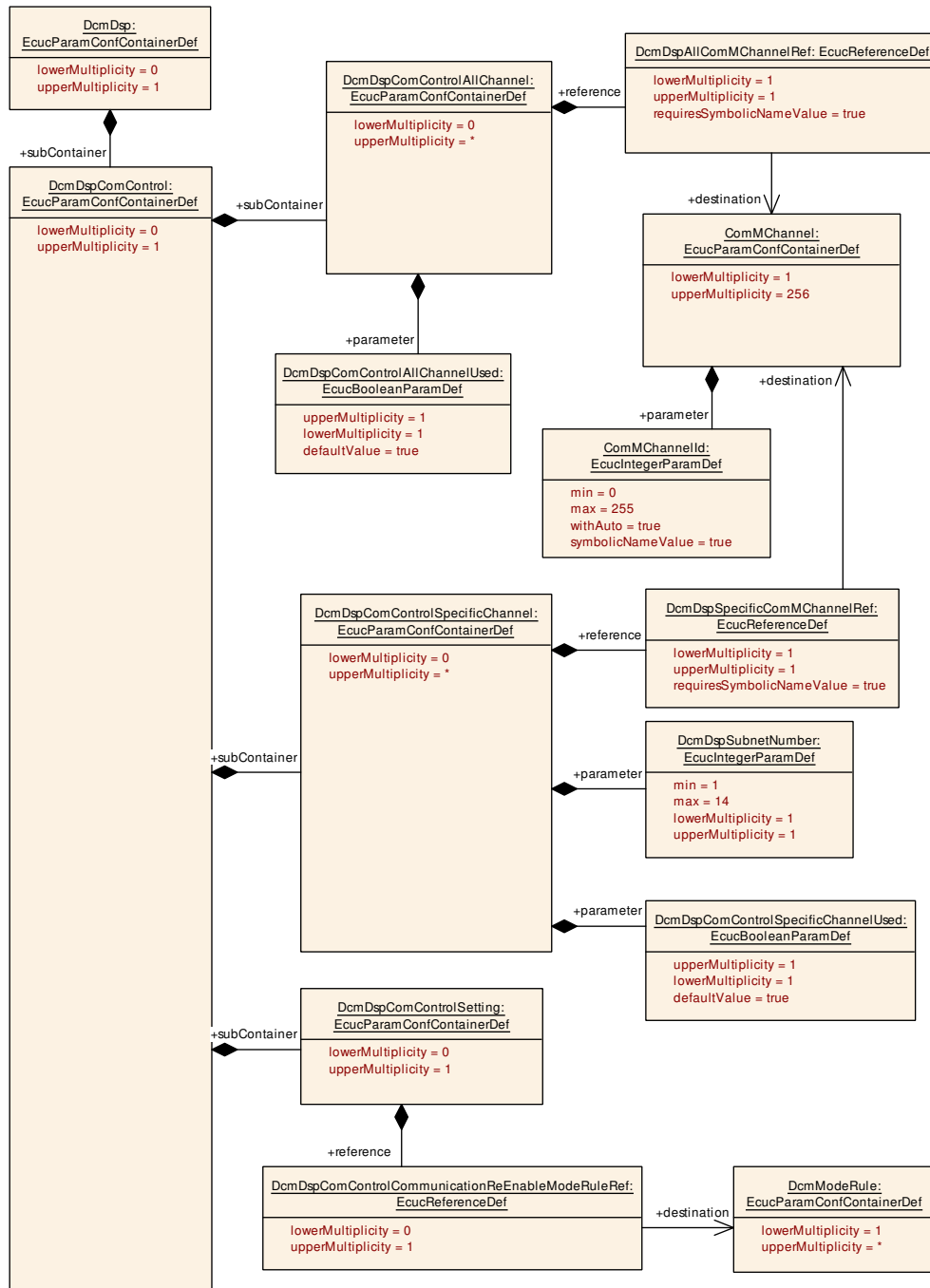


Figure 10.17: DcmDspComControl configuration overview

10.2.5.6.2 DcmDspComControlAllChannel

[ECUC_Dcm_00901] Definition of EcucParamConfContainerDef DcmDspComControlAllChannel

Container Name	DcmDspComControlAllChannel
Parent Container	DcmDspComControl
Description	Collection of ComM channels which shall be controlled if all networks are addressed.
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspComControlAllChannelUsed	1	[ECUC_Dcm_01045]
DcmDspAllComMChannelRef	1	[ECUC_Dcm_00902]

No Included Containers

]

[[ECUC_Dcm_01045](#)] Definition of EcucBooleanParamDef DcmDspComControlAllChannelUsed [

Parameter Name	DcmDspComControlAllChannelUsed		
Parent Container	DcmDspComControlAllChannel		
Description	Allow to activate or deactivate the usage of a ComM channel collection to be controlled, for multi purpose ECUs true = ComM channel collection used false = ComM channel collection not used		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

]

[[ECUC_Dcm_00902](#)] Definition of EcucReferenceDef DcmDspAllComMChannelRef [

Parameter Name	DcmDspAllComMChannelRef		
Parent Container	DcmDspComControlAllChannel		
Description	Reference to ComM channel.		
Multiplicity	1		
Type	Symbolic name reference to ComMChannel		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

]

10.2.5.6.3 DcmDspComControlSetting

[ECUC_Dcm_00943] Definition of EcucParamConfContainerDef DcmDspComControlSetting [

Container Name	DcmDspComControlSetting
Parent Container	DcmDspComControl
Description	Provide the configuration of the Communication control.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspComControlCommunicationReEnableModeRuleRef	0..1	[ECUC_Dcm_00944]

No Included Containers

]

[ECUC_Dcm_00944] Definition of EcucReferenceDef DcmDspComControlCommunicationReEnableModeRuleRef [

Parameter Name	DcmDspComControlCommunicationReEnableModeRuleRef		
Parent Container	DcmDspComControlSetting		
Description	Reference to DcmModeRule Mode rule which controls re-enabling of communication by DCM. [ref. SWS_Dcm_00753]		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

10.2.5.6.4 DcmDspComControlSpecificChannel

[ECUC_Dcm_00903] Definition of EcucParamConfContainerDef DcmDspComControlSpecificChannel [

Container Name	DcmDspComControlSpecificChannel
Parent Container	DcmDspComControl
Description	Assigns subnet number to ComM channel which will be controlled.
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspComControlSpecificChannelUsed	1	[ECUC_Dcm_01046]
DcmDspSubnetNumber	1	[ECUC_Dcm_00905]
DcmDspSpecificComMChannelRef	1	[ECUC_Dcm_00904]

No Included Containers

]

[[ECUC_Dcm_01046](#)] Definition of EcucBooleanParamDef DcmDspComControlSpecificChannelUsed [

Parameter Name	DcmDspComControlSpecificChannelUsed		
Parent Container	DcmDspComControlSpecificChannel		
Description	Allow to activate or deactivate the usage of a Subnet assigned to the ComM channel which will be controlled, for multi purpose ECUs. true = Subnet used false = Subnet not used		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

]

[[ECUC_Dcm_00905](#)] Definition of EcucIntegerParamDef DcmDspSubnetNumber [

Parameter Name	DcmDspSubnetNumber		
Parent Container	DcmDspComControlSpecificChannel		
Description	Subnet Number which controls the specific ComMChannel.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 14		
Default value	-		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	-	





	Post-build time	–	
Dependency			

[ECUC_Dcm_00904] Definition of EcucReferenceDef DcmDspSpecificComMChannelRef

Parameter Name	DcmDspSpecificComMChannelRef		
Parent Container	DcmDspComControlSpecificChannel		
Description	Reference to ComM channel.		
Multiplicity	1		
Type	Symbolic name reference to ComMChannel		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

10.2.5.6.5 DcmDspComControlSubNode

[ECUC_Dcm_01033] Definition of EcucParamConfContainerDef DcmDspComControlSubNode

Container Name	DcmDspComControlSubNode
Parent Container	DcmDspComControl
Description	This container gives information about the node identification number and the ComM channel used to address a sub-network.
Multiplicity	0..65535
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspComControlSubNodeId	1	[ECUC_Dcm_01031]
DcmDspComControlSubNodeUsed	1	[ECUC_Dcm_01032]
DcmDspComControlSubNodeComMChannelRef	1	[ECUC_Dcm_01030]

No Included Containers

[ECUC_Dcm_01031] Definition of EcucIntegerParamDef DcmDspComControlSubNodeId

Parameter Name	DcmDspComControlSubNodeId		
Parent Container	DcmDspComControlSubNode		
Description	The node identification number DcmDspComControlSubNodeId is addressed by the CommunicationControl (0x28) request.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_01032] Definition of EcucBooleanParamDef DcmDspComControlSubNodeUsed

Parameter Name	DcmDspComControlSubNodeUsed		
Parent Container	DcmDspComControlSubNode		
Description	This parameter determines if a node control function is available or not.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

[ECUC_Dcm_01030] Definition of EcucReferenceDef DcmDspComControlSubNodeComMChannelRef

Parameter Name	DcmDspComControlSubNodeComMChannelRef		
Parent Container	DcmDspComControlSubNode		
Description	This parameter references a ComM channel where this node is connected to.		
Multiplicity	1		
Type	Symbolic name reference to ComMChannel		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

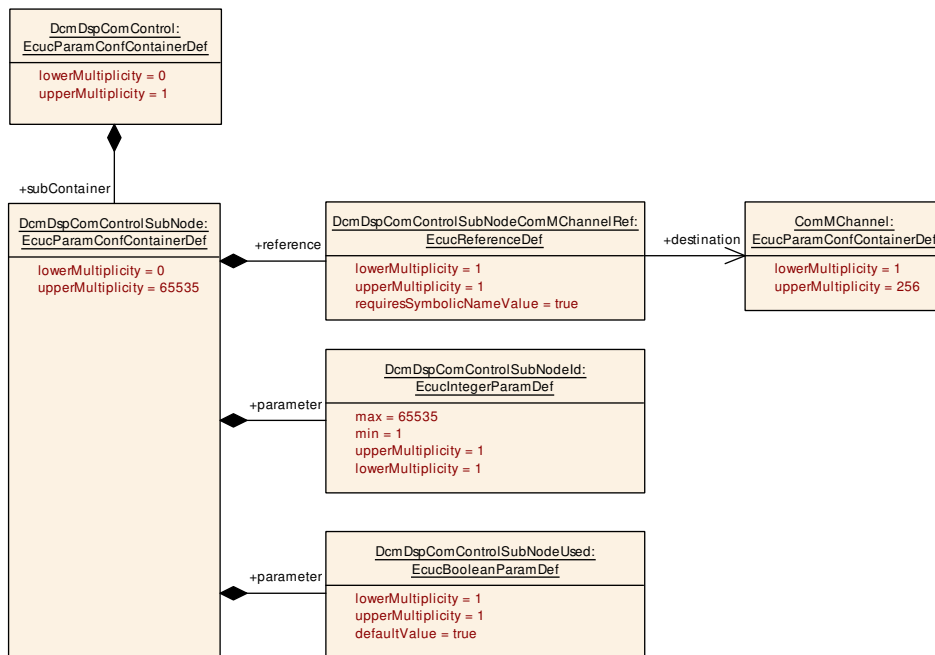


Figure 10.18: DcmDspComControlSubNode configuration overview

10.2.5.7 DcmDspCommonAuthorization

[ECUC_Dcm_01025] Definition of EcucParamConfContainerDef DcmDspCommonAuthorization [

Container Name	DcmDspCommonAuthorization
Parent Container	DcmDsp
Description	This container contains the configuration (parameters) for the common Authorization being equal for several services / sub-services.
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspCommonAuthorizationModeRuleRef	0..1	[ECUC_Dcm_01028]
DcmDspCommonAuthorizationSecurityLevelRef	0..*	[ECUC_Dcm_01026]
DcmDspCommonAuthorizationSessionRef	0..*	[ECUC_Dcm_01027]

No Included Containers

]

[ECUC_Dcm_01028] Definition of EcucReferenceDef DcmDspCommonAuthorizationModeRuleRef

Parameter Name	DcmDspCommonAuthorizationModeRuleRef		
Parent Container	DcmDspCommonAuthorization		
Description	Reference to DcmModeRule Mode rule which controls this service/ sub-service. If there is no reference, no check of the mode rule shall be done.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_01026] Definition of EcucReferenceDef DcmDspCommonAuthorizationSecurityLevelRef

Parameter Name	DcmDspCommonAuthorizationSecurityLevelRef		
Parent Container	DcmDspCommonAuthorization		
Description	Reference to DcmDspSecurityRow Security levels allowed to control this service/ sub-service. If there is no reference, no check of security level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSecurityRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_01027] Definition of EcucReferenceDef DcmDspCommonAuthorizationSessionRef

Parameter Name	DcmDspCommonAuthorizationSessionRef		
Parent Container	DcmDspCommonAuthorization		
Description	Reference to DcmDspSessionRow Sessions allowed to control this service/ sub-service. If there is no reference, no check of session level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSessionRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

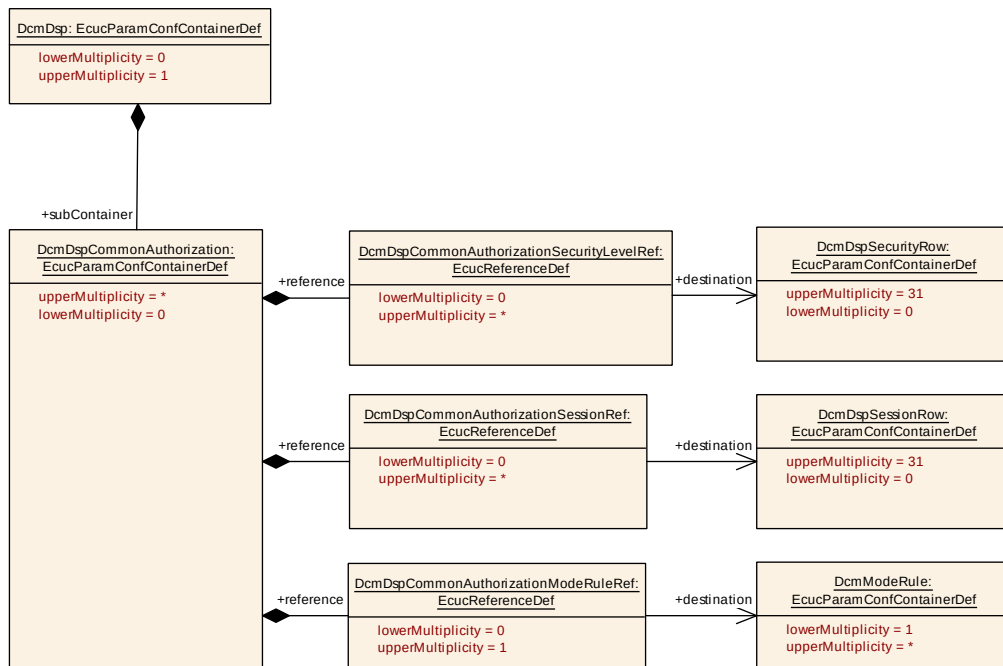


Figure 10.19: DcmDspCommonAuthorization configuration overview

10.2.5.8 DIDs

10.2.5.8.1 DcmDspDid

[ECUC_Dcm_00601] Definition of EcucParamConfContainerDef DcmDspDid

Container Name	DcmDspDid
Parent Container	DcmDsp
Description	This container contains the configuration (parameters) of the DID.
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDidIdentifier	1	[ECUC_Dcm_00602]
DcmDspDidSize	0..1	[ECUC_Dcm_01099]
DcmDspDidUsed	1	[ECUC_Dcm_00805]
DcmDspDidUsePort	1	[ECUC_Dcm_01122]
DcmDspDidBndMBlockIdRef	0..1	[ECUC_Dcm_01185]
DcmDspDidInfoRef	1	[ECUC_Dcm_00604]
DcmDspDidRef	0..*	[ECUC_Dcm_00606]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspDidSignal	0..*	This container defines the reference to 1 DcmDspData container and position relevant for this DID.
DcmDspDidSignalCompositePool	0..*	Defines a collection of all contained signals for this root signal.
DcmDspDidSupportInfo	0..1	This container defines the support information to declare the usability of the data bytes within the DIDs

]

[ECUC_Dcm_00602] Definition of EcucIntegerParamDef DcmDspDidIdentifier [

Parameter Name	DcmDspDidIdentifier		
Parent Container	DcmDspDid		
Description	2 byte Identifier of the DID Within each DcmConfigSet all DcmDspDidIdentifier values shall be unique.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01099] Definition of EcucIntegerParamDef DcmDspDidSize [

Parameter Name	DcmDspDidSize		
Parent Container	DcmDspDid		
Description	Length of a DID in byte(s).		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00805] Definition of EcucBooleanParamDef DcmDspDidUsed [

Parameter Name	DcmDspDidUsed		
Parent Container	DcmDspDid		
Description	Allow to activate or deactivate the usage of a DID, for multi purpose ECUs true = DID available false = DID not available		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

]

[ECUC_Dcm_01122] Definition of EcucEnumerationParamDef DcmDspDidUse Port [

Parameter Name	DcmDspDidUsePort	
Parent Container	DcmDspDid	
Description	Selects application interface type for DID data elements between a single operation for all data elements or data element specific operations.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	USE_ATOMIC_BNDM	The DID shall be read/written from/to BndM.
	USE_ATOMIC_NV_DATA_INTERFACE	A single sender receiver interface with Nv Data-Ports is used for all data elements of this DID.
	USE_ATOMIC_SENDER_RECEIVER_INTERFACE	A single sender receiver interface is used to access all data elements of this DID. The sender receiver interface is typed with IsService=false.





	USE_ATOMIC_SENDER_RECEIVER_INTERFACE_AS_SERVICE	A single sender receiver interface is used to access all data elements of this DID. The sender receiver interface is typed with IsService=true.		
	USE_DATA_ELEMENT_SPECIFIC_INTERFACES	The data elements of this DID are collected by using the data element specific interfaces defined by DcmDspDataUsePort.		
Default value	USE_DATA_ELEMENT_SPECIFIC_INTERFACES			
Post-Build Variant Value	false			
Value Configuration Class	Pre-compile time	X	All Variants	
	Link time	—		
	Post-build time	—		
Dependency				

└

[ECUC_Dcm_01185] Definition of EcucReferenceDef DcmDspDidBndMBlockIdRef

Parameter Name	DcmDspDidBndMBlockIdRef		
Parent Container	DcmDspDid		
Description	Associate this DID with a BndM blockId.		
Multiplicity	0..1		
Type	Symbolic name reference to BndMBlockDescriptor		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency	DcmDspDidUsePort == USE_ATOMIC_BNDM		

└

[ECUC_Dcm_00604] Definition of EcucReferenceDef DcmDspDidInfoRef

Parameter Name	DcmDspDidInfoRef		
Parent Container	DcmDspDid		
Description	Reference to DcmDspDidInfo containing information on this DID.		
Multiplicity	1		
Type	Reference to DcmDspDidInfo		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00606] Definition of EcucReferenceDef DcmDspDidRef [

Parameter Name	DcmDspDidRef		
Parent Container	DcmDspDid		
Description	Reference to DcmDspDid in case this DID refer to one or several other DID's Attributes: requiresIndex=true		
Multiplicity	0..*		
Type	Reference to DcmDspDid		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

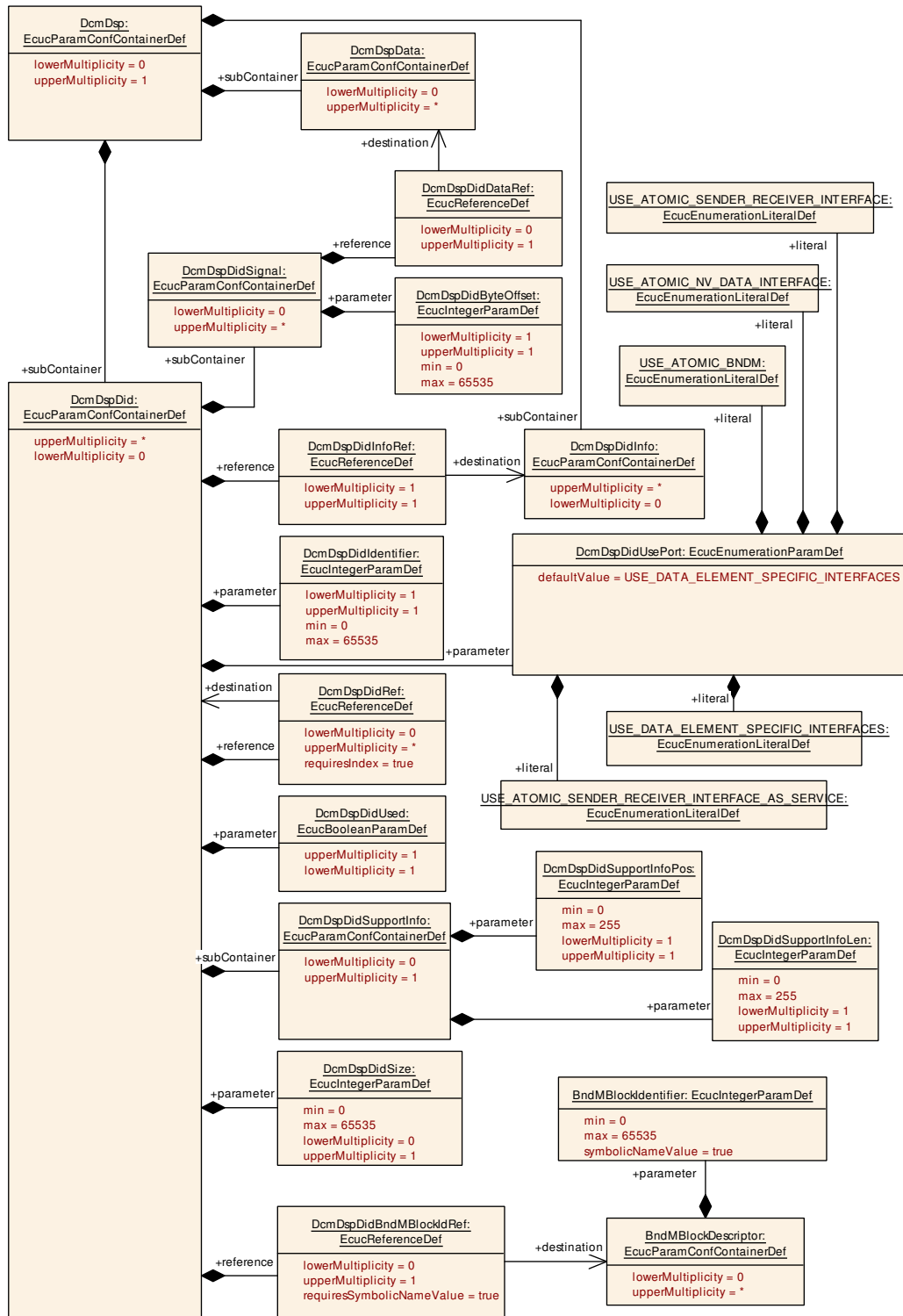


Figure 10.20: DcmDspDid configuration overview

10.2.5.8.2 DcmDspDidInfo

[ECUC_Dcm_00607] Definition of EcucParamConfContainerDef DcmDspDidInfo

Container Name	DcmDspDidInfo
Parent Container	DcmDsp
Description	This container contains the configuration (parameters) of the DID's Info
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDDDIDMaxElements	0..1	[ECUC_Dcm_00970]
DcmDspDidDynamicallyDefined	1	[ECUC_Dcm_00612]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspDidControl	0..1	This container contains the configuration (parameters) of the DID control.
DcmDspDidRead	0..1	This container contains the configuration (parameters) of the DID read.
DcmDspDidWrite	0..1	This container contains the configuration (parameters) of the DID write.

[ECUC_Dcm_00970] Definition of EcucIntegerParamDef DcmDspDDDIDMaxElements

Parameter Name	DcmDspDDDIDMaxElements		
Parent Container	DcmDspDidInfo		
Description	Maximum number of source elements of a DDDID.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_00612] Definition of EcucBooleanParamDef DcmDspDidDynamicallyDefined

Parameter Name	DcmDspDidDynamicallyDefined		
Parent Container	DcmDspDidInfo		
Description	Indicates if this DID can be dynamically defined true = DID can be dynamically defined false = DID can not be dynamically defined		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

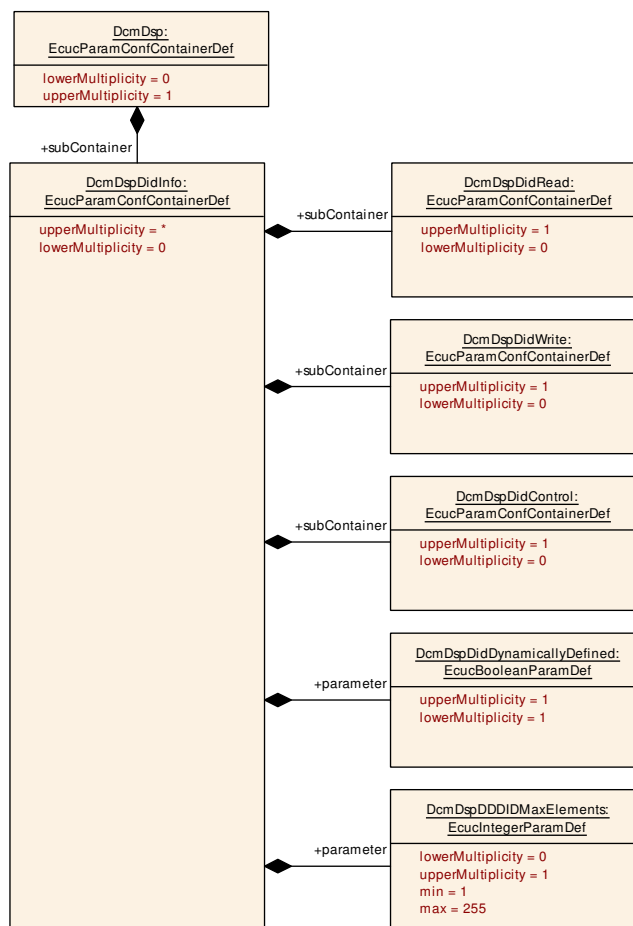


Figure 10.21: DcmDspDidInfo configuration overview

10.2.5.8.3 DcmDspDidRead

[ECUC_Dcm_00613] Definition of EcucParamConfContainerDef DcmDspDidRead

Container Name	DcmDspDidRead
Parent Container	DcmDspDidInfo
Description	This container contains the configuration (parameters) of the DID read.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDidReadModeRuleRef	0..1	[ECUC_Dcm_00917]
DcmDspDidReadRoleRef	0..32	[ECUC_Dcm_01141]
DcmDspDidReadSecurityLevelRef	0..*	[ECUC_Dcm_00614]
DcmDspDidReadSessionRef	0..*	[ECUC_Dcm_00615]

No Included Containers

]

[ECUC_Dcm_00917] Definition of EcucReferenceDef DcmDspDidReadModeRuleRef

Parameter Name	DcmDspDidReadModeRuleRef		
Parent Container	DcmDspDidRead		
Description	Reference to DcmModeRule Mode rule which controls to read this DID. If there is no reference, no check of the mode rule shall be done.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01141] Definition of EcucReferenceDef DcmDspDidReadRoleRef [

Parameter Name	DcmDspDidReadRoleRef		
Parent Container	DcmDspDidRead		
Description	Reference to DcmDspAuthenticationRow that defines a role in that this DID can be read.		
Multiplicity	0..32		
Type	Reference to DcmDspAuthenticationRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00614] Definition of EcucReferenceDef DcmDspDidReadSecurityLevelRef [

Parameter Name	DcmDspDidReadSecurityLevelRef		
Parent Container	DcmDspDidRead		
Description	Reference to DcmDspSecurityRow Referenced security levels are allowed to read this DID. If there is no reference, no check of security level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSecurityRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00615] Definition of EcucReferenceDef DcmDspDidReadSessionRef [

Parameter Name	DcmDspDidReadSessionRef		
Parent Container	DcmDspDidRead		
Description	Reference to DcmDspSessionRow Referenced sessions are allowed to read this DID. If there is no reference, no check of session level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSessionRow		
Post-Build Variant Multiplicity	false		





Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

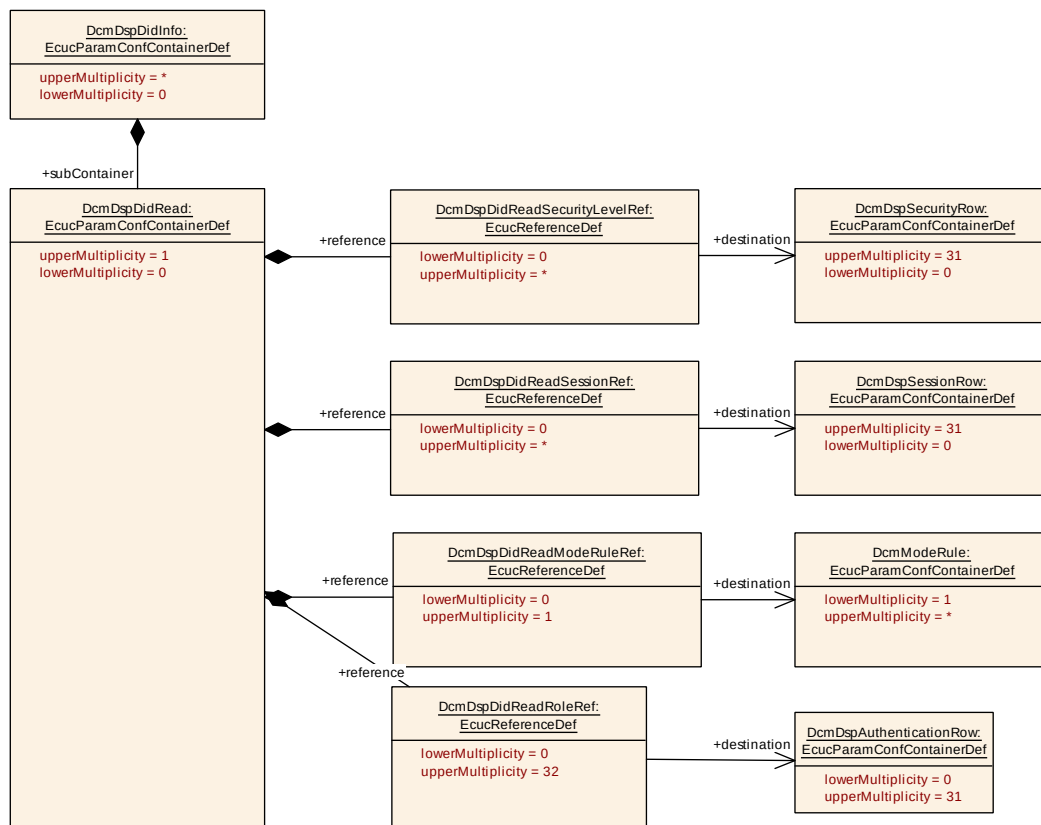


Figure 10.22: DcmDspDidRead configuration overview

10.2.5.8.4 DcmDspDidSignal

[ECUC_Dcm_00813] Definition of EcucParamConfContainerDef DcmDspDidSignal

Container Name	DcmDspDidSignal
Parent Container	DcmDspDid , DcmDspDidSignalCompositePool
Description	This container defines the reference to 1 DcmDspData container and position relevant for this DID.
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDataByteSize	0..1	[ECUC_Dcm_01106]
DcmDspDidByteOffset	1	[ECUC_Dcm_01105]
DcmDspDidDataRef	0..1	[ECUC_Dcm_00808]
DcmDspDidSignalCompositeRef	0..*	[ECUC_Dcm_01232]

No Included Containers

For parameter table [ECUC_Dcm_01106] [DcmDspDataByteSize](#), see definition below container [DcmDspData](#).

[ECUC_Dcm_01105] Definition of EcucIntegerParamDef DcmDspDidByteOffset [

Parameter Name	DcmDspDidByteOffset		
Parent Container	DcmDspDidSignal		
Description	Defines the absolute byte offset of the data defined by DcmDspDidDataRef reference to DcmDspData container in the DID.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_00808] Definition of EcucReferenceDef DcmDspDidDataRef [

Parameter Name	DcmDspDidDataRef
Parent Container	DcmDspDidSignal
Description	Reference to 1 DcmDspData container relevant for this DID.
Multiplicity	0..1
Type	Reference to DcmDspData
Post-Build Variant Value	false





Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01232] Definition of EcucReferenceDef DcmDspDidSignalCompositeRef

Parameter Name	DcmDspDidSignalCompositeRef		
Parent Container	DcmDspDidSignal		
Description	Reference to the contained signal.		
Multiplicity	0..*		
Type	Reference to DcmDspDidSignalCompositePool		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

10.2.5.8.5 DcmDspDidSignalCompositePool

[ECUC_Dcm_01231] Definition of EcucParamConfContainerDef DcmDspDidSignalCompositePool

Container Name	DcmDspDidSignalCompositePool		
Parent Container	DcmDspDid		
Description	Defines a collection of all contained signals for this root signal.		
Multiplicity	0..*		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspDidSignal	0..*	This container defines the reference to 1 DcmDspData container and position relevant for this DID.

]

10.2.5.8.6 DcmDspDidSupportInfo

[ECUC_Dcm_01102] Definition of EcucParamConfContainerDef DcmDspDidSupportInfo [

Container Name	DcmDspDidSupportInfo
Parent Container	DcmDspDid
Description	This container defines the support information to declare the usability of the data bytes within the DIDs
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDidSupportInfoLen	1	[ECUC_Dcm_01103]
DcmDspDidSupportInfoPos	1	[ECUC_Dcm_01100]

No Included Containers

]

[ECUC_Dcm_01103] Definition of EcucIntegerParamDef DcmDspDidSupportInfoLen [

Parameter Name	DcmDspDidSupportInfoLen		
Parent Container	DcmDspDidSupportInfo		
Description	Length of the support information in bytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01100] Definition of EcucIntegerParamDef DcmDspDidSupportInfoPos

Parameter Name	DcmDspDidSupportInfoPos		
Parent Container	DcmDspDidSupportInfo		
Description	Position of the support information in bytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

10.2.5.8.7 DcmDspDidRange

[ECUC_Dcm_00937] Definition of EcucParamConfContainerDef DcmDspDidRange

Container Name	DcmDspDidRange
Parent Container	DcmDsp
Description	This container defines the DID Range
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDidRangeHasGaps	1	[ECUC_Dcm_00941]
DcmDspDidRangeIdentifierLowerLimit	1	[ECUC_Dcm_00938]
DcmDspDidRangeIdentifierUpperLimit	1	[ECUC_Dcm_00939]
DcmDspDidRangeIsDidAvailableFnc	0..1	[ECUC_Dcm_00946]
DcmDspDidRangeMaxDataLength	1	[ECUC_Dcm_00940]
DcmDspDidRangeReadDataLengthFnc	0..1	[ECUC_Dcm_01067]
DcmDspDidRangeReadDidFnc	0..1	[ECUC_Dcm_00947]
DcmDspDidRangeUsePort	1	[ECUC_Dcm_00945]
DcmDspDidRangeWriteDidFnc	0..1	[ECUC_Dcm_00948]
DcmDspDidRangeInfoRef	1	[ECUC_Dcm_00950]

No Included Containers

[ECUC_Dcm_00941] Definition of EcucBooleanParamDef DcmDspDidRangeHas Gaps

Parameter Name	DcmDspDidRangeHasGaps		
Parent Container	DcmDspDidRange		
Description	Parameter specifying if there are gaps in the DID range (parameter set to TRUE) or not (parameter set to FALSE)		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_00938] Definition of EcucIntegerParamDef DcmDspDidRangeldentifierLowerLimit

Parameter Name	DcmDspDidRangeldentifierLowerLimit		
Parent Container	DcmDspDidRange		
Description	Lower limit of DID range.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_00939] Definition of EcucIntegerParamDef DcmDspDidRangeldentifierUpperLimit

Parameter Name	DcmDspDidRangeldentifierUpperLimit		
Parent Container	DcmDspDidRange		
Description	Upper limit of DID range.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		





Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00946] Definition of EcucFunctionNameDef DcmDspDidRangels DidAvailableFnc

Parameter Name	DcmDspDidRangelsDidAvailableFnc		
Parent Container	DcmDspDidRange		
Description	Function name to request from application if a specific DID is available within the range or not. Only relevant if DcmDspDidRangeUsePort is set to false. This parameter is related to the interface Xxx_IsDidAvailable.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00940] Definition of EcucIntegerParamDef DcmDspDidRangeMaxDataLength

Parameter Name	DcmDspDidRangeMaxDataLength		
Parent Container	DcmDspDidRange		
Description	Maximum data length in bytes		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01067] Definition of EcucFunctionNameDef DcmDspDidRangeReadDataLengthFnc

Parameter Name	DcmDspDidRangeReadDataLengthFnc		
Parent Container	DcmDspDidRange		
Description	Function name to request from application the length of the data of a range DID. Only relevant if DcmDspDidRangeUsePort is set to false. This parameter is related to the interface Xxx_ReadDidRangeDataLength.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00947] Definition of EcucFunctionNameDef DcmDspDidRangeReadDidFnc

Parameter Name	DcmDspDidRangeReadDidFnc		
Parent Container	DcmDspDidRange		
Description	Function name to request from application the data range value of a DID.(Read Data-function). Only relevant if DcmDspDidRangeUsePort is set to false. This parameter is related to the interface Xxx_ReadDidData.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00945] Definition of EcucBooleanParamDef DcmDspDidRangeUsePort

Parameter Name	DcmDspDidRangeUsePort		
Parent Container	DcmDspDidRange		
Description	When the parameter DcmDspDidRangeUsePort is set to true the DCM will access the Data using an R-Port requiring a PortInterface DataServices_DIDRange. In that case, DcmDspDidRangeIsDidAvailableFnc, DcmDspDidRangeReadDidFnc and DcmDspDidRangeWriteDidFnc are ignored and the RTE APIs are used. When the parameter DcmDspDidRangeUsePort is false, the DCM calls the functions defined in DcmDspDidRangeIsDidAvailableFnc, DcmDspDidRangeReadDidFnc and DcmDspDidRangeWriteDidFnc.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00948] Definition of EcucFunctionNameDef DcmDspDidRangeWriteDidFnc

Parameter Name	DcmDspDidRangeWriteDidFnc		
Parent Container	DcmDspDidRange		
Description	Function name to request application to write the data range value of a DID.(Write Data-function). Only relevant if DcmDspDidRangeUsePort is set to false. This parameter is related to the interface Xxx_WriteDidData.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00950] Definition of EcucReferenceDef DcmDspDidRangeInfoRef

Parameter Name	DcmDspDidRangeInfoRef		
Parent Container	DcmDspDidRange		
Description	Reference to DcmDspDidInfo containing information on this DID Range.		





Multiplicity	1		
Type	Reference to DcmDspDidInfo		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

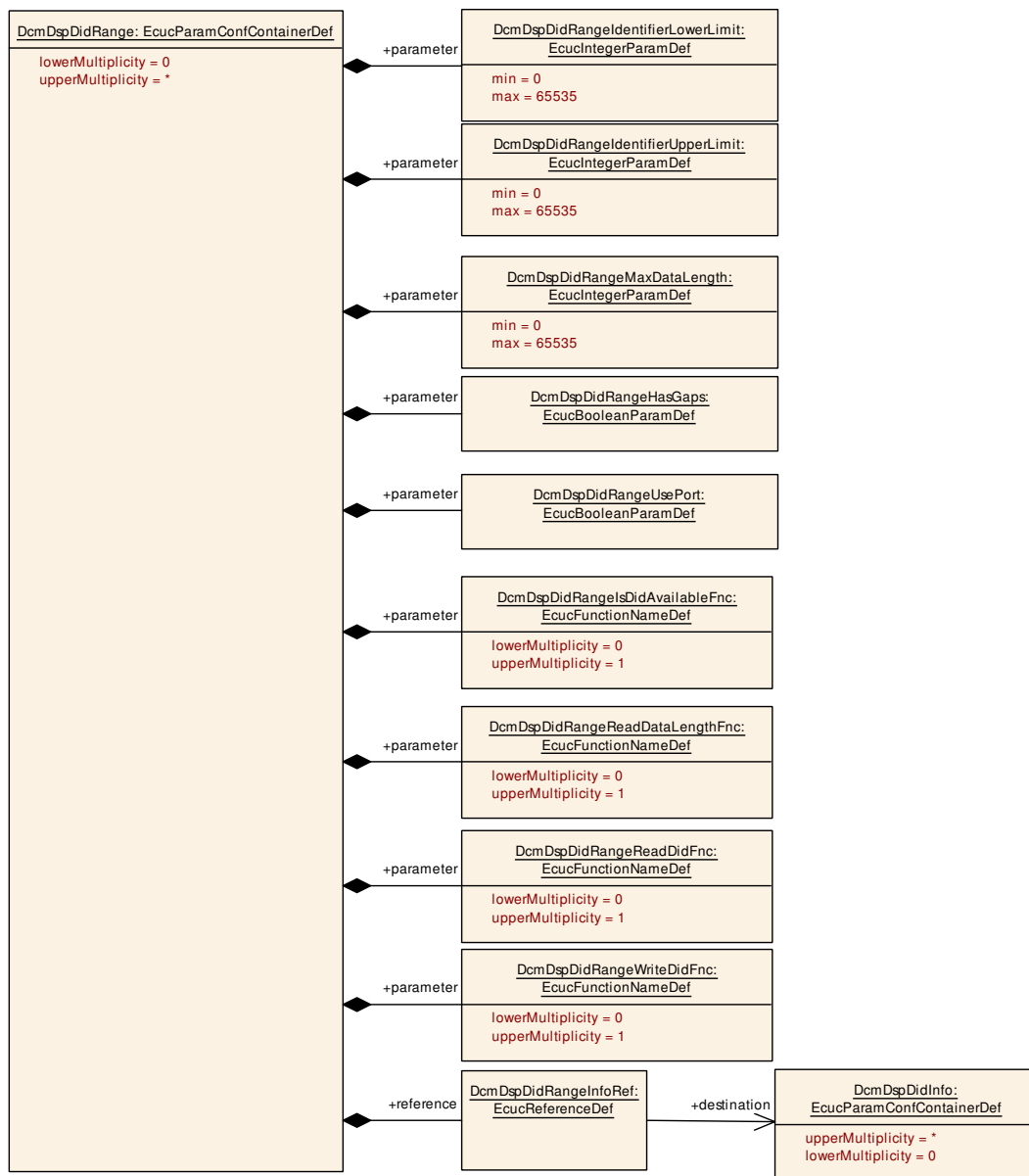


Figure 10.23: DcmDspDidRange configuration overview

10.2.5.8.8 DcmDspDidWrite

[ECUC_Dcm_00616] Definition of EcucParamConfContainerDef DcmDspDidWrite

Container Name	DcmDspDidWrite
Parent Container	DcmDspDidInfo
Description	This container contains the configuration (parameters) of the DID write.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDidWriteModeRuleRef	0..1	[ECUC_Dcm_00922]
DcmDspDidWriteRoleRef	0..32	[ECUC_Dcm_01142]
DcmDspDidWriteSecurityLevelRef	0..*	[ECUC_Dcm_00617]
DcmDspDidWriteSessionRef	0..*	[ECUC_Dcm_00618]

No Included Containers

]

[ECUC_Dcm_00922] Definition of EcucReferenceDef DcmDspDidWriteModeRuleRef

Parameter Name	DcmDspDidWriteModeRuleRef		
Parent Container	DcmDspDidWrite		
Description	Reference to DcmModeRule Mode rule which controls to write this DID. If there is no reference, no check of the mode rule shall be done.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01142] Definition of EcucReferenceDef DcmDspDidWriteRoleRef [

Parameter Name	DcmDspDidWriteRoleRef		
Parent Container	DcmDspDidWrite		
Description	Reference to DcmDspAuthenticationRow that defines a role in that this DID can be written.		
Multiplicity	0..32		
Type	Reference to DcmDspAuthenticationRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00617] Definition of EcucReferenceDef DcmDspDidWriteSecurityLevelRef [

Parameter Name	DcmDspDidWriteSecurityLevelRef		
Parent Container	DcmDspDidWrite		
Description	Reference to DcmDspSecurityRow Referenced security levels are allowed to write this DID. If there is no reference, no check of security level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSecurityRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00618] Definition of EcucReferenceDef DcmDspDidWriteSessionRef [

Parameter Name	DcmDspDidWriteSessionRef		
Parent Container	DcmDspDidWrite		
Description	Reference to DcmDspSessionRow Referenced sessions are allowed to write this DID. If there is no reference, no check of session level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSessionRow		
Post-Build Variant Multiplicity	false		





Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

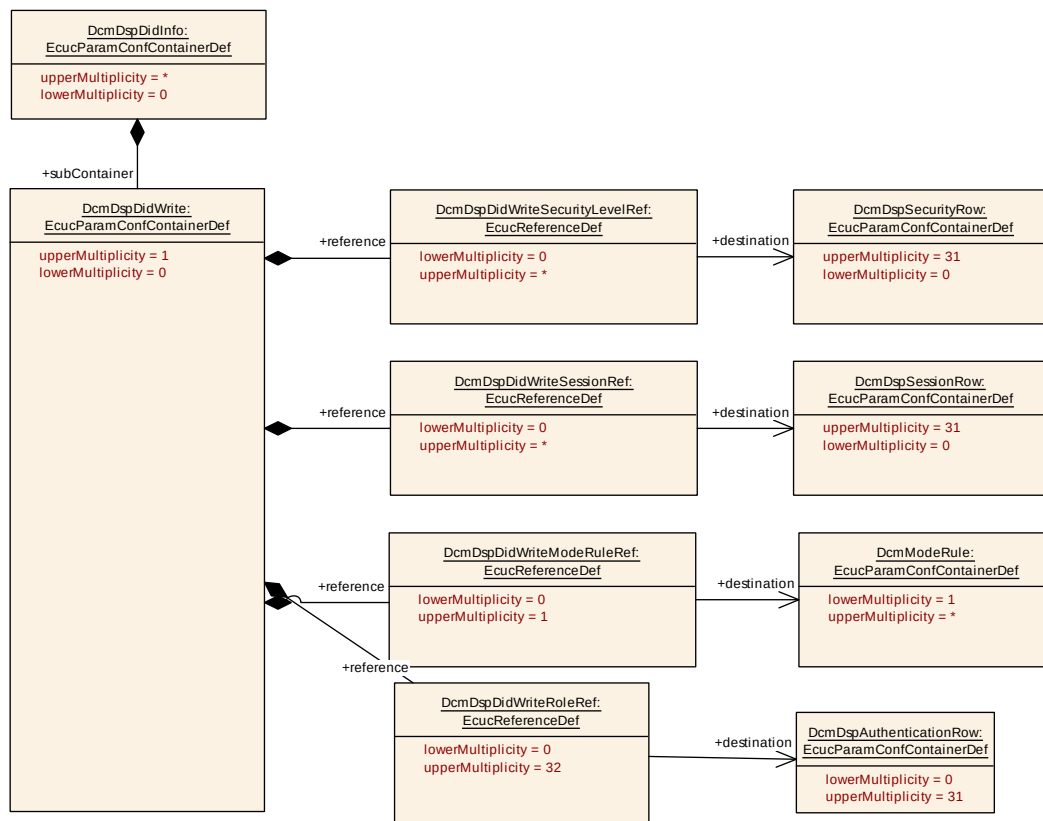


Figure 10.24: DcmDspDidWrite configuration overview

10.2.5.9 DcmDspControlDTCSetting

[ECUC_Dcm_00935] Definition of EcucParamConfContainerDef DcmDspControlDTCSetting

Container Name	DcmDspControlIDTCSetting
Parent Container	DcmDsp
Description	Provide the configuration of the ControlIDTCSetting mechanism.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspControlIDTCSettingReEnableModeRuleRef	0..1	[ECUC_Dcm_00936]

No Included Containers

]

[ECUC_Dcm_00936] Definition of EcucReferenceDef DcmDspControlIDTCSettingReEnableModeRuleRef [

Parameter Name	DcmDspControlIDTCSettingReEnableModeRuleRef		
Parent Container	DcmDspControlIDTCSetting		
Description	Reference to DcmModeRule Mode rule which controls re-enabling of controlIDTCsetting by DCM. The DCM module shall execute a ControlIDTCSetting.Off (call Dem_EnabledDTCSetting()) in case that the referenced mode rule is not fulfilled anymore.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

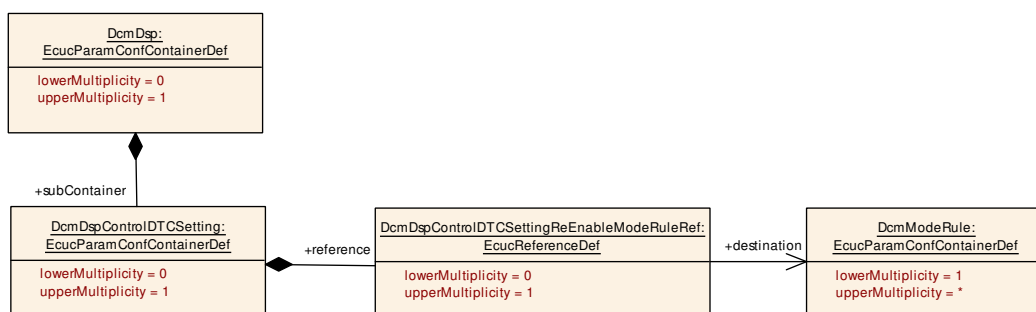


Figure 10.25: DcmDspControlIDTCSetting configuration overview

10.2.5.10 Data elements

10.2.5.10.1 DcmDspData

[ECUC_Dcm_00869] Definition of EcucParamConfContainerDef DcmDspData [

Container Name	DcmDspData
Parent Container	DcmDsp
Description	This container contains the configuration (parameters) of a Data belonging to a DID
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDataByteSize	0..1	[ECUC_Dcm_01106]
DcmDspDataConditionCheckReadFnc	0..1	[ECUC_Dcm_00677]
DcmDspDataConditionCheckReadFncUsed	0..1	[ECUC_Dcm_00955]
DcmDspDataEcuSignal	0..1	[ECUC_Dcm_00825]
DcmDspDataEndianness	0..1	[ECUC_Dcm_00986]
DcmDspDataFreezeCurrentStateFnc	0..1	[ECUC_Dcm_00674]
DcmDspDataGetScalingInfoFnc	0..1	[ECUC_Dcm_00676]
DcmDspDataReadDataLengthFnc	0..1	[ECUC_Dcm_00671]
DcmDspDataReadEcuSignal	0..1	[ECUC_Dcm_00824]
DcmDspDataReadFnc	0..1	[ECUC_Dcm_00669]
DcmDspDataResetToDefaultFnc	0..1	[ECUC_Dcm_00673]
DcmDspDataReturnControlToEcuFnc	0..1	[ECUC_Dcm_00672]
DcmDspDataShortTermAdjustmentFnc	0..1	[ECUC_Dcm_00675]
DcmDspDataType	1	[ECUC_Dcm_00985]
DcmDspDataUsePort	1	[ECUC_Dcm_00713]
DcmDspDataWriteFnc	0..1	[ECUC_Dcm_00670]
DcmDspOdxDataDescription	0..1	[ECUC_Dcm_00988]
DcmDspDataInfoRef	0..1	[ECUC_Dcm_00811]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspDiagnosisScaling	0..1	This container contains the configuration (parameters) of an alternative Diagnosis Representation. Out if this the scaling between Diagnosis and ECU internal representation and vice versa can be calculated.
DcmDspDidDataSupportInfo	0..1	This container defines the supported information.
DcmDspExternalSRDataElement Class	0..1	This container defines the source of data in a provided port which shall be read respectively the target of data in a required port which shall be written. This container shall contain either one DcmSubElementInDataElementInstance OR DcmDataElementInstance OR DcmSubElementInImplDataElementInstance reference.

]

[ECUC_Dcm_01106] Definition of EcucIntegerParamDef DcmDspDataByteSize [

Parameter Name	DcmDspDataByteSize		
Parent Container	DcmDspData , DcmDspDidSignal		
Description	Defines the array length in bytes or the maximum array length for variable datalengths.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00677] Definition of EcucFunctionNameDef DcmDspDataConditionCheckReadFnc [

Parameter Name	DcmDspDataConditionCheckReadFnc		
Parent Container	DcmDspData		
Description	Function name to demand application if the conditions (e.g. System state) to read the DID are correct. (ConditionCheckRead-function). Multiplicity shall be equal to parameter DcmDspDataReadFnc. This parameter is related to the interface Xxx_ConditionCheckRead.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00955] Definition of EcucBooleanParamDef DcmDspDataConditionCheckReadFncUsed [

Parameter Name	DcmDspDataConditionCheckReadFncUsed		
Parent Container	DcmDspData		
Description	This parameter determines if a condition check function is available or not.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00825] Definition of EcucFunctionNameDef DcmDspDataEcuSignal

[

Parameter Name	DcmDspDataEcuSignal		
Parent Container	DcmDspData		
Description	Function name to control the access to a certain ECU Signal by the DCM. (IoHwAb_ Dcm_<symbolic name of ECU signal>-function).		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00986] Definition of EcucEnumerationParamDef DcmDspDataEndianness

Parameter Name	DcmDspDataEndianness		
Parent Container	DcmDspData		
Description	Defines the endianness of the data belonging to a DID in a diagnostic request or response message.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BIG_ENDIAN	Most significant byte shall be stored at the lowest address.	
	LITTLE_ENDIAN	Most significant byte shall be stored at the highest address.	
	OPAQUE	Opaque data endianness	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

[ECUC_Dcm_00674] Definition of EcucFunctionNameDef DcmDspDataFreezeCurrentStateFnc

Parameter Name	DcmDspDataFreezeCurrentStateFnc		
Parent Container	DcmDspData		
Description	Function name to request to application to freeze the current state of an IOControl. (FreezeCurrentState-function). This parameter is related to the interface Xxx_FreezeCurrentState.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00676] Definition of EcucFunctionNameDef DcmDspDataGetScalingInfoFnc

Parameter Name	DcmDspDataGetScalingInfoFnc		
Parent Container	DcmDspData		
Description	Function name to request to application the scaling information of the DID. (GetScaling Information-function). This parameter is related to the interface Xxx_GetScaling Information.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00671] Definition of EcucFunctionNameDef DcmDspDataReadDataLengthFnc

Parameter Name	DcmDspDataReadDataLengthFnc		
Parent Container	DcmDspData		
Description	Function name to request from application the data length of a DID. (ReadData Length-function). This parameter is related to the interface Xxx_ReadDataLength.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00824] Definition of EcucFunctionNameDef DcmDspDataReadEcuSignal

Parameter Name	DcmDspDataReadEcuSignal		
Parent Container	DcmDspData		
Description	Function name for read access to a certain ECU Signal by the DCM. (IoHwAb_Dcm_Read<EcuSignalName>-function). Only relevant if DcmDspDataUsePort==USE_ECU_SIGNAL.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00669] Definition of EcucFunctionNameDef DcmDspDataReadFnc

Parameter Name	DcmDspDataReadFnc		
Parent Container	DcmDspData		
Description	Function name to request from application the data value of a DID. (Read Data-function). This parameter is related to the interface Xxx_ReadData.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00673] Definition of EcucFunctionNameDef DcmDspDataResetToDefaultFnc

Parameter Name	DcmDspDataResetToDefaultFnc		
Parent Container	DcmDspData		
Description	Function name to request to application to reset an IOControl to default value. (ResetToDefault-function). This parameter is related to the interface Xxx_ResetToDefault.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00672] Definition of EcucFunctionNameDef DcmDspDataReturnControlToEcuFnc

Parameter Name	DcmDspDataReturnControlToEcuFnc		
Parent Container	DcmDspData		
Description	Function name to request to application to return control to ECU of an IOControl. (ReturnControlToECU-function). This parameter is related to the interface Xxx_ReturnControlToECU.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00675] Definition of EcucFunctionNameDef DcmDspDataShortTermAdjustmentFnc

Parameter Name	DcmDspDataShortTermAdjustmentFnc		
Parent Container	DcmDspData		
Description	Function name to request to application to adjust the IO signal. (ShortTerm Adjustment-function). This parameter is related to the interface Xxx_ShortTermAdjustment.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00985] Definition of EcucEnumerationParamDef DcmDspDataType

[

Parameter Name	DcmDspDataType	
Parent Container	DcmDspData	
Description	Provide the implementation data type of data belonging to a DID.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	BOOLEAN	Type of the data is boolean.
	DOUBLE	Type of the data is double (64 bit float)
	DOUBLE_N	Type of the data is double (64 bit float) array
	FLOAT	Type of the data is float.
	FLOAT_N	Type of the data is float array.
	SINT16	Type of the data is sint16.
	SINT16_N	Type of the data is sint16 array.
	SINT32	Type of the data is sint32.
	SINT32_N	Type of the data is sint32 array.
	SINT64	Type of the data is sint64
	SINT64_N	Type of the data is sint64 array
	SINT8	Type of the data is sint8.
	SINT8_N	Type of the data is sint8 array.
	UINT16	Type of the data is uint16.
	UINT16_N	Type of the data is uint16 array.
	UINT32	Type of the data is uint32.





	UINT32_N	Type of the data is uint32 array.	
	UINT64	Type of the data is uint64.	
	UINT64_N	Type of the data is uint64 array	
	UINT8	Type of the data is uint8.	
	UINT8_DYN	Type of the data is uint8 array with dynamic length.	
	UINT8_N	Type of the data is uint8 array.	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

┌

[ECUC_Dcm_00713] Definition of EcucEnumerationParamDef DcmDspDataUse Port

Parameter Name	DcmDspDataUsePort		
Parent Container	DcmDspData		
Description	Defines which interface shall be used to access the data.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	USE_DATA_ASYNC_CLIENT_SERVER	The DCM will access the Data using an R-Port requiring a asynchronous ClientServerInterface DataServices_{Data}. The R-Port is named Data Services_{Data} where {Data} is the name of the container DcmDspData.	
	USE_DATA_ASYNC_CLIENT_SERVER_ERROR	The Dcm will access the Data using an R-Port requiring a asynchronous ClientServerInterface DataServices_{Data}. The parameter ErrorCode can be returned to allow the application to trigger a negative response during the operation. The R-Port is named DataServices_{Data} where {Data} is the name of the container DcmDspData.	
	USE_DATA_ASYNC_FNC	The DCM will access the Data using the functions that are defined in parameters of type EcucFunctionNameDef (but without DcmDsp DataReadDataLengthFnc) in the DcmDspData container. DCM_E_PENDING return is allowed. OpStatus is existing as IN parameter.	
	USE_DATA_ASYNC_FNC_ERROR	The DCM will access the Data using the functions that are defined in parameters of type EcucFunctionNameDef (but without DcmDsp DataReadDataLengthFnc) in the DcmDspData container. DCM_E_PENDING return is allowed. OpStatus is existing as IN parameter. The parameter ErrorCode can be returned to allow the application to trigger a negative response during the operation.	





	USE_DATA_ASYNC_FNC_PROXY	The DCM will access the Data using the functions that are defined in parameters of type EcucFunctionNameDef (with DataLength and ErrorCode parameter). DCM_E_PENDING return is allowed. OpStatus is existing as IN parameter. Tags: atp.Status=draft	
	USE_DATA_SENDER_RECEIVER	The DCM will access the Data using an Port requiring a SenderReceiverInterface (with is Service=false) DataServices_{Data}. The Port is namedDataServices_{Data} where {Data} is the name of the container DcmDspData.	
	USE_DATA_SENDER_RECEIVER_AS_SERVICE	The DCM will access the Data using an service Port requiring a SenderReceiverInterface (with is Service=true) DataServices_{Data} . The Port is namedDataServices_{Data} where {Data} is the name of the container DcmDspData.	
	USE_DATA_SYNC_CLIENT_SERVER	The DCM will access the Data using an R-Port requiring a synchronous ClientServerInterface DataServices_{Data}. The R-Port is named Data Services_{Data} where {Data} is the name of the container DcmDspData.	
	USE_DATA_SYNC_FNC	The DCM will access the Data using the functions that are defined in parameters of type EcucFunctionNameDef (but without DcmDsp DataReadDataLengthFnc) in the DcmDspData container. DCM_E_PENDING return value is not allowed and OpStatus parameter is not existing in the prototype.	
	USE_DATA_SYNC_FNC_PROXY	The DCM will access the Data using the functions that are defined in parameters of type EcucFunctionNameDef (with DataLength and ErrorCode parameter). DCM_E_PENDING return value is not allowed and OpStatus parameter is not existing in the prototype. Tags: atp.Status=draft	
	USE_ECU_SIGNAL	The DCM will access the Data using a direct access to IoHwAb	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00670] Definition of EcucFunctionNameDef DcmDspDataWriteFnc

[

Parameter Name	DcmDspDataWriteFnc
Parent Container	DcmDspData
Description	Function name to request application to write the data value of a DID. (Write Data-function). This parameter is related to the interface Xxx_WriteData.
Multiplicity	0..1
Type	EcucFunctionNameDef
Default value	–





Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00988] Definition of EcucAddInfoParamDef DcmDspOdxDataDescription [

Parameter Name	DcmDspOdxDataDescription		
Parent Container	DcmDspData		
Description	Defines additional description for ODX documentation		
Multiplicity	0..1		
Type	EcucAddInfoParamDef		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00811] Definition of EcucReferenceDef DcmDspDataInfoRef [

Parameter Name	DcmDspDataInfoRef		
Parent Container	DcmDspData		
Description	Reference to 1 DcmDspDataInfo		
Multiplicity	0..1		
Type	Reference to DcmDspDataInfo		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

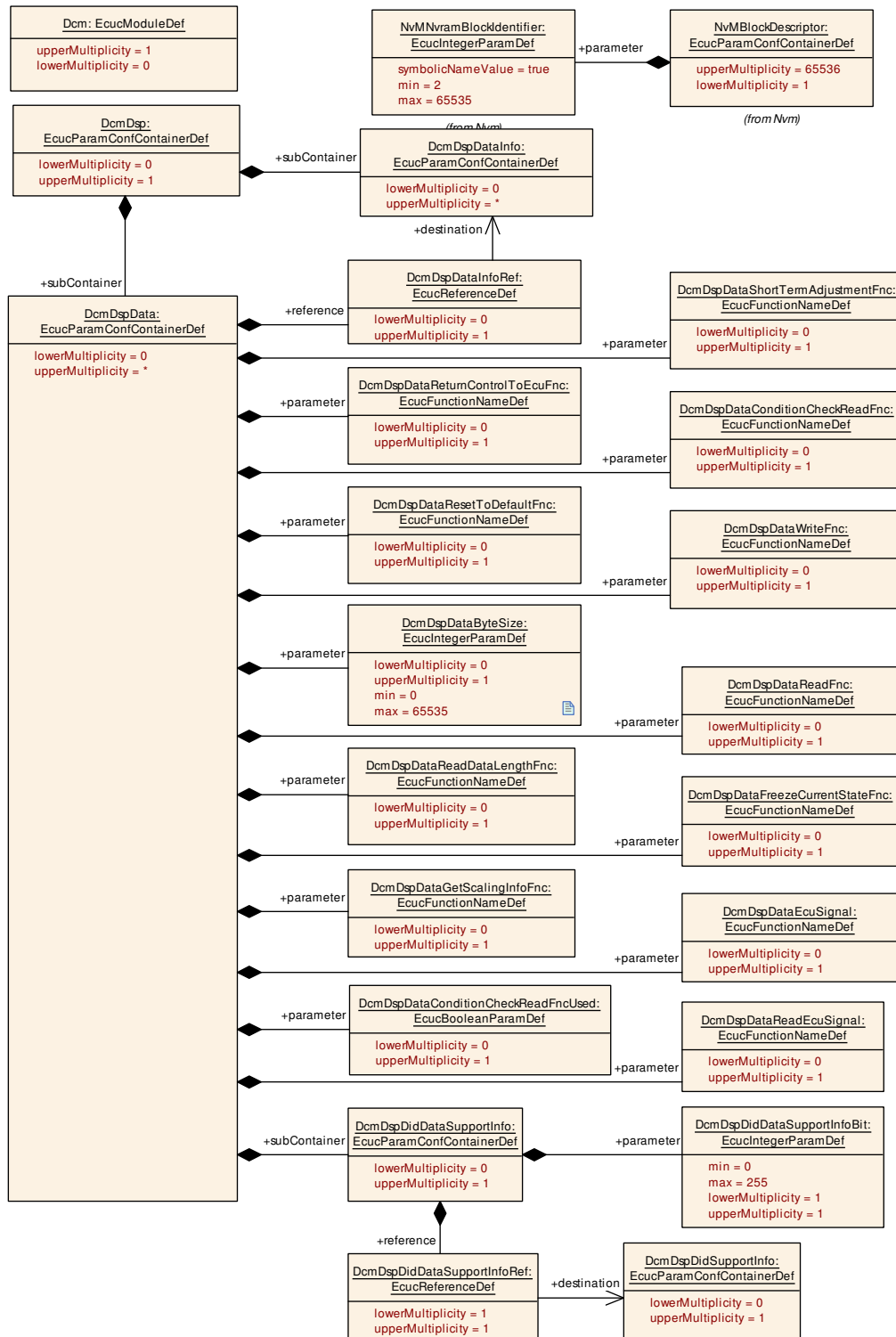


Figure 10.26: DcmDspData1 configuration overview

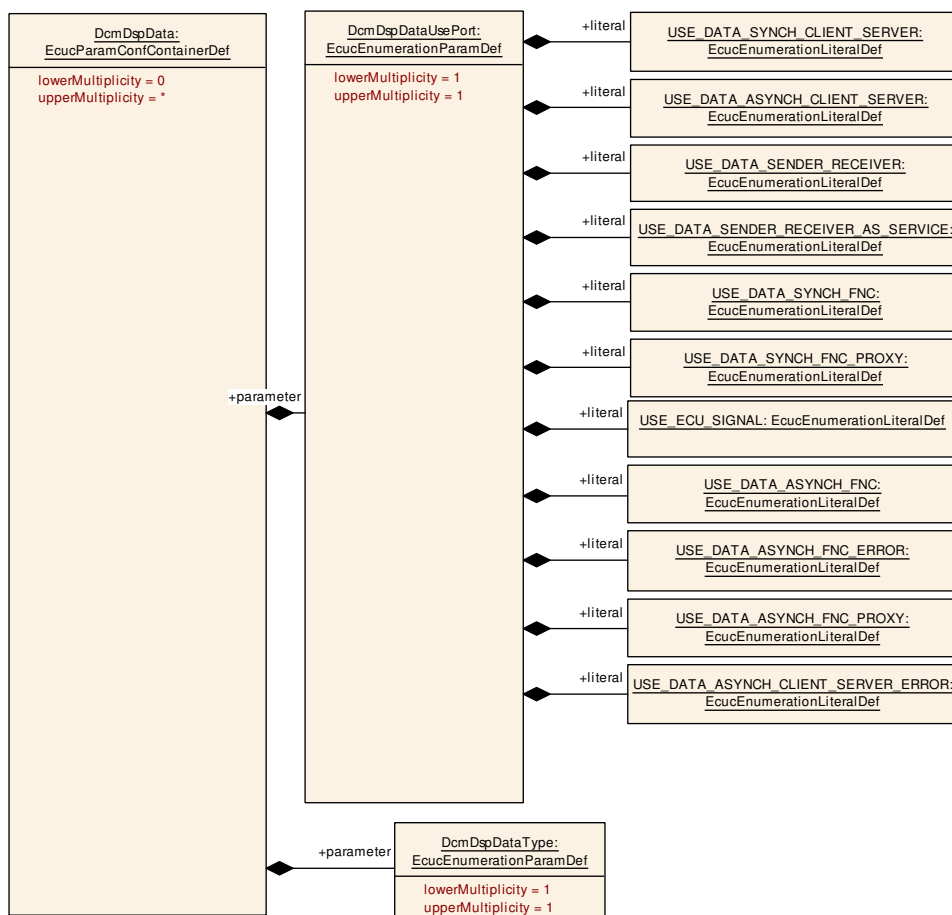


Figure 10.27: DcmDspData2 configuration overview

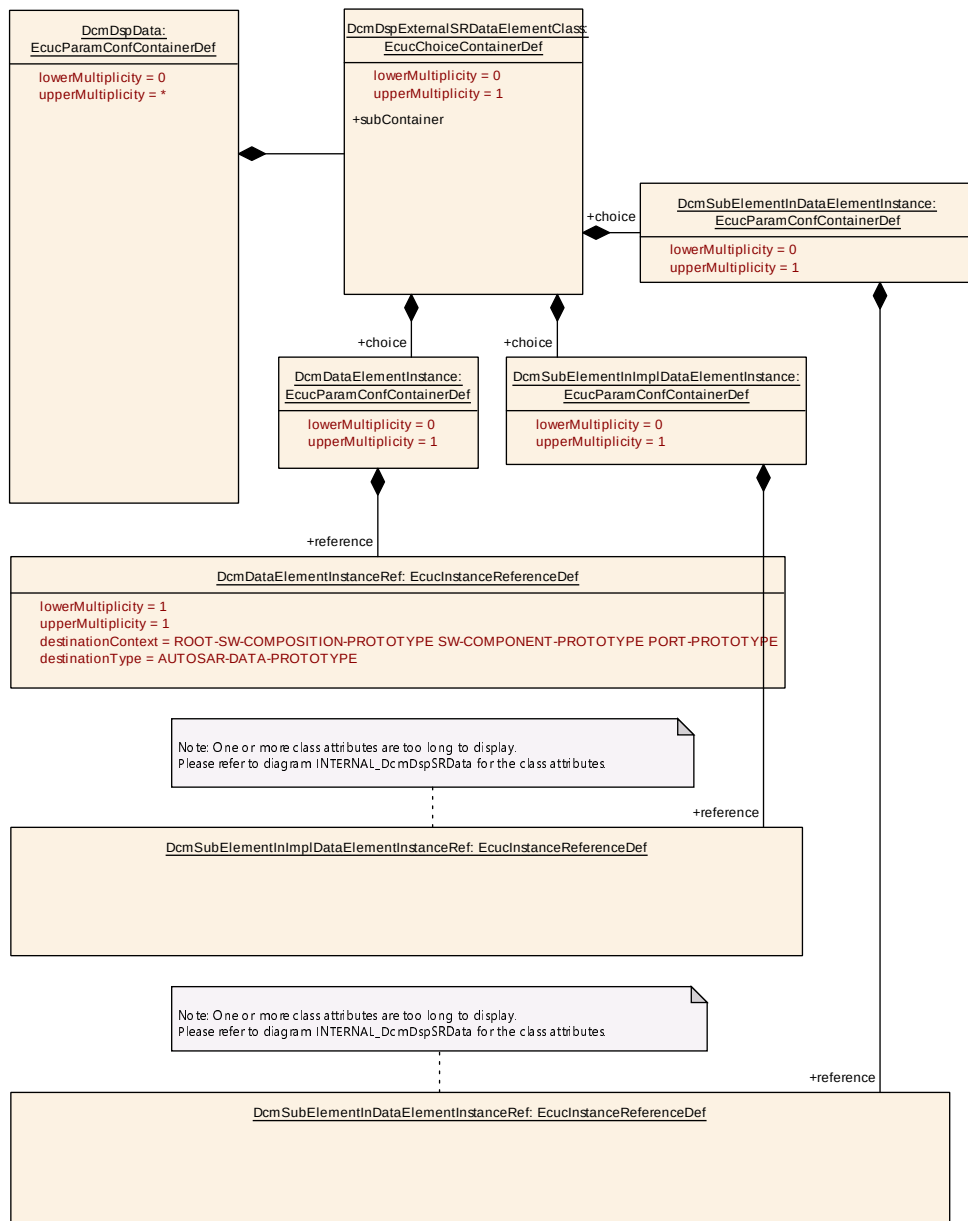


Figure 10.29: DcmDspSRData2 configuration overview

10.2.5.10.2 DcmDspDiagnosisScaling

[ECUC_Dcm_00993] Definition of EcucChoiceContainerDef DcmDspDiagnosisScaling

Choice Container Name	DcmDspDiagnosisScaling
Parent Container	DcmDspData , DcmDspPidService01
Description	This container contains the configuration (parameters) of an alternative Diagnosis Representation. Out if this the scaling between Diagnosis and ECU internal representation and vice versa can be calculated.
Multiplicity	0..1

No Included Parameters

Container Choices		
Container Name	Multiplicity	Dependency
DcmDspAlternativeDataInterface	0..1	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of a Variable DataPrototype in a DataInterface. Additionally a reference to PortInterfaceMapping can be defined which provide already the mapping rules between the Variable DataPrototype in a DataInterface used by the software component (DcmDspExternalSRDataElementClass) and the intended Diagnosis Representation defined by DcmData Element.
DcmDspAlternativeDataType	0..1	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of an ApplicationDataType. Additionally the definition of a text table mapping can be a defined for ApplicationDataTypes with a CompuMethod of category TEXTTABLE and SCALE_LINEAR_AND_TEXTTABLE.
DcmDspAlternativeDiagnosticData Element	0..1	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of Diagnostic Extract.

]

10.2.5.10.3 DcmDspArgumentScaling

[ECUC_Dcm_01062] Definition of EcucChoiceContainerDef DcmDspArgument Scaling [

Choice Container Name	DcmDspArgumentScaling
Parent Container	DcmDspRequestRoutineResultsInSignal , DcmDspRequestRoutineResultsOutSignal , DcmDspStartRoutineInSignal , DcmDspStartRoutineOutSignal , DcmDspStopRoutineInSignal , DcmDspStopRoutineOutSignal
Description	This container contains the configuration (arguments) of an alternative Diagnosis Representation. Out if this the scaling between Diagnosis and ECU internal representation and vice versa can be calculated.
Multiplicity	0..1

No Included Parameters

Container Choices		
Container Name	Multiplicity	Dependency
DcmDspAlternativeArgumentData	0..1	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of a ArgumentDataPrototype.
DcmDspAlternativeDataType	0..1	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of an ApplicationDataType. Additionally the definition of a text table mapping can be a defined for ApplicationDataTypes with a CompuMethod of category TEXTTABLE and SCALE_LINEAR_AND_TEXTTABLE.





Container Choices		
Container Name	Multiplicity	Dependency
DcmDspAlternativeDiagnosticDataElement	0..1	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of Diagnostic Extract.

]

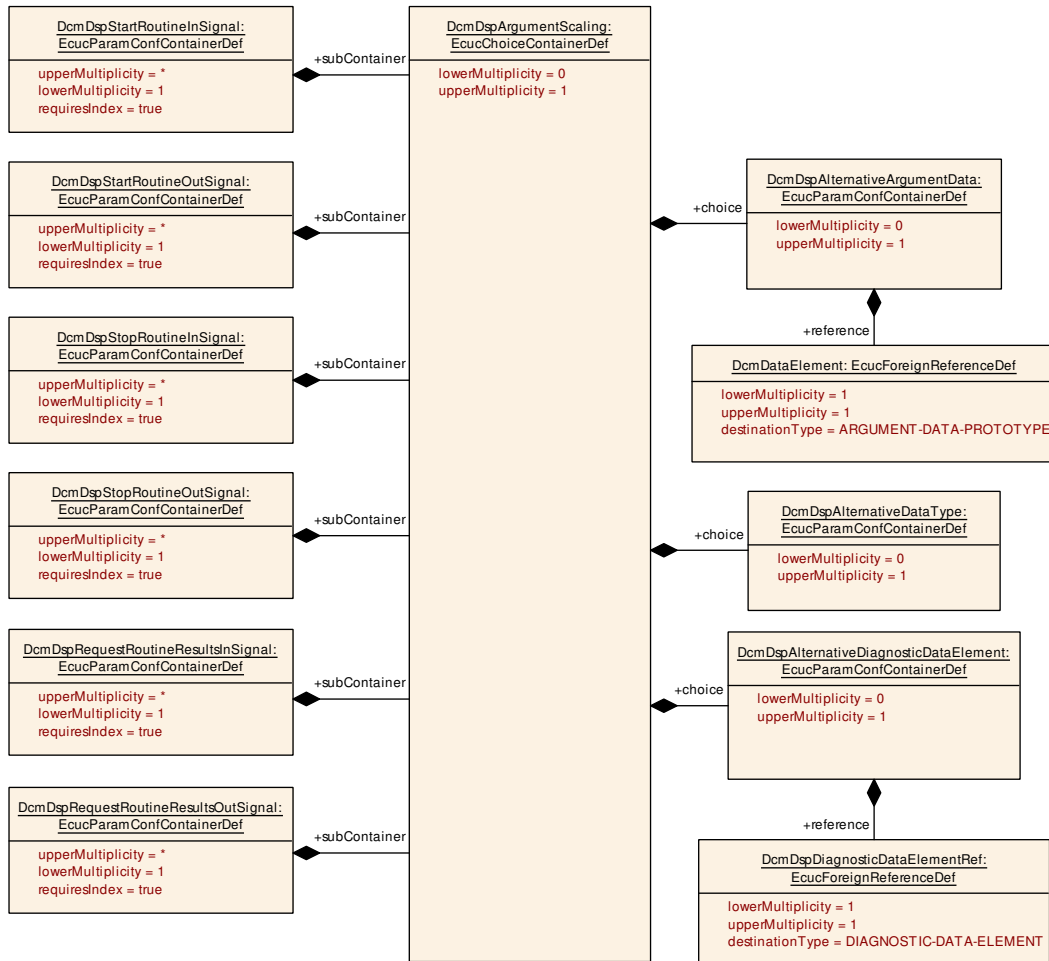


Figure 10.30: DcmDspArgumentScaling configuration overview

10.2.5.10.4 DcmDspAlternativeArgumentData

[ECUC_Dcm_01055] Definition of EcucParamConfContainerDef DcmDspAlternativeArgumentData [

Container Name	DcmDspAlternativeArgumentData
Parent Container	DcmDspArgumentScaling
Description	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of a ArgumentDataPrototype .
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDataElement	1	[ECUC_Dcm_01056]

No Included Containers

]

[[ECUC_Dcm_01056](#)] Definition of EcucForeignReferenceDef DcmDataElement [

Parameter Name	DcmDataElement		
Parent Container	DcmDspAlternativeArgumentData		
Description	Alternative Diagnosis Representation for the data defined by the means of a ArgumentDataPrototype . The CompuMethod of the data type of the referenced ArgumentDataPrototype will be applied to the data type of the ArgumentDataPrototype in the interface used by the Dcm.		
Multiplicity	1		
Type	Foreign reference to ARGUMENT-DATA-PROTOTYPE		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.10.5 DcmDspTextTableMapping

[[ECUC_Dcm_00999](#)] Definition of EcucParamConfContainerDef DcmDspTextTableMapping [

Container Name	DcmDspTextTableMapping
Parent Container	DcmDspAlternativeDataType
Description	The purpose of the DcmDspTextTableMapping is to associate a texttable value defined in the context of the Dcm to a texttable value defined in the context of a CompuMethod referenced by a DataType that shall be taken to create a dataElement in a Sender ReceiverInterface. By this means it is possible to create a primitive version of a TexttableMapping (which can only be applied if a dataElement already exists). In other words, the DcmDspTextTableMapping provides a similar mechanism to the TexttableMapping in a situation where the TexttableMapping cannot be applied since the SenderReceiverInterface for the PortPrototype on the Dcm ServiceComponent does not yet exist.
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDiagnosisRepresentationDataValue	1	[ECUC_Dcm_01001]
DcmDspInternalDataValue	1	[ECUC_Dcm_01000]

No Included Containers

]

[ECUC_Dcm_01001] Definition of EcucIntegerParamDef DcmDspDiagnosisRepresentationDataValue [

Parameter Name	DcmDspDiagnosisRepresentationDataValue		
Parent Container	DcmDspTextTableMapping		
Description	The data value in the diagnosis representation.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 18446744073709551615		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01000] Definition of EcucIntegerParamDef DcmDspInternalData Value [

Parameter Name	DcmDspInternalDataValue		
Parent Container	DcmDspTextTableMapping		
Description	The ECU internal data value.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 18446744073709551615		





Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.10.6 DcmDspAlternativeDataInterface

[ECUC_Dcm_00994] Definition of EcucParamConfContainerDef DcmDspAlternativeDataInterface [

Container Name	DcmDspAlternativeDataInterface
Parent Container	DcmDspDiagnosisScaling
Description	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of a VariableDataPrototype in a DataInterface. Additionally a reference to PortInterfaceMapping can be defined which provide already the mapping rules between the VariableDataPrototype in a DataInterface used by the software component (DcmDspExternalSRDataElementClass) and the intended Diagnosis Representation defined by DcmDataElement.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDataElement	0..1	[ECUC_Dcm_00995]
DcmNestedDataElement	0..1	[ECUC_Dcm_01233]
DcmPortInterfaceMapping	0..1	[ECUC_Dcm_00996]

No Included Containers

]

[ECUC_Dcm_00995] Definition of EcucForeignReferenceDef DcmDataElement [

Parameter Name	DcmDataElement		
Parent Container	DcmDspAlternativeDataInterface		
Description	Alternative Diagnosis Representation for the data defined by the means of a VariableDataPrototype in a DataInterface. The CompuMethod of the data type of the referenced VariableDataPrototype will be applied to the data type of the VariableDataPrototype in the interface used by the Dcm.		
Multiplicity	0..1		
Type	Foreign reference to VARIABLE-DATA-PROTOTYPE		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	





	Post-build time	–	
Dependency			

[ECUC_Dcm_01233] Definition of EcucInstanceReferenceDef DcmNestedDataElement

Parameter Name	DcmNestedDataElement		
Parent Container	DcmDspAlternativeDataInterface		
Description	Alternative Diagnosis Representation for the data defined by the means of a VariableDataPrototype in a DataInterface. The CompuMethod of the data type of the referenced VariableDataPrototype will be applied to the data type of the VariableDataPrototype in the interface used by the Dcm.		
Multiplicity	0..1		
Type	Instance reference to DATA-PROTOTYPE context: VARIABLE-DATA-PROTOTYPE DATA-PROTOTYPE		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00996] Definition of EcucForeignReferenceDef DcmPortInterfaceMapping

Parameter Name	DcmPortInterfaceMapping		
Parent Container	DcmDspAlternativeDataInterface		
Description	Optional reference to PortInterfaceMapping which defines the mapping rules. The PortInterfaceMapping is used to get the DataPrototypeMapping that describes a conversion between the data prototype referenced by DcmDataElement and the data prototype referenced from DcmDspExternalSRDataElementClass.		
Multiplicity	0..1		
Type	Foreign reference to PORT-INTERFACE-MAPPING		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

10.2.5.10.7 DcmDspAlternativeDataType

[ECUC_Dcm_00997] Definition of EcucParamConfContainerDef DcmDspAlternativeDataType

Container Name	DcmDspAlternativeDataType
Parent Container	DcmDspArgumentScaling , DcmDspDiagnosisScaling
Description	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of an ApplicationDataType. Additionally the definition of a text table mapping can be defined for ApplicationDataTypes with a CompuMethod of category TEXTTABLE and SCALE_LINEAR_AND_TEXTTABLE.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmApplicationDataType	1	[ECUC_Dcm_00998]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspTextTableMapping	0..*	The purpose of the DcmDspTextTableMapping is to associate a texttable value defined in the context of the Dcm to a texttable value defined in the context of a CompuMethod referenced by a DataType that shall be taken to create a dataElement in a SenderReceiverInterface. By this means it is possible to create a primitive version of a TexttableMapping (which can only be applied if a dataElement already exists). In other words, the DcmDspTextTableMapping provides a similar mechanism to the TexttableMapping in a situation where the TexttableMapping cannot be applied since the SenderReceiverInterface for the PortPrototype on the Dcm ServiceComponent does not yet exist.

]

[ECUC_Dcm_00998] Definition of EcucForeignReferenceDef DcmApplicationDataType

Parameter Name	DcmApplicationDataType		
Parent Container	DcmDspAlternativeDataType		
Description	Alternative Diagnosis Representation for the data defined by the means of a ApplicationDataType of category VALUE, BOOLEAN or ARRAY. The CompuMethod that applies to the referenced ApplicationDataType in case of category VALUE or BOOLEAN will be applied to the data type of the VariableData Prototype in the interface used by the Dcm.		
Multiplicity	1		
Type	Foreign reference to APPLICATION-DATA-TYPE		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.10.8 DcmDspAlternativeDiagnosticDataElement**[ECUC_Dcm_01084] Definition of EcucParamConfContainerDef DcmDspAlternativeDiagnosticDataElement** [

Container Name	DcmDspAlternativeDiagnosticDataElement
Parent Container	DcmDspArgumentScaling , DcmDspDiagnosisScaling
Description	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of Diagnostic Extract.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDiagnosticDataElementRef	1	[ECUC_Dcm_01085]

No Included Containers

]

[ECUC_Dcm_01085] Definition of EcucForeignReferenceDef DcmDspDiagnosticDataElementRef [

Parameter Name	DcmDspDiagnosticDataElementRef		
Parent Container	DcmDspAlternativeDiagnosticDataElement		
Description	Alternative Diagnosis Representation for the data defined by the means of a Diagnostic DataElement in the Diagnostic Extract. This EcucForeignReference enables the access to all SwDataDefProps, in particular BaseType, CompuMethod and DataConstr The CompuMethod and DataConstr that applies to the referenced DiagnosticData Element will be applied to the data type of the VariableDataPrototype in the interface used by the Dcm. The mapped ImplementationDataType needs to match the given BaseType.		
Multiplicity	1		
Type	Foreign reference to DIAGNOSTIC-DATA-ELEMENT		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.10.9 DcmDataElementInstance**[ECUC_Dcm_01010] Definition of EcucParamConfContainerDef DcmDataElementInstance** [

Container Name	DcmDataElementInstance
Parent Container	DcmDspExternalSRDataElementClass , DcmDspPidService01ExternalSRDataElementClass
Description	Instance Reference to the primitive data in a port where the data element is typed with an ApplicationPrimitiveDataType or an ImplementationDataType.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDataElementInstanceRef	1	[ECUC_Dcm_00991]

No Included Containers

└

[ECUC_Dcm_00991] Definition of EcucInstanceReferenceDef DcmDataElementInstanceRef

Parameter Name	DcmDataElementInstanceRef		
Parent Container	DcmDataElementInstance		
Description	Instance Reference to the primitive or array data which shall be read or written. Supported are VariableDataPrototypes in SenderReceiverInterfaces and NvData Interfaces and ParameterDataPrototypes in ParameterInterfaces (read only). This reference is applicable if the AutosarDataPrototype is typed with a ApplicationPrimitiveDataType of category VALUE or BOOLEAN or ApplicationArrayDataType or if the AutosarDataPrototype is typed with a ImplementationDataType of category VALUE, ARRAY or TYPE_REFERENCE that in turn boils down to VALUE or ARRAY		
Multiplicity	1		
Type	Instance reference to AUTOSAR-DATA-PROTOTYPE context: ROOT-SW-COMPOSITION-PROTOTYPE SW-COMPONENT-PROTOTYPE PORT-PROTOTYPE		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

10.2.5.10.10 DcmDspExternalSRDataElementClass

[ECUC_Dcm_00989] Definition of EcucChoiceContainerDef DcmDspExternalSRDataElementClass

Choice Container Name	DcmDspExternalSRDataElementClass
Parent Container	DcmDspData
Description	This container defines the source of data in a provided port which shall be read respectively the target of data in a required port which shall be written. This container shall contain either one DcmSubElementInDataElementInstance OR DcmDataElementInstance OR DcmSubElementInImplDataElementInstance reference.
Multiplicity	0..1

No Included Parameters

Container Choices		
Container Name	Multiplicity	Dependency
DcmDataElementInstance	0..1	Instance Reference to the primitive data in a port where the data element is typed with an <code>ApplicationPrimitiveDataType</code> or an <code>ImplementationDataType</code> .
DcmSubElementInDataElementInstance	0..1	Instance Reference to the primitive sub-element (at any level) of composite data in a port where the data element is typed with an <code>ApplicationCompositeDataType</code> .
DcmSubElementInImplDataElementInstance	0..1	Instance Reference to the primitive sub-element (at any level) of composite data in a port where the data element is typed with an <code>ImplementationDataType</code> .

]

10.2.5.10.11 DcmDspPidService01ExternalSRDataElementClass

[ECUC_Dcm_01080] Definition of EcucChoiceContainerDef DcmDspPidService01ExternalSRDataElementClass [

Choice Container Name	DcmDspPidService01ExternalSRDataElementClass
Parent Container	DcmDspPidService01
Description	This container defines the source of data in a provided port which shall be read respectively the target of data in a required port which shall be written. This container shall contain either one <code>DcmSubElementInDataElementInstance</code> OR <code>DcmDataElementInstance</code> OR <code>DcmSubElementInImplDataElementInstance</code> reference.
Multiplicity	0..1

No Included Parameters

Container Choices		
Container Name	Multiplicity	Dependency
DcmDataElementInstance	0..1	Instance Reference to the primitive data in a port where the data element is typed with an <code>ApplicationPrimitiveDataType</code> or an <code>ImplementationDataType</code> .
DcmSubElementInDataElementInstance	0..1	Instance Reference to the primitive sub-element (at any level) of composite data in a port where the data element is typed with an <code>ApplicationCompositeDataType</code> .
DcmSubElementInImplDataElementInstance	0..1	Instance Reference to the primitive sub-element (at any level) of composite data in a port where the data element is typed with an <code>ImplementationDataType</code> .

]

10.2.5.10.12 DcmSubElementInDataElementInstance

[ECUC_Dcm_01009] Definition of EcucParamConfContainerDef DcmSubElementInDataElementInstance [

Container Name	DcmSubElementInDataElementInstance
Parent Container	DcmDspExternalSRDataElementClass , DcmDspPidService01ExternalSRDataElementClass
Description	Instance Reference to the primitive sub-element (at any level) of composite data in a port where the data element is typed with an ApplicationCompositeDataType.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmSubElementInDataElementInstanceRef	1	[ECUC_Dcm_00990]

No Included Containers

]

[ECUC_Dcm_00990] Definition of EcucInstanceReferenceDef DcmSubElementInDataElementInstanceRef [

Parameter Name	DcmSubElementInDataElementInstanceRef		
Parent Container	DcmSubElementInDataElementInstance		
Description	Instance Reference to the primitive sub-element (at any level) of composite data in a port which shall be read. Supported are VariableDataPrototypes in SenderReceiver Interfaces and NvDataInterfaces and ParameterDataPrototypes in ParameterInterfaces (read only). This reference is applicable if the AutosarDataPrototype is typed with a ApplicationCompositeDataType.		
Multiplicity	1		
Type	Instance reference to APPLICATION-COMPOSITE-ELEMENT-DATA-PROTOTYPE context: ROOT-SW-COMPOSITION-PROTOTYPE SW-COMPONENT-PROTOTYPE PORT-PROTOTYPE AUTOSAR-DATA-PROTOTYPE APPLICATION-COMPOSITE-ELEMENT-DATA-PROTOTYPE*		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.10.13 DcmSubElementInImplDataElementInstance

[ECUC_Dcm_01011] Definition of EcucParamConfContainerDef DcmSubElementInImplDataElementInstance [

Container Name	DcmSubElementInImplDataElementInstance
Parent Container	DcmDspExternalSRDataElementClass , DcmDspPidService01ExternalSRDataElementClass
Description	Instance Reference to the primitive sub-element (at any level) of composite data in a port where the data element is typed with an ImplementationDataType.





Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmSubElementInImplDataElementInstanceRef	1	[ECUC_Dcm_00992]

No Included Containers

└

[ECUC_Dcm_00992] Definition of EcucInstanceReferenceDef DcmSubElementInImplDataElementInstanceRef

Parameter Name	DcmSubElementInImplDataElementInstanceRef		
Parent Container	DcmSubElementInImplDataElementInstance		
Description	Instance Reference to the primitive sub-element (at any level) of composite data in a port which shall be read. Supported are VariableDataPrototypes in SenderReceiver Interfaces and NvDataInterfaces and ParameterDataPrototypes in ParameterInterfaces (read only). This reference is applicable if the AutosarDataPrototype is typed with a ImplementationDataType of category STRUCTURE or ARRAY. Please note that in case of ARRAY the index attribute in the target reference has to be set to select a single array element.		
Multiplicity	1		
Type	Instance reference to IMPLEMENTATION-DATA-TYPE-ELEMENT context: ROOT-SW-COMPOSITION-PROTOTYPE SW-COMPONENT-PROTOTYPE PORT-PROTOTYPE AUTOSAR-DATA-PROTOTYPE IMPLEMENTATION-DATA-TYPE-ELEMENT*		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

10.2.5.10.14 DcmDspDidDataSupportInfo

[ECUC_Dcm_01104] Definition of EcucParamConfContainerDef DcmDspDidDataSupportInfo

Container Name	DcmDspDidDataSupportInfo
Parent Container	DcmDspData
Description	This container defines the supported information.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDidDataSupportInfoBit	1	[ECUC_Dcm_01097]
DcmDspDidDataSupportInfoRef	1	[ECUC_Dcm_01098]

No Included Containers

]

[[ECUC_Dcm_01097](#)] Definition of EcucIntegerParamDef DcmDspDidDataSupportInfoBit [

Parameter Name	DcmDspDidDataSupportInfoBit		
Parent Container	DcmDspDidDataSupportInfo		
Description	Referenced Bit of the SupportInfo		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[[ECUC_Dcm_01098](#)] Definition of EcucReferenceDef DcmDspDidDataSupportInfoRef [

Parameter Name	DcmDspDidDataSupportInfoRef		
Parent Container	DcmDspDidDataSupportInfo		
Description	Reference to DcmDspDidSupportInfo		
Multiplicity	1		
Type	Reference to DcmDspDidSupportInfo		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

10.2.5.10.15 DcmDspDataInfo

[ECUC_Dcm_00810] Definition of EcucParamConfContainerDef DcmDspDataInfo

Container Name	DcmDspDataInfo		
Parent Container	DcmDsp		
Description	This container contains the configuration (parameters) of one Data.		
Multiplicity	0..*		
Configuration Parameters			
Included Parameters			
Parameter Name	Multiplicity	ECUC ID	
DcmDspDataScalingInfoSize	0..1	[ECUC_Dcm_00611]	
No Included Containers			

[ECUC_Dcm_00611] Definition of EcucIntegerParamDef DcmDspDataScalingInfoSize

Parameter Name	DcmDspDataScalingInfoSize		
Parent Container	DcmDspDataInfo		
Description	If Scaling information service is available for this Data, it provides the size in bytes of the scaling information.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	—		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

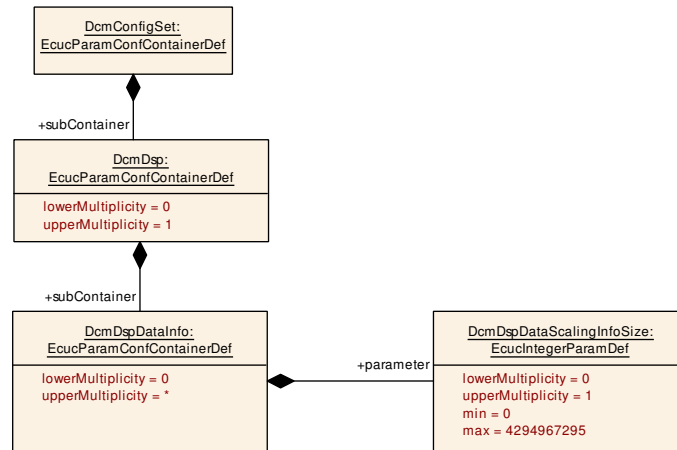


Figure 10.31: DcmDspDataInfo configuration overview

10.2.5.11 DcmDspDidControl

[ECUC_Dcm_00619] Definition of EcucParamConfContainerDef DcmDspDidControl [

Container Name	DcmDspDidControl
Parent Container	DcmDspDidInfo
Description	This container contains the configuration (parameters) of the DID control.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDidControlMask	1	[ECUC_Dcm_01059]
DcmDspDidControlMaskSize	0..1	[ECUC_Dcm_01060]
DcmDspDidFreezeCurrentState	1	[ECUC_Dcm_00624]
DcmDspDidResetToDefault	1	[ECUC_Dcm_00623]
DcmDspDidShortTermAdjustment	1	[ECUC_Dcm_00625]
DcmDspDidControlModeRuleRef	0..1	[ECUC_Dcm_00923]
DcmDspDidControlRoleRef	0..32	[ECUC_Dcm_01143]
DcmDspDidControlSecurityLevelRef	0..*	[ECUC_Dcm_00620]
DcmDspDidControlSessionRef	0..*	[ECUC_Dcm_00621]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspDidControlEnableMask	0..32	The shortname of the container value defines the symbol of the controlMask.

]

[ECUC_Dcm_01059] Definition of EcucEnumerationParamDef DcmDspDidControlMask

Parameter Name	DcmDspDidControlMask		
Parent Container	DcmDspDidControl		
Description	This indicates the presence of "controlEnableMask" in SWC service interfaces and defines how the Dcm treats a service request.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DCM_CONTROLMASK_EXTERNAL	The control enable mask record shall be forwarded within each interface and is handled externally.	
	DCM_CONTROLMASK_INTERNAL	The control enable mask record is handled internally and Dcm controls only the included signals.	
	DCM_CONTROLMASK_NO	No control enable mask handling.	
Default value	DCM_CONTROLMASK_INTERNAL		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	—	
	Post-build time	—	
Dependency			

[ECUC_Dcm_01060] Definition of EcucIntegerParamDef DcmDspDidControlMaskSize

Parameter Name	DcmDspDidControlMaskSize		
Parent Container	DcmDspDidControl		
Description	The value defines the size of the controlEnableMaskRecord in bytes.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967294		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00624] Definition of EcucBooleanParamDef DcmDspDidFreezeCurrentState

Parameter Name	DcmDspDidFreezeCurrentState		
Parent Container	DcmDspDidControl		
Description	This indicates the presence of "FreezeCurrentState".		
Multiplicity	1		





Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00623] Definition of EcucBooleanParamDef DcmDspDidResetToDefault

Parameter Name	DcmDspDidResetToDefault		
Parent Container	DcmDspDidControl		
Description	This indicates the presence of "ResetToDefault".		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00625] Definition of EcucBooleanParamDef DcmDspDidShortTermAdjustment

Parameter Name	DcmDspDidShortTermAdjustment		
Parent Container	DcmDspDidControl		
Description	This indicates the presence of "ShortTermAdjustment".		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00923] Definition of EcucReferenceDef DcmDspDidControlModeRuleRef

Parameter Name	DcmDspDidControlModeRuleRef		
Parent Container	DcmDspDidControl		
Description	Reference to DcmModeRule Mode rule which controls this DID. If there is no reference, no check of the mode rule shall be done.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_01143] Definition of EcucReferenceDef DcmDspDidControlRoleRef

Parameter Name	DcmDspDidControlRoleRef		
Parent Container	DcmDspDidControl		
Description	Reference to DcmDspAuthenticationRow that defines a role in that this IO can be controlled.		
Multiplicity	0..32		
Type	Reference to DcmDspAuthenticationRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00620] Definition of EcucReferenceDef DcmDspDidControlSecurityLevelRef

Parameter Name	DcmDspDidControlSecurityLevelRef		
Parent Container	DcmDspDidControl		
Description	Reference to DcmDspSecurityRow Security levels allowed to control this DID. If there is no reference, no check of security level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSecurityRow		





Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00621] Definition of EcucReferenceDef DcmDspDidControlSession Ref [

Parameter Name	DcmDspDidControlSessionRef		
Parent Container	DcmDspDidControl		
Description	Reference to DcmDspSessionRow Sessions allowed to control this DID. If there is no reference, no check of session level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSessionRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

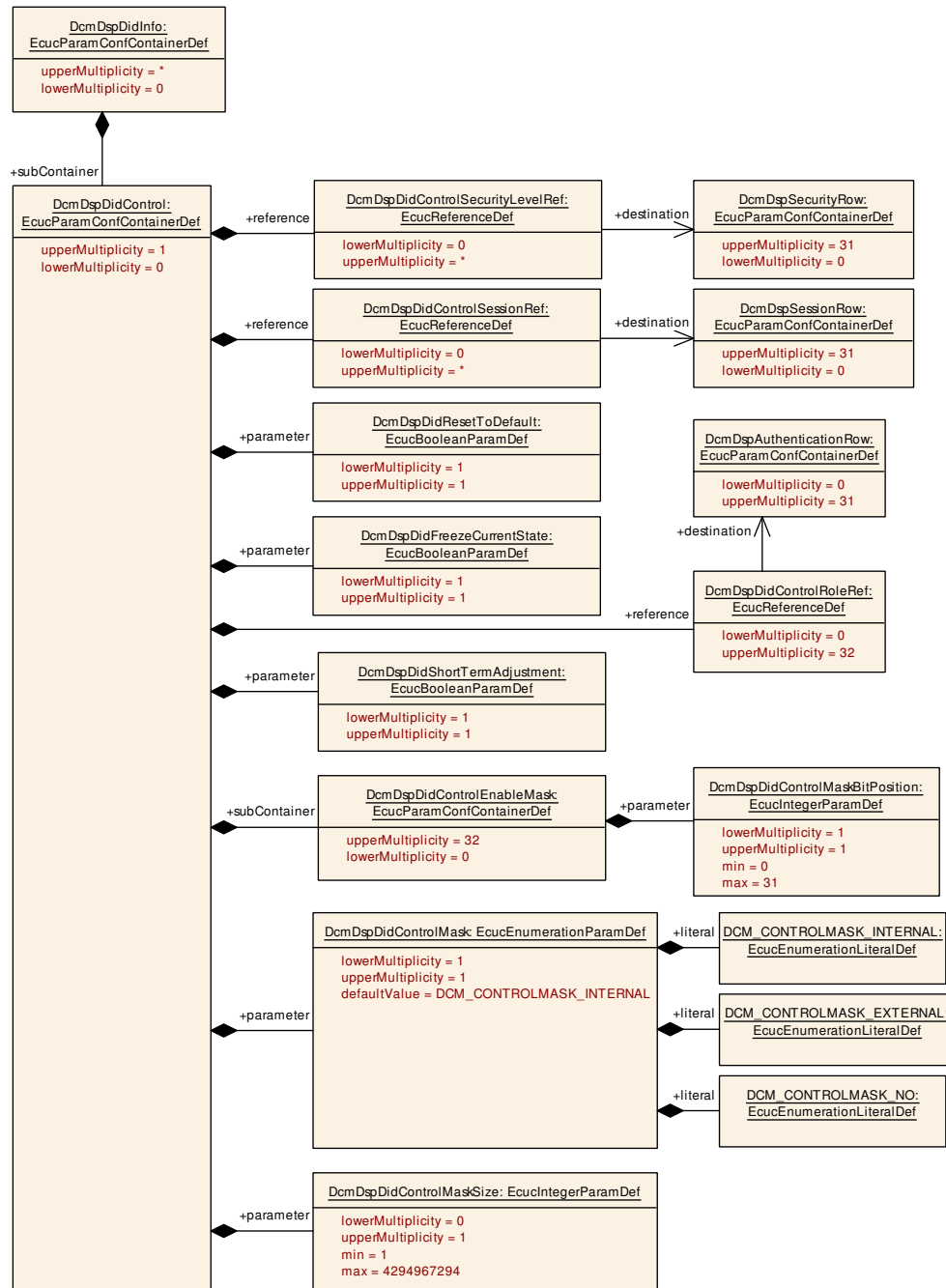


Figure 10.32: DcmDspDidControl configuration overview

10.2.5.12 DcmDspDidControlEnableMask

[ECUC_Dcm_01057] Definition of EcucParamConfContainerDef DcmDspDidControlEnableMask

Container Name	DcmDspDidControlEnableMask
Parent Container	DcmDspDidControl
Description	The shortname of the container value defines the symbol of the controlMask.
Multiplicity	0..32
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspDidControlMaskBitPosition	1	[ECUC_Dcm_01058]

No Included Containers

]

[ECUC_Dcm_01058] Definition of EcucIntegerParamDef DcmDspDidControlMaskBitPosition [

Parameter Name	DcmDspDidControlMaskBitPosition		
Parent Container	DcmDspDidControlEnableMask		
Description	Defines the position of the bit in the controlMask starting. The counting order is from the most significant bit (MSB first) to least significant bit. A value of 0 identifies the most significant bit in the MSB of the control enable mask. A value of 0 represents also the first controlState parameter. The bit endianness is identical to the controlMask in UDS. The number of possible and available bits is given by DcmDspDidControlMaskSize.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 31		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.13 Ecu Reset

10.2.5.13.1 DcmDspEcuReset

[ECUC_Dcm_01111] Definition of EcucParamConfContainerDef DcmDspEcuReset [

Container Name	DcmDspEcuReset
Parent Container	DcmDsp
Description	This container contains the configuration for DcmDspEcuReset service





Multiplicity	0..1	
Configuration Parameters		
No Included Parameters		
Included Containers		
Container Name	Multiplicity	Dependency
DcmDspEcuResetRow	1..*	This container contains the configuration for each DcmDspEcu Reset subservice.

└

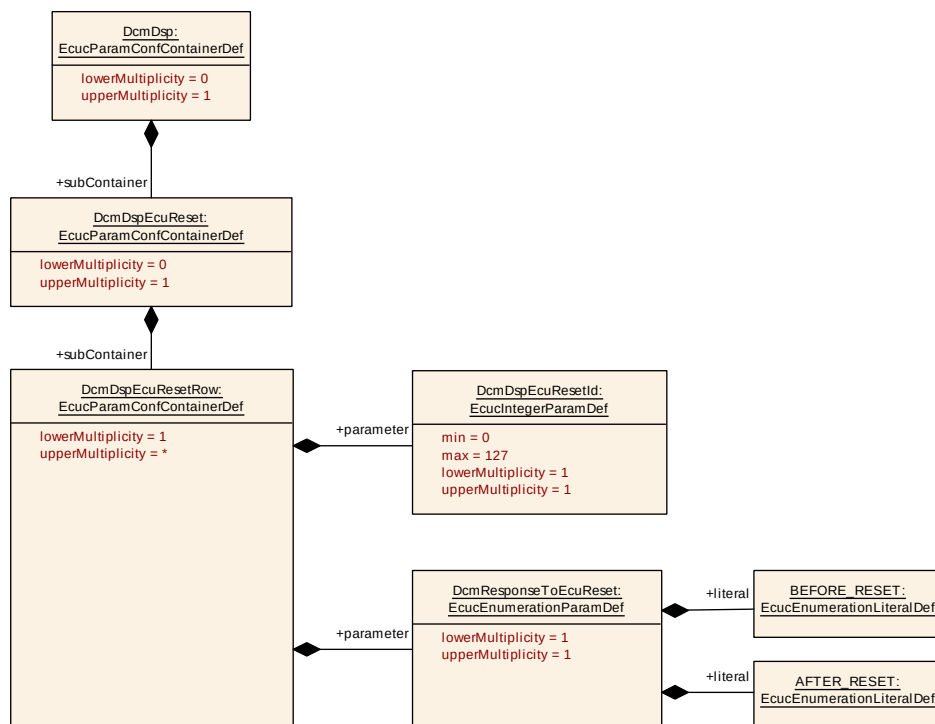


Figure 10.33: DcmDspEcuReset configuration overview

10.2.5.13.2 DcmDspEcuResetRow

[ECUC_Dcm_01112] Definition of EcucParamConfContainerDef DcmDspEcuResetRow

Container Name	DcmDspEcuResetRow
Parent Container	DcmDspEcuReset
Description	This container contains the configuration for each DcmDspEcuReset subservice.
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspEcuResetId	1	[ECUC_Dcm_01113]
DcmResponseToEcuReset	1	[ECUC_Dcm_01039]

No Included Containers

]

[ECUC_Dcm_01113] Definition of EcucIntegerParamDef DcmDspEcuResetId [

Parameter Name	DcmDspEcuResetId		
Parent Container	DcmDspEcuResetRow		
Description	Defines the subfunction ID		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 127		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01039] Definition of EcucEnumerationParamDef DcmResponseToEcuReset [

Parameter Name	DcmResponseToEcuReset		
Parent Container	DcmDspEcuResetRow		
Description	Defines the answer to EcuReset service should come: Before or after the reset.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	AFTER_RESET		Answer to EcuReset service should come after the reset.
	BEFORE_RESET		Answer to EcuReset service should come before the reset.
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.14 Memory

10.2.5.14.1 DcmDspMemory

[ECUC_Dcm_00784] Definition of EcucParamConfContainerDef DcmDspMemory

Container Name	DcmDspMemory	
Parent Container	DcmDsp	
Description	This container contains the configuration of the memory access.	
Multiplicity	0..1	
Configuration Parameters		
No Included Parameters		
Included Containers		
Container Name	Multiplicity	Dependency
DcmDspAddressAndLengthFormatIdentifier	0..1	This container contains the configuration of the supported AddressAndLengthFormatIdentifiers for memory access.
DcmDspMemoryIdInfo	1..*	Provides the value of memory identifier used to select the desired memory device. This container contains the configuration of the memory access requested through diagnostic services : ReadMemoryByAddress, WriteMemoryByAddress, DynamicallyDefineDataIdentifier, RequestDownload and RequestUpload.

10.2.5.14.2 DcmDspMemoryTransfer

[ECUC_Dcm_01132] Definition of EcucParamConfContainerDef DcmDspMemoryTransfer

Container Name	DcmDspMemoryTransfer		
Parent Container	DcmDsp		
Description	This container contains the configuration of the memory transfer.		
Multiplicity	0..1		
Configuration Parameters			
Included Parameters			
Parameter Name		Multiplicity	ECUC ID
DcmDspMemoryTransferUsePort		1	[ECUC_Dcm_01133]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspAddressAndLengthFormatIdentifier	0..1	This container contains the configuration of the supported AddressAndLengthFormatIdentifiers for memory access.
DcmDspMemoryTransferIdInfo	1..*	Provides the value of memory identifier used to select the desired memory device This container contains the configuration of the memory access requested through diagnostic services : RequestDownload, RequestUpload, TransferData, RequestTransferExit

]

[ECUC_Dcm_01133] Definition of EcucBooleanParamDef DcmDspMemoryTransferUsePort [

Parameter Name	DcmDspMemoryTransferUsePort		
Parent Container	DcmDspMemoryTransfer		
Description	If this parameter is set to true, the Dcm uses a port requiring a PortInterface Upload Download. If the parameter is false, the DCM uses the according C-API callouts.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.14.3 DcmDspAddressAndLengthFormatIdentifier

[ECUC_Dcm_00963] Definition of EcucParamConfContainerDef DcmDspAddressAndLengthFormatIdentifier [

Container Name	DcmDspAddressAndLengthFormatIdentifier		
Parent Container	DcmDspMemory , DcmDspMemoryTransfer		
Description	This container contains the configuration of the supported AddressAndLengthFormatIdentifiers for memory access.		
Multiplicity	0..1		
Configuration Parameters			

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspSupportedAddressAndLengthFormatIdentifier	1..*	[ECUC_Dcm_00964]

No Included Containers

]

[ECUC_Dcm_00964] Definition of EcucIntegerParamDef DcmDspSupportedAddressAndLengthFormatIdentifier [

Parameter Name	DcmDspSupportedAddressAndLengthFormatIdentifier		
Parent Container	DcmDspAddressAndLengthFormatIdentifier		
Description	This parameter defines the supported AddressAndLengthFormatIdentifier of the request message.		
Multiplicity	1..*		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

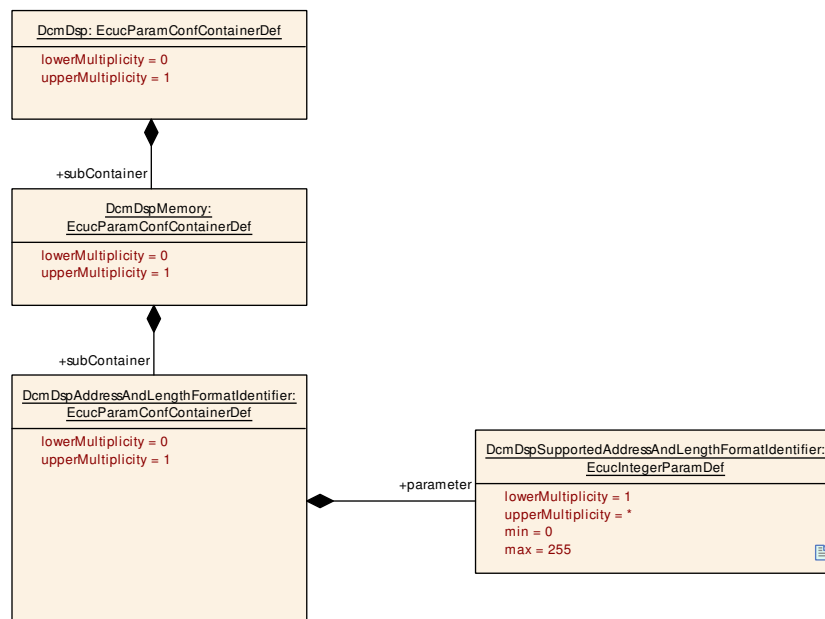


Figure 10.34: DcmDspMemoryFormatIdentifier configuration overview

10.2.5.14.4 DcmDspMemoryIdInfo

[ECUC_Dcm_00911] Definition of EcucParamConfContainerDef DcmDspMemoryIdInfo [

Container Name	DcmDspMemoryIdInfo
Parent Container	DcmDspMemory
Description	Provides the value of memory identifier used to select the desired memory device. This container contains the configuration of the memory access requested through diagnostic services : ReadMemoryByAddress, WriteMemoryByAddress, Dynamically DefineDataIdentifier, RequestDownload and RequestUpload.
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspMemoryIdValue	0..1	[ECUC_Dcm_00913]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspReadMemoryRangeByLabelInfo	0..*	Provides a memory range allowed for reading via labels (lower and higher address configured as strings).
DcmDspReadMemoryRangeInfo	0..*	Provides the range of memory address allowed for reading
DcmDspWriteMemoryRangeByLabelInfo	0..*	Provides a memory range allowed for writing via labels (lower and higher address configured as strings).
DcmDspWriteMemoryRangeInfo	0..*	Provides the range of memory address allowed for writing.

]

[ECUC_Dcm_00913] Definition of EcucIntegerParamDef DcmDspMemoryIdValue

[

Parameter Name	DcmDspMemoryIdValue		
Parent Container	DcmDspMemoryIdInfo		
Description	Value of the memory device identifier used. Each DcmDspMemoryIdInfo should have a unique ID. The MemoryIdValue is retrieved from the request messages (RMBA, WMBA, RD, RU, DDDI) according to ISO-14229-1 with the most significant byte of the request parameter memoryAddress.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

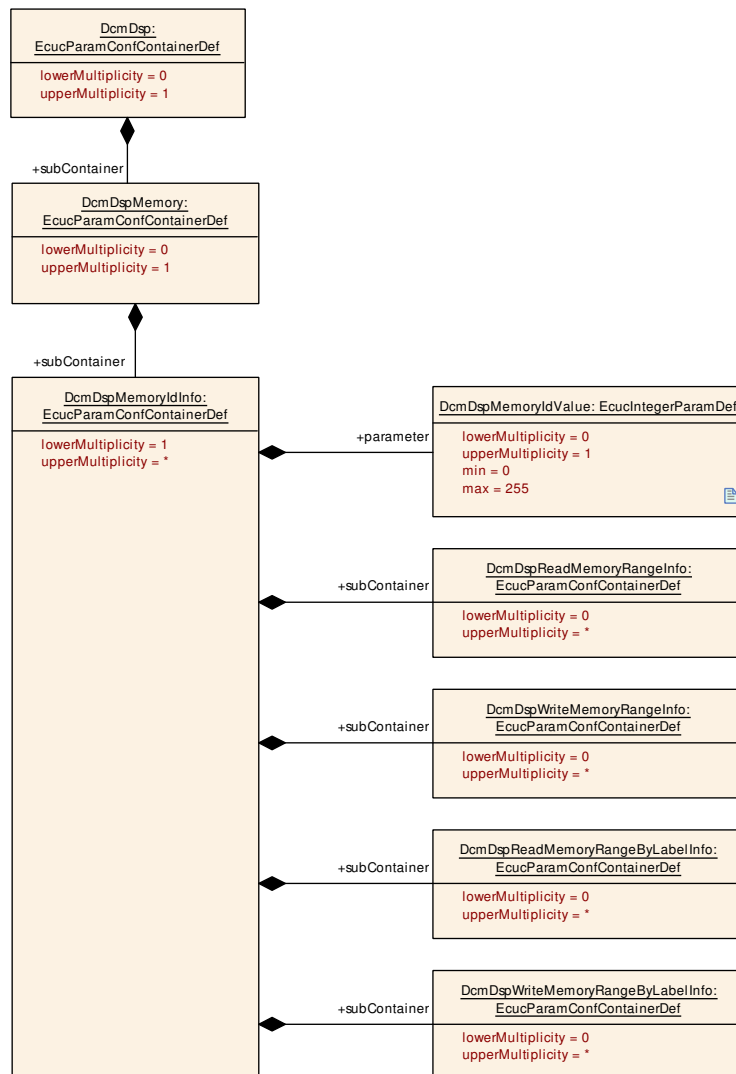


Figure 10.35: DcmDspMemoryIdInfo configuration overview

10.2.5.14.5 DcmDspMemoryTransferIdInfo

[ECUC_Dcm_01137] Definition of EcucParamConfContainerDef DcmDspMemoryTransferIdInfo [

Container Name	DcmDspMemoryTransferIdInfo
Parent Container	DcmDspMemoryTransfer
Description	Provides the value of memory identifier used to select the desired memory device This container contains the configuration of the memory access requested through diagnostic services : RequestDownload, RequestUpload, TransferData, Request TransferExit
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspMemoryIdValue	0..1	[ECUC_Dcm_01138]

No Included Containers

]

[ECUC_Dcm_01138] Definition of EcucIntegerParamDef DcmDspMemoryIdValue

[

Parameter Name	DcmDspMemoryIdValue		
Parent Container	DcmDspMemoryTransferIdInfo		
Description	Value of the memory device identifier used. Each DcmDspMemoryIdInfo should have a unique ID. The MemoryIdValue is retrieved from the request messages (RMBA, WMBA, RD, RU, DDDI) according to ISO-14229-1 with the most significant byte of the request parameter memoryAddress.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

10.2.5.14.6 DcmDspReadMemoryRangeByLabelInfo

[ECUC_Dcm_01068] Definition of EcucParamConfContainerDef DcmDspReadMemoryRangeByLabelInfo

[

Container Name	DcmDspReadMemoryRangeByLabelInfo
Parent Container	DcmDspMemoryIdInfo
Description	Provides a memory range allowed for reading via labels (lower and higher address configured as strings).
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspReadMemoryRangeByLabelHigh	1	[ECUC_Dcm_01070]
DcmDspReadMemoryRangeByLabelLow	1	[ECUC_Dcm_01069]
DcmDspReadMemoryRangeModeRuleRef	0..1	[ECUC_Dcm_01072]
DcmDspReadMemoryRangeSecurityLevelRef	0..*	[ECUC_Dcm_01071]
DcmDspReadMemoryRangeSessionLevelRef	0..*	[ECUC_Dcm_01088]

No Included Containers

]

[ECUC_Dcm_01070] Definition of EcucStringParamDef DcmDspReadMemoryRangeByLabelHigh [

Parameter Name	DcmDspReadMemoryRangeByLabelHigh		
Parent Container	DcmDspReadMemoryRangeByLabelInfo		
Description	High memory address as label (string) of a range allowed for reading.		
Multiplicity	1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	[a-zA-Z0-9_]([a-zA-Z0-9_])*		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01069] Definition of EcucStringParamDef DcmDspReadMemoryRangeByLabelLow [

Parameter Name	DcmDspReadMemoryRangeByLabelLow		
Parent Container	DcmDspReadMemoryRangeByLabelInfo		
Description	Low memory address as label (string) of a range allowed for reading.		
Multiplicity	1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	[a-zA-Z0-9_]([a-zA-Z0-9_])*		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01072] Definition of EcucReferenceDef DcmDspReadMemoryRangeModeRuleRef [

Parameter Name	DcmDspReadMemoryRangeModeRuleRef		
Parent Container	DcmDspReadMemoryRangeByLabelInfo		
Description	Reference to DcmModeRule Mode rule which controls read access on this memory address. If there is no reference, no check of the mode rule shall be done.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01071] Definition of EcucReferenceDef DcmDspReadMemoryRangeSecurityLevelRef [

Parameter Name	DcmDspReadMemoryRangeSecurityLevelRef		
Parent Container	DcmDspReadMemoryRangeByLabelInfo		
Description	Link to the Security Access Levels needed for read access on this memory address. If there is no reference, no check of security level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSecurityRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01088] Definition of EcucReferenceDef DcmDspReadMemoryRangeSessionLevelRef [

Parameter Name	DcmDspReadMemoryRangeSessionLevelRef		
Parent Container	DcmDspReadMemoryRangeByLabelInfo		
Description	Link to the session level needed for access to this memory address range. If there is no reference, no check of session level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSessionRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

10.2.5.14.7 DcmDspReadMemoryRangeInfo**[ECUC_Dcm_00785] Definition of EcucParamConfContainerDef DcmDspReadMemoryRangeInfo** [

Container Name	DcmDspReadMemoryRangeInfo
Parent Container	DcmDspMemoryIdInfo
Description	Provides the range of memory address allowed for reading
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspReadMemoryRangeHigh	1	[ECUC_Dcm_00787]
DcmDspReadMemoryRangeLow	1	[ECUC_Dcm_00786]
DcmDspReadMemoryRangeModeRuleRef	0..1	[ECUC_Dcm_00920]
DcmDspReadMemoryRangeSecurityLevelRef	0..*	[ECUC_Dcm_00788]
DcmDspReadMemoryRangeSessionLevelRef	0..*	[ECUC_Dcm_01086]

No Included Containers

]

[ECUC_Dcm_00787] Definition of EcucIntegerParamDef DcmDspReadMemoryRangeHigh

Parameter Name	DcmDspReadMemoryRangeHigh		
Parent Container	DcmDspReadMemoryRangeInfo		
Description	High memory address of a range allowed for reading		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967294		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_00786] Definition of EcucIntegerParamDef DcmDspReadMemoryRangeLow

Parameter Name	DcmDspReadMemoryRangeLow		
Parent Container	DcmDspReadMemoryRangeInfo		
Description	Low memory address of a range allowed for reading		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967294		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_00920] Definition of EcucReferenceDef DcmDspReadMemoryRangeModeRuleRef

Parameter Name	DcmDspReadMemoryRangeModeRuleRef		
Parent Container	DcmDspReadMemoryRangeInfo		
Description	Reference to DcmModeRule Mode rule which controls read access on this memory address. If there is no reference, no check of the mode rule shall be done.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		





Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00788] Definition of EcucReferenceDef DcmDspReadMemoryRangeSecurityLevelRef [

Parameter Name	DcmDspReadMemoryRangeSecurityLevelRef		
Parent Container	DcmDspReadMemoryRangeInfo		
Description	Link to the Security Access Levels needed for read access on this memory address. If there is no reference, no check of security level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSecurityRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01086] Definition of EcucReferenceDef DcmDspReadMemoryRangeSessionLevelRef [

Parameter Name	DcmDspReadMemoryRangeSessionLevelRef		
Parent Container	DcmDspReadMemoryRangeInfo		
Description	Link to the session level needed for access to this memory address range. If there is no reference, no check of session level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSessionRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	





Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

10.2.5.14.8 DcmDspWriteMemoryRangeByLabelInfo

[ECUC_Dcm_01073] Definition of EcucParamConfContainerDef DcmDspWriteMemoryRangeByLabelInfo

Container Name	DcmDspWriteMemoryRangeByLabelInfo
Parent Container	DcmDspMemoryIdInfo
Description	Provides a memory range allowed for writing via labels (lower and higher address configured as strings).
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspWriteMemoryRangeByLabelHigh	1	[ECUC_Dcm_01075]
DcmDspWriteMemoryRangeByLabelLow	1	[ECUC_Dcm_01074]
DcmDspWriteMemoryRangeModeRuleRef	0..1	[ECUC_Dcm_01077]
DcmDspWriteMemoryRangeSecurityLevelRef	0..*	[ECUC_Dcm_01076]
DcmDspWriteMemoryRangeSessionLevelRef	0..*	[ECUC_Dcm_01089]

No Included Containers

└

[ECUC_Dcm_01075] Definition of EcucStringParamDef DcmDspWriteMemoryRangeByLabelHigh

Parameter Name	DcmDspWriteMemoryRangeByLabelHigh		
Parent Container	DcmDspWriteMemoryRangeByLabelInfo		
Description	High memory address as label (string) of a range allowed for writing.		
Multiplicity	1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	[a-zA-Z0-9_]([a-zA-Z0-9\._])*		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	





Dependency	

]

[ECUC_Dcm_01074] Definition of EcucStringParamDef DcmDspWriteMemoryRangeByLabelLow [

Parameter Name	DcmDspWriteMemoryRangeByLabelLow		
Parent Container	DcmDspWriteMemoryRangeByLabelInfo		
Description	Low memory address as label (string) of a range allowed for writing.		
Multiplicity	1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	[a-zA-Z0-9_]([a-zA-Z0-9\._])*		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01077] Definition of EcucReferenceDef DcmDspWriteMemoryRangeModeRuleRef [

Parameter Name	DcmDspWriteMemoryRangeModeRuleRef		
Parent Container	DcmDspWriteMemoryRangeByLabelInfo		
Description	Reference to DcmModeRule Mode rule which controls write access on this memory address. If there is no reference, no check of the mode rule shall be done.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01076] Definition of EcucReferenceDef DcmDspWriteMemoryRangeSecurityLevelRef [

Parameter Name	DcmDspWriteMemoryRangeSecurityLevelRef		
Parent Container	DcmDspWriteMemoryRangeByLabelInfo		
Description	Link to the Security Access Levels needed for write access on this memory address. If there is no reference, no check of security level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSecurityRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01089] Definition of EcucReferenceDef DcmDspWriteMemoryRangeSessionLevelRef [

Parameter Name	DcmDspWriteMemoryRangeSessionLevelRef		
Parent Container	DcmDspWriteMemoryRangeByLabelInfo		
Description	Link to the session level needed for access to this memory address range. If there is no reference, no check of session level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSessionRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

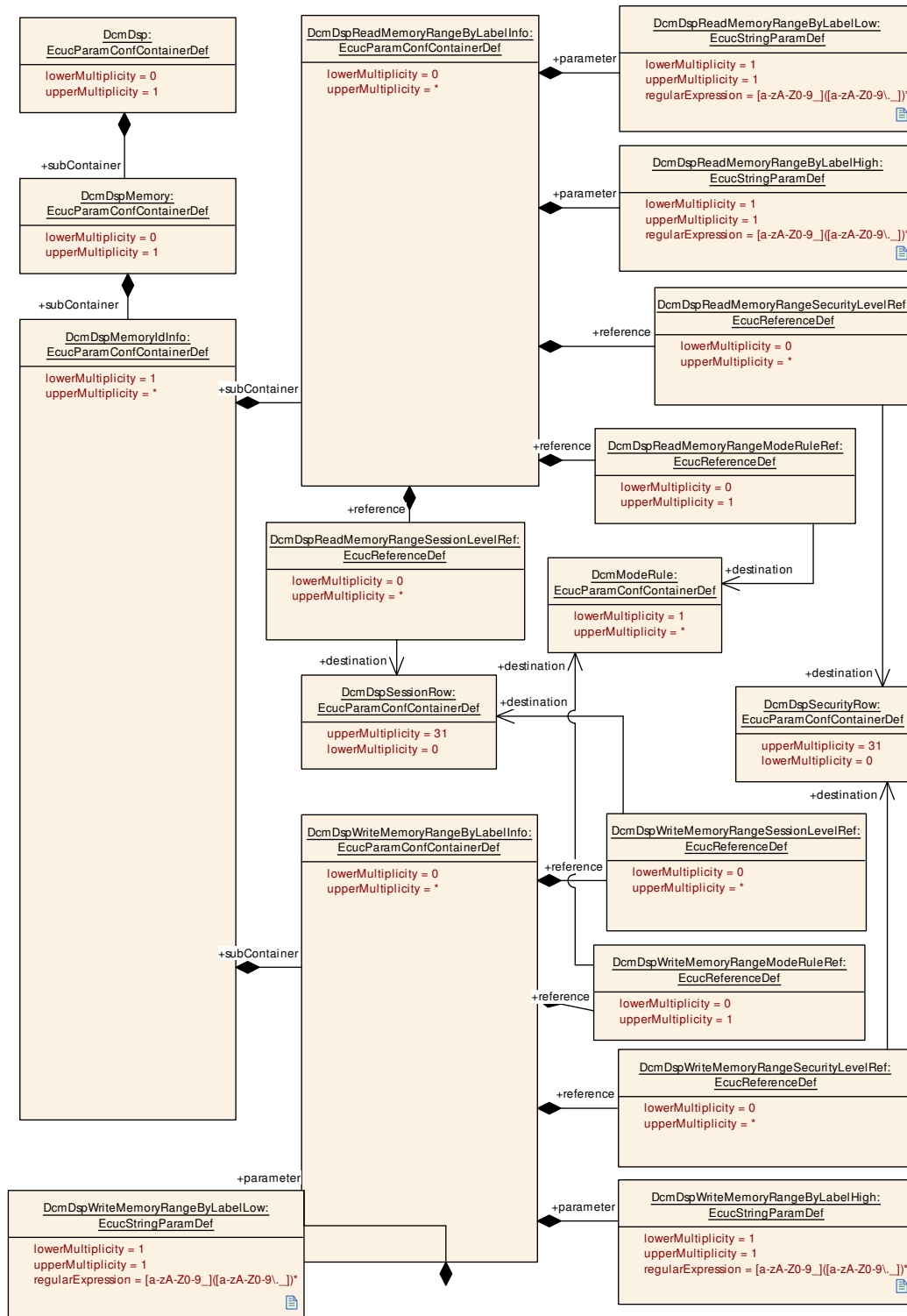


Figure 10.36: DcmDspMemoryReadWriteByLabel configuration overview

10.2.5.14.9 DcmDspWriteMemoryRangeInfo

[ECUC_Dcm_00789] Definition of EcucParamConfContainerDef DcmDspWriteMemoryRangeInfo

Container Name	DcmDspWriteMemoryRangeInfo
Parent Container	DcmDspMemoryIdInfo
Description	Provides the range of memory address allowed for writing.
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspWriteMemoryRangeHigh	1	[ECUC_Dcm_00791]
DcmDspWriteMemoryRangeLow	1	[ECUC_Dcm_00790]
DcmDspWriteMemoryRangeModeRuleRef	0..1	[ECUC_Dcm_00916]
DcmDspWriteMemoryRangeSecurityLevelRef	0..*	[ECUC_Dcm_00793]
DcmDspWriteMemoryRangeSessionLevelRef	0..*	[ECUC_Dcm_01087]

No Included Containers

]

[ECUC_Dcm_00791] Definition of EcucIntegerParamDef DcmDspWriteMemoryRangeHigh

Parameter Name	DcmDspWriteMemoryRangeHigh		
Parent Container	DcmDspWriteMemoryRangeInfo		
Description	High memory address of a range allowed for writing.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967294		
Default value	—		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_00790] Definition of EcucIntegerParamDef DcmDspWriteMemoryRangeLow

Parameter Name	DcmDspWriteMemoryRangeLow
Parent Container	DcmDspWriteMemoryRangeInfo
Description	Low memory address of a range allowed for writing





Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967294		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00916] Definition of EcucReferenceDef DcmDspWriteMemoryRangeModeRuleRef [

Parameter Name	DcmDspWriteMemoryRangeModeRuleRef		
Parent Container	DcmDspWriteMemoryRangeInfo		
Description	Reference to DcmModeRule Mode rule which controls write access on this memory address. If there is no reference, no check of the mode rule shall be done.		
Multiplicity	0..1		
Type	Reference to DcmModeRule		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00793] Definition of EcucReferenceDef DcmDspWriteMemoryRangeSecurityLevelRef [

Parameter Name	DcmDspWriteMemoryRangeSecurityLevelRef		
Parent Container	DcmDspWriteMemoryRangeInfo		
Description	Link to the Security Access Levels needed for write access on this memory address. If there is no reference, no check of security level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSecurityRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD



△

Value Configuration Class	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
Dependency	Post-build time	–	

└

[ECUC_Dcm_01087] Definition of EcucReferenceDef DcmDspWriteMemoryRangeSessionLevelRef

Parameter Name	DcmDspWriteMemoryRangeSessionLevelRef		
Parent Container	DcmDspWriteMemoryRangeInfo		
Description	Link to the session level needed for access to this memory address range. If there is no reference, no check of session level shall be done.		
Multiplicity	0..*		
Type	Reference to DcmDspSessionRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

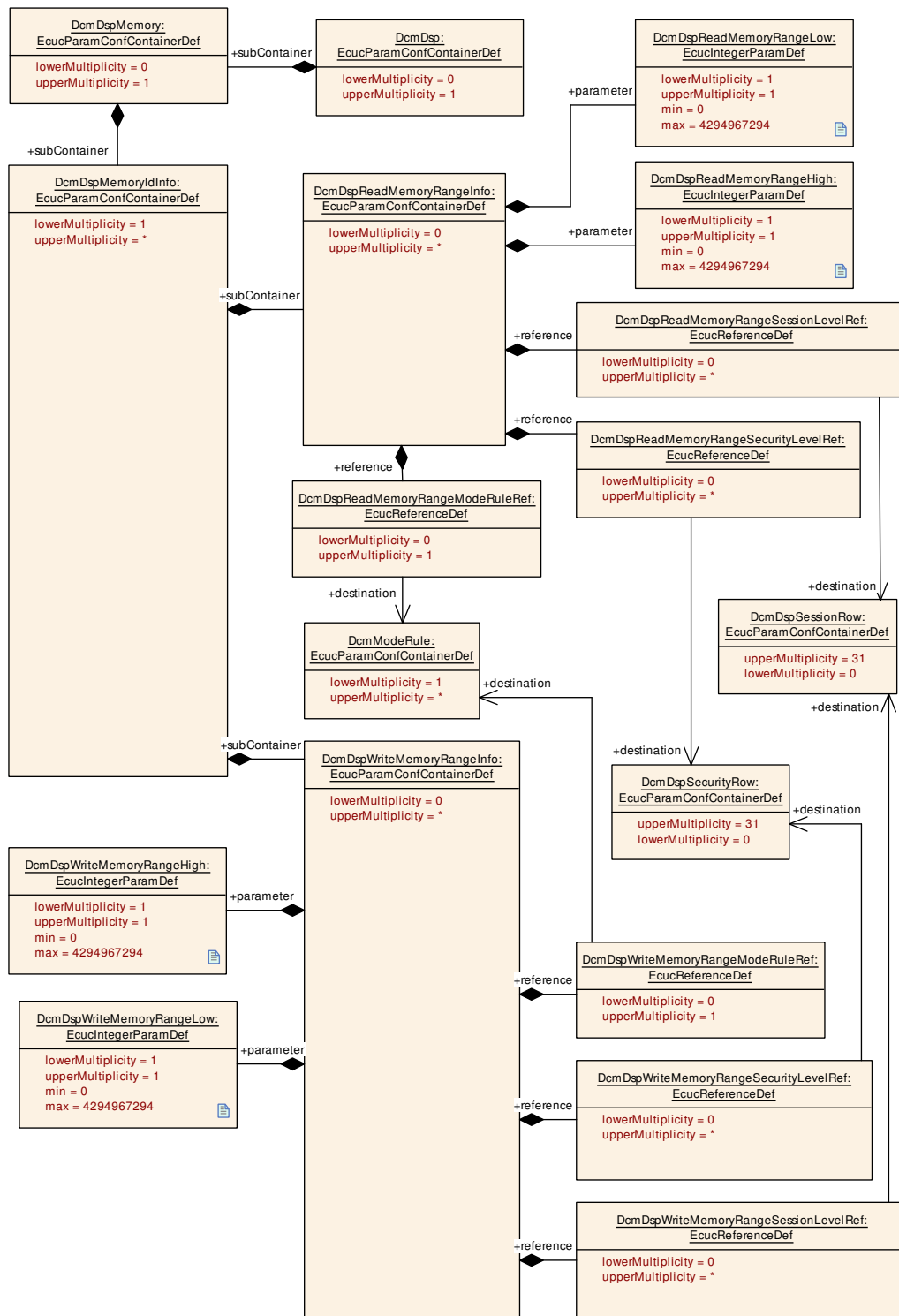


Figure 10.37: DcmDspMemoryReadWrite configuration overview

10.2.5.15 PIDs

10.2.5.15.1 DcmDspPid

[ECUC_Dcm_00626] Definition of EcucParamConfContainerDef DcmDspPid [

Container Name	DcmDspPid
Parent Container	DcmDsp
Description	This container defines the availability of a PID to the DCM.
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspPidIdentifier	1	[ECUC_Dcm_00627]
DcmDspPidService	1	[ECUC_Dcm_00893]
DcmDspPidSize	1	[ECUC_Dcm_00870]
DcmDspPidUsed	1	[ECUC_Dcm_00806]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspPidData	1..*	This container defines the parameter for a Signal in the PID.
DcmDspPidSupportInfo	0..*	This container defines the support information (typically byte A) to declare the usability of the data bytes within the so-called packeted PIDs (e.g. PID\$68).

]

[ECUC_Dcm_00627] Definition of EcucIntegerParamDef DcmDspPidIdentifier [

Parameter Name	DcmDspPidIdentifier		
Parent Container	DcmDspPid		
Description	1 byte Identifier of the PID Within each DcmConfigSet all DcmDspPidIdentifier values shall be unique.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00893] Definition of EcucEnumerationParamDef DcmDspPidService

Parameter Name	DcmDspPidService		
Parent Container	DcmDspPid		
Description	Indicates if a PID is used with service \$01 and/or \$02		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DCM_SERVICE_01	A PID is used with service \$01 only.	
	DCM_SERVICE_01_02	A PID is used with service \$01 and \$02. Allowed with a PID configuration containing data elements on byte basis.	
	DCM_SERVICE_02	A PID is used with service \$02 only. Allowed with a PID configuration containing data elements on byte basis.	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

[ECUC_Dcm_00870] Definition of EcucIntegerParamDef DcmDspPidSize

Parameter Name	DcmDspPidSize		
Parent Container	DcmDspPid		
Description	Length of a PID in byte(s).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	—		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

[ECUC_Dcm_00806] Definition of EcucBooleanParamDef DcmDspPidUsed

Parameter Name	DcmDspPidUsed		
Parent Container	DcmDspPid		
Description	Allow to activate or deactivate the usage of a PID, for multi purpose ECUs true = PID is available false = PID is not available		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		





Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

└

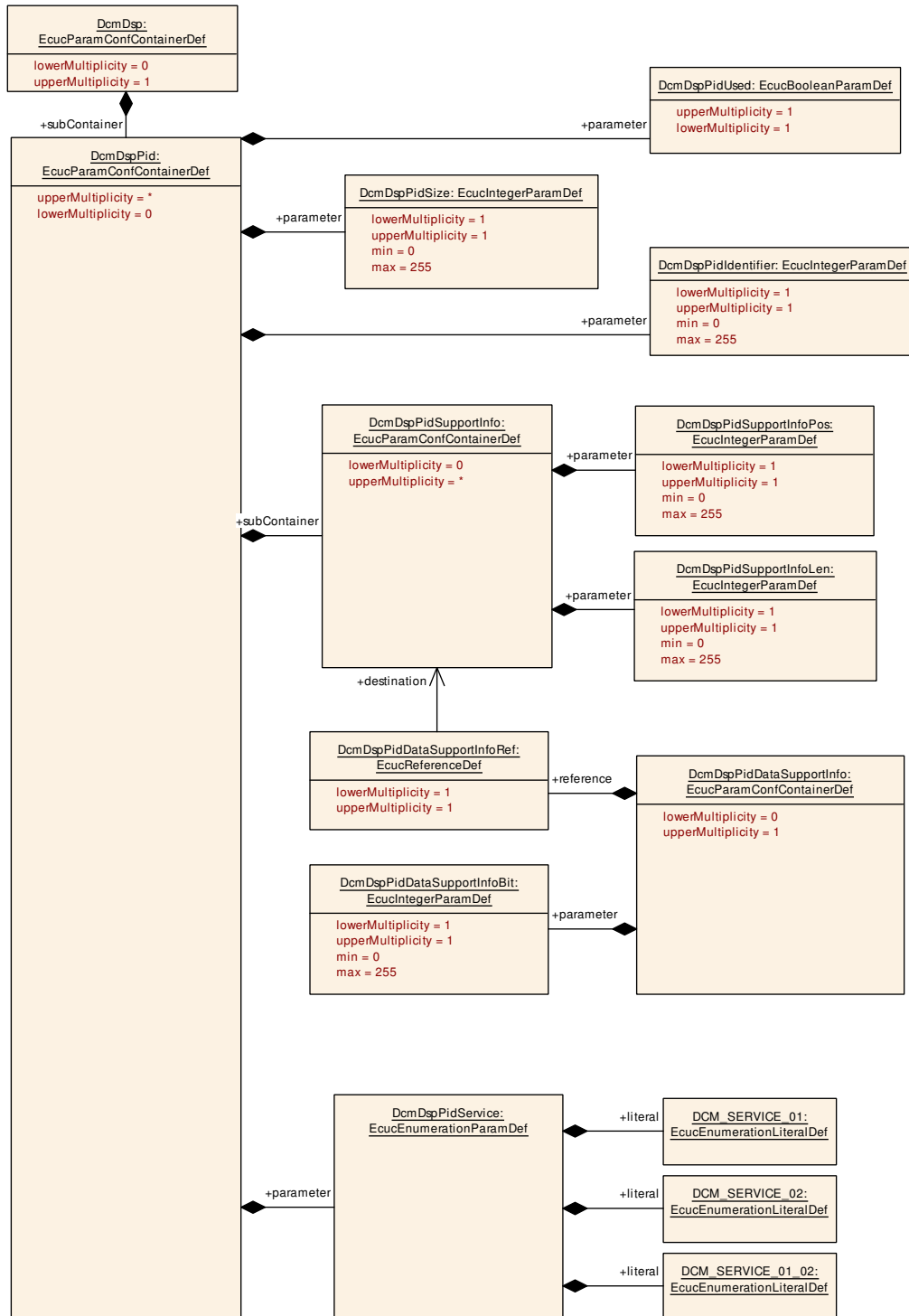


Figure 10.38: DcmDspPid1 configuration overview

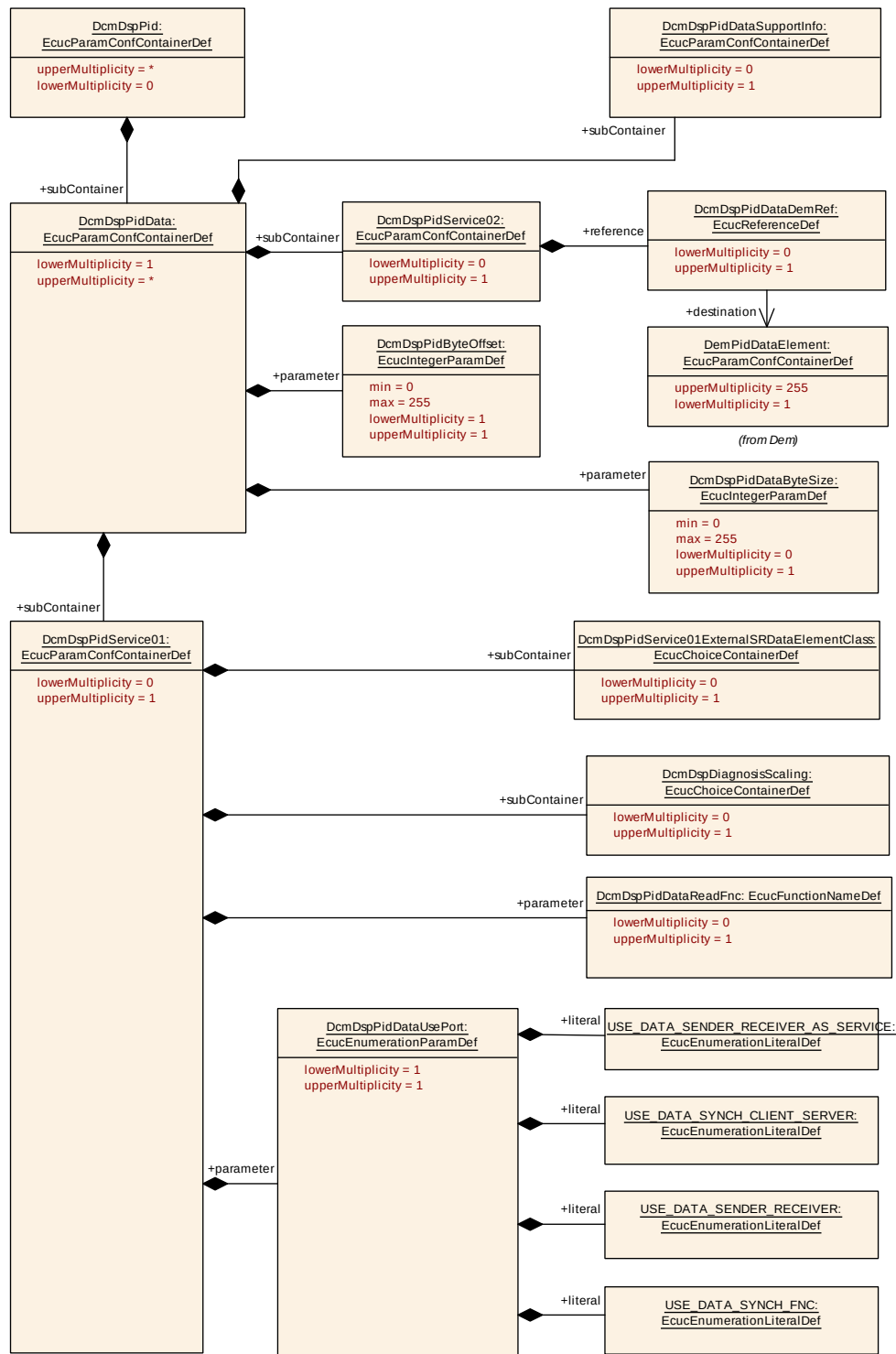


Figure 10.39: DcmDspPid2 configuration overview

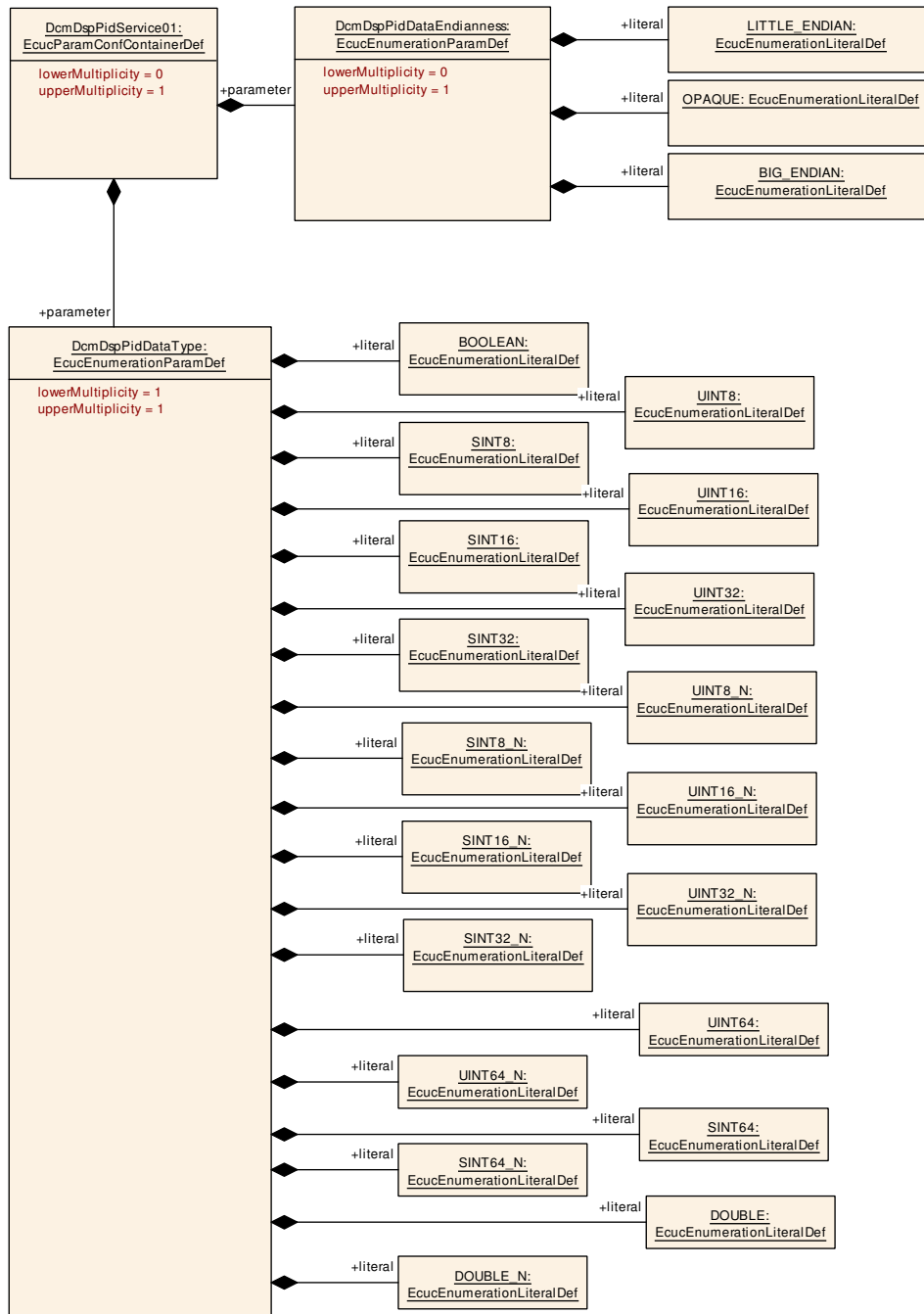


Figure 10.40: DcmDspPid3 configuration overview

10.2.5.15.2 DcmDspPidSupportInfo

[ECUC_Dcm_00871] Definition of EcucParamConfContainerDef DcmDspPidSupportInfo

Container Name	DcmDspPidSupportInfo
Parent Container	DcmDspPid
Description	This container defines the support information (typically byte A) to declare the usability of the data bytes within the so-called packeted PIDs (e.g. PID\$68).
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspPidSupportInfoLen	1	[ECUC_Dcm_00873]
DcmDspPidSupportInfoPos	1	[ECUC_Dcm_00872]

No Included Containers

]

[ECUC_Dcm_00873] Definition of EcucIntegerParamDef DcmDspPidSupportInfo Len

Parameter Name	DcmDspPidSupportInfoLen		
Parent Container	DcmDspPidSupportInfo		
Description	Length of the support information in bytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00872] Definition of EcucIntegerParamDef DcmDspPidSupportInfo Pos

Parameter Name	DcmDspPidSupportInfoPos		
Parent Container	DcmDspPidSupportInfo		
Description	Position of the support information in bytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME





	Post-build time	–	
Dependency			

]

10.2.5.15.3 DcmDspPidData

[ECUC_Dcm_00865] Definition of EcucParamConfContainerDef DcmDspPidData

[

Container Name	DcmDspPidData
Parent Container	DcmDspPid
Description	This container defines the parameter for a Signal in the PID.
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspPidByteOffset	1	[ECUC_Dcm_01107]
DcmDspPidDataByteSize	0..1	[ECUC_Dcm_01108]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspPidDataSupportInfo	0..1	This container defines the supported information.
DcmDspPidService01	0..1	Contains specific configuration parameter of PID for service \$01. This container exists only if DcmDspPidService is set to DCM_SERVICE_01 or DCM_SERVICE_01_02.
DcmDspPidService02	0..1	Contains specific configuration parameter of PID for service \$02. This container exists only if DcmDspPidService is set to DCM_SERVICE_02 or DCM_SERVICE_01_02.

]

[ECUC_Dcm_01107] Definition of EcucIntegerParamDef DcmDspPidByteOffset

[

Parameter Name	DcmDspPidByteOffset		
Parent Container	DcmDspPidData		
Description	This is the position in bytes of the PID structure and will not start at position 0 in case a support information is available (for packeted PIDs).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME





	Post-build time	–	
Dependency			

[ECUC_Dcm_01108] Definition of EcucIntegerParamDef DcmDspPidDataByteSize

Parameter Name	DcmDspPidDataByteSize		
Parent Container	DcmDspPidData		
Description	Defines the array length in bytes or the maximum array length for variable datalengths.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

10.2.5.15.4 DcmDspPidService01

[ECUC_Dcm_00894] Definition of EcucParamConfContainerDef DcmDspPidService01

Container Name	DcmDspPidService01
Parent Container	DcmDspPidData
Description	Contains specific configuration parameter of PID for service \$01. This container exists only if DcmDspPidService is set to DCM_SERVICE_01 or DCM_SERVICE_01_02.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspPidDataEndianness	0..1	[ECUC_Dcm_01012]
DcmDspPidDataReadFnc	0..1	[ECUC_Dcm_00629]
DcmDspPidDataType	1	[ECUC_Dcm_01018]
DcmDspPidDataUsePort	1	[ECUC_Dcm_00720]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspDiagnosisScaling	0..1	This container contains the configuration (parameters) of an alternative Diagnosis Representation. Out if this the scaling between Diagnosis and ECU internal representation and vice versa can be calculated.
DcmDspPidService01ExternalSRDataElementClass	0..1	This container defines the source of data in a provided port which shall be read respectively the target of data in a required port which shall be written. This container shall contain either one DcmSubElementInDataElementInstance OR DcmDataElementInstance OR DcmSubElementInImplDataElementInstance reference.

]

[ECUC_Dcm_01012] Definition of EcucEnumerationParamDef DcmDspPidDataEndianness

Parameter Name	DcmDspPidDataEndianness		
Parent Container	DcmDspPidService01		
Description	Defines the endianness of the data belonging to a PID in a diagnostic response message. If no DcmDspPidDataEndianness is defined the value of DcmDspDataDefaultEndianness is applicable.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BIG_ENDIAN	Most significant byte shall be stored at the lowest address.	
	LITTLE_ENDIAN	Most significant byte shall be stored at the highest address	
	OPAQUE	Opaque data endianness	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00629] Definition of EcucFunctionNameDef DcmDspPidDataReadFnc

Parameter Name	DcmDspPidDataReadFnc		
Parent Container	DcmDspPidService01		
Description	Function name for reading PID data value. This is only relevant if DcmDspPidDataUsePort==USE_DATA_SYNCH_FNC. This parameter is related to the interface Xxx_ReadData.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		





Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01018] Definition of EcucEnumerationParamDef DcmDspPidData Type [

Parameter Name	DcmDspPidDataType		
Parent Container	DcmDspPidService01		
Description	Provide the implementation data type of data belonging to a PID.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	BOOLEAN	Type of the data is boolean.	
	DOUBLE	Type of the data is double (64 bit float)	
	DOUBLE_N	Type of the data is double (64 bit float) array	
	SINT16	Type of the data is sint16.	
	SINT16_N	Type of the data is sint16 array.	
	SINT32	Type of the data is sint32.	
	SINT32_N	Type of the data is sint32 array.	
	SINT64	Type of the data is sint64	
	SINT64_N	Type of the data is sint64 array	
	SINT8	Type of the data is sint8.	
	SINT8_N	Type of the data is sint8 array.	
	UINT16	Type of the data is uint16.	
	UINT16_N	Type of the data is uint16 array.	
	UINT32	Type of the data is uint32.	
	UINT32_N	Type of the data is uint32 array.	
	UINT64	Type of the data is uint64.	
	UINT64_N	Type of the data is uint64 array	
	UINT8	Type of the data is uint8.	
	UINT8_N	Type of the data is uint8 array.	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME





	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

┌

[ECUC_Dcm_00720] Definition of EcucEnumerationParamDef DcmDspPidData UsePort

Parameter Name	DcmDspPidDataUsePort		
Parent Container	DcmDspPidService01		
Description	If this parameter is set to USE_DATA_SYNCH_FNC, the Dcm will use the function defined in DcmDspPidDataReadFnc to get the PID data value. If this parameter is set to USE_DATA_SYNCH_CLIENT_SERVER, the Dcm will have an R-Port requiring the interface DataServices_{Data}. If this parameter is set to USE_DATA_SENDER_RECEIVER or USE_DATA_SENDER_RECEIVER_AS_SERVICE, the DCM will have an R-Port requiring a SenderReceiver Interface. The R-Port is named DataServices_{Data} where {Data} is the name of the container DcmDspPidData.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	USE_DATA_SENDER_RECEIVER	–	
	USE_DATA_SENDER_RECEIVER_AS_SERVICE	–	
	USE_DATA_SYNCH_CLIENT_SERVER	–	
	USE_DATA_SYNCH_FNC	–	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

10.2.5.15.5 DcmDspPidService02

[ECUC_Dcm_00895] Definition of EcucParamConfContainerDef DcmDspPidService02

Container Name	DcmDspPidService02
Parent Container	DcmDspPidData
Description	Contains specific configuration parameter of PID for service \$02. This container exists only if DcmDspPidService is set to DCM_SERVICE_02 or DCM_SERVICE_01_02.





Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspPidDataDemRef	0..1	[ECUC_Dcm_00887]

No Included Containers

└

[[ECUC_Dcm_00887](#)] Definition of EcucReferenceDef DcmDspPidDataDemRef [

Parameter Name	DcmDspPidDataDemRef		
Parent Container	DcmDspPidService02		
Description	Reference to DemPidDataElement in DEM configuration. Allows to link the DCM PID and DEM PID configuration for Mode \$02.		
Multiplicity	0..1		
Type	Reference to DemPidDataElement		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

10.2.5.15.6 DcmDspPidDataSupportInfo

[[ECUC_Dcm_00874](#)] Definition of EcucParamConfContainerDef DcmDspPidDataSupportInfo [

Container Name	DcmDspPidDataSupportInfo
Parent Container	DcmDspPidData
Description	This container defines the supported information.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspPidDataSupportInfoBit	1	[ECUC_Dcm_00876]
DcmDspPidDataSupportInfoRef	1	[ECUC_Dcm_00875]

No Included Containers

]

[ECUC_Dcm_00876] Definition of EcucIntegerParamDef DcmDspPidDataSupportInfoBit

Parameter Name	DcmDspPidDataSupportInfoBit		
Parent Container	DcmDspPidDataSupportInfo		
Description	Referenced Bit of the SupportInfo		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00875] Definition of EcucReferenceDef DcmDspPidDataSupportInfoRef

Parameter Name	DcmDspPidDataSupportInfoRef		
Parent Container	DcmDspPidDataSupportInfo		
Description	Reference to DcmDspPidSupportInfo		
Multiplicity	1		
Type	Reference to DcmDspPidSupportInfo		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

10.2.5.16 DcmDspRequestControl

[ECUC_Dcm_00637] Definition of EcucParamConfContainerDef DcmDspRequestControl

Container Name	DcmDspRequestControl
Parent Container	DcmDsp
Description	This container contains the configuration (parameters) of the "Request control of on-board system, test or component" service (Service \$08). The DCM will request the control using an R-Port requiring a PortInterface RequestControlServices_{Tid}. The R-Port is named RequestControlServices_{Tid} where {Tid} is the name of the container DcmDspRequestControl.
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspRequestControlInBufferSize	1	[ECUC_Dcm_00722]
DcmDspRequestControlInfoByte	0..1	[ECUC_Dcm_01078]
DcmDspRequestControlOutBufferSize	1	[ECUC_Dcm_00723]
DcmDspRequestControlTestId	1	[ECUC_Dcm_00656]

No Included Containers

└

[ECUC_Dcm_00722] Definition of EcucIntegerParamDef DcmDspRequestControlInBufferSize

Parameter Name	DcmDspRequestControlInBufferSize		
Parent Container	DcmDspRequestControl		
Description	Number of bytes to be provided in the input buffer of the interface RequestControlServices_{Tid} for OBD Service \$08		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01078] Definition of EcucIntegerParamDef DcmDspRequestControlInfoByte

Parameter Name	DcmDspRequestControlInfoByte
Parent Container	DcmDspRequestControl
Description	Manufacturer specific value reported to the tester for the record identifiers 0xE000 to 0xE1FF. (WWH-OBd use cases)
Multiplicity	0..1
Type	EcucIntegerParamDef





Range	0 .. 255	
Default value	–	
Post-Build Variant Value	false	
Value Configuration Class	Pre-compile time	X VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X VARIANT-LINK-TIME
	Post-build time	–
Dependency		

└

[ECUC_Dcm_00723] Definition of EcucIntegerParamDef DcmDspRequestControl OutBufferSize

Parameter Name	DcmDspRequestControlOutBufferSize		
Parent Container	DcmDspRequestControl		
Description	Number of bytes to be provided in the output buffer of the interface RequestControl Services_{Tid} for OBD Service \$08		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00656] Definition of EcucIntegerParamDef DcmDspRequestControl TestId

Parameter Name	DcmDspRequestControlTestId		
Parent Container	DcmDspRequestControl		
Description	Test Id for Service \$08		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

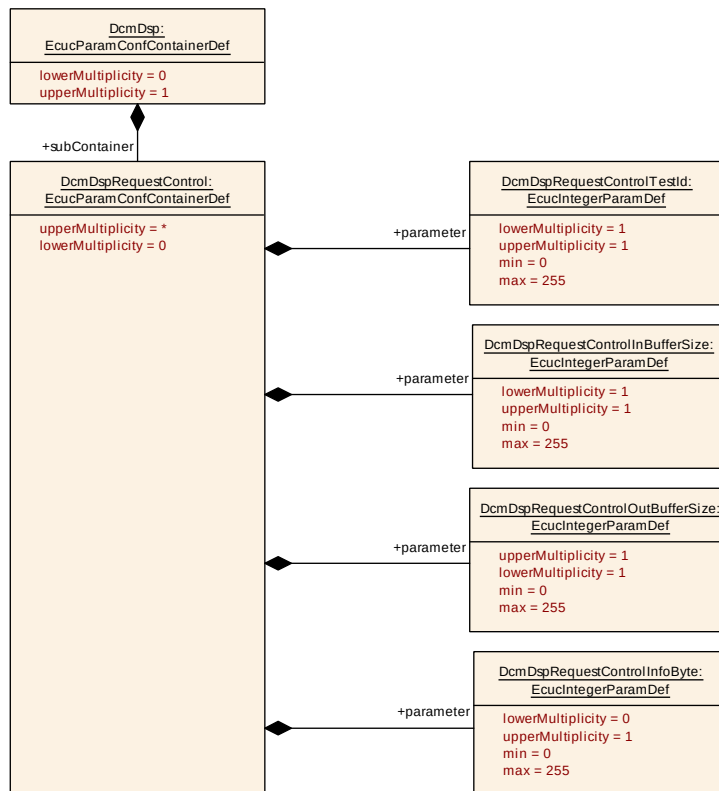


Figure 10.41: DcmDspRequestControl configuration overview

10.2.5.17 DcmDspRequestFileTransfer

[ECUC_Dcm_01034] Definition of EcucParamConfContainerDef DcmDspRequestFileTransfer

Container Name	DcmDspRequestFileTransfer
Parent Container	DcmDsp
Description	This container contains the configuration for RequestFileTransfer. This container only exists if RequestFileTransfer is configured.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmRequestFileTransferFileSizeOrDirInfoParameterLength	1	[ECUC_Dcm_01035]
DcmRequestFileTransferLengthFormatIdentifier	1	[ECUC_Dcm_01036]
DcmRequestFileTransferMaxFileAndDirName	0..1	[ECUC_Dcm_01130]
DcmRequestFileTransferUsePort	1	[ECUC_Dcm_01131]

No Included Containers

]

[ECUC_Dcm_01035] Definition of EcucIntegerParamDef DcmRequestFileTransferFileSizeOrDirInfoParameterLength [

Parameter Name	DcmRequestFileTransferFileSizeOrDirInfoParameterLength		
Parent Container	DcmDspRequestFileTransfer		
Description	Defines the length (number of bytes, i.e. the value of fileSizeOrDirInfoParameterLength) of the fileSizeUncompressedOrDirInfoLength and fileSizeCompressed in the response of RequestFileTransfer.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4		
Default value	4		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01036] Definition of EcucIntegerParamDef DcmRequestFileTransferLengthFormatIdentifier [

Parameter Name	DcmRequestFileTransferLengthFormatIdentifier		
Parent Container	DcmDspRequestFileTransfer		
Description	Defines the length (number of bytes) of the maxNumberOfBlockLength parameter in the response of RequestFileTransfer.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4		
Default value	4		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	—	
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_01130] Definition of EcucIntegerParamDef DcmRequestFileTransferMaxFileAndDirName [

Parameter Name	DcmRequestFileTransferMaxFileAndDirName	
Parent Container	DcmDspRequestFileTransfer	
Description	Defines the maximum size allowed for the FileAndDirName parameter with RTE interfaces used for RequestFileTransfer.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	1 .. 65535	
Default value	1	





Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01131] Definition of EcucBooleanParamDef DcmRequestFileTransferUsePort

Parameter Name	DcmRequestFileTransferUsePort		
Parent Container	DcmDspRequestFileTransfer		
Description	Defines if a C/S or C function call shall be used for RequestFileTransfer processing.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

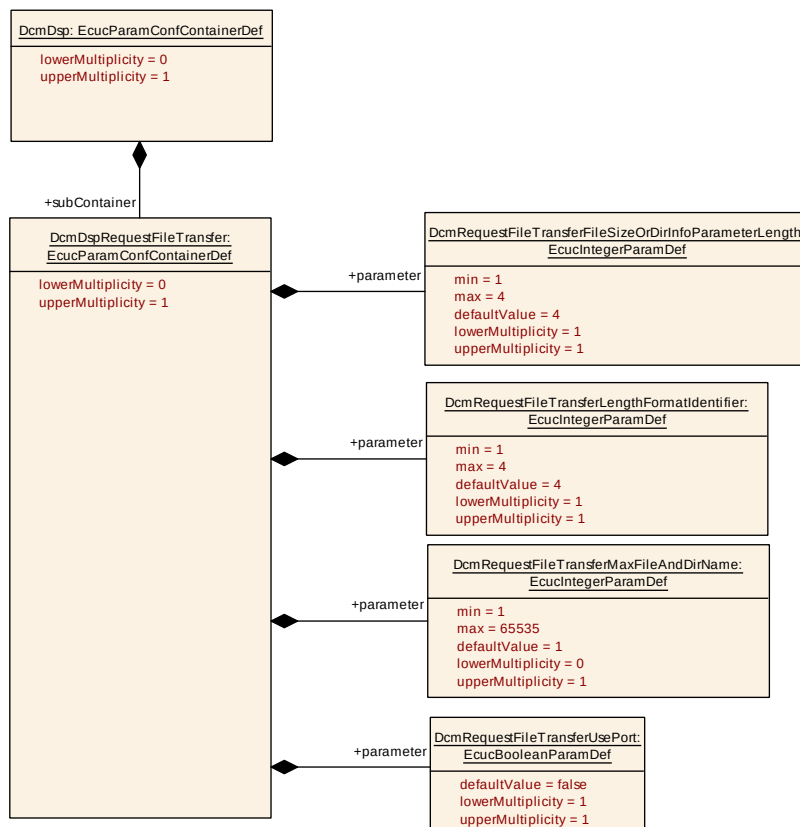


Figure 10.42: DcmDspRequestFileTransfer configuration overview

10.2.5.18 Response on Event

10.2.5.18.1 DcmDspRoe

[ECUC_Dcm_00858] Definition of EcucParamConfContainerDef DcmDspRoe [

Container Name	DcmDspRoe
Parent Container	DcmDsp
Description	Provide the configuration of the ResponseOnEvent mechanism.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspRoeMaxNumChangeOfDataIdentifierEvents	0..1	[ECUC_Dcm_01236]
DcmDspRoeMaxNumComparisonOfValueEvents	0..1	[ECUC_Dcm_01237]
DcmDspRoeMaxNumOfStoredDTCStatusChangedEvents	0..1	[ECUC_Dcm_01239]
DcmDspRoeMaxSupportedDIDLength	0..1	[ECUC_Dcm_01238]
DcmDspRoeSchedulerRate	1	[ECUC_Dcm_01235]
DcmDspRoeStoreEventSupport	1	[ECUC_Dcm_01234]

No Included Containers

]

[ECUC_Dcm_01236] Definition of EcucIntegerParamDef DcmDspRoeMaxNumChangeOfDataIdentifierEvents [

Parameter Name	DcmDspRoeMaxNumChangeOfDataIdentifierEvents		
Parent Container	DcmDspRoe		
Description	Maximum number of events that can be simultaneously configured with SubFunction onChangeOfDataIdentifier.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01237] Definition of EcucIntegerParamDef DcmDspRoeMaxNumComparisonOfValueEvents [

Parameter Name	DcmDspRoeMaxNumComparisonOfValueEvents		
Parent Container	DcmDspRoe		
Description	Maximum number of events that can be simultaneously configured with SubFunction onComparisonOfValues.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01239] Definition of EcucIntegerParamDef DcmDspRoeMaxNumOfStoredDTCStatusChangedEvents [

Parameter Name	DcmDspRoeMaxNumOfStoredDTCStatusChangedEvents		
Parent Container	DcmDspRoe		
Description	Maximum number of changed DTC events that can be stored within one DcmDspRoe SchedulerRate interval.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01238] Definition of EcucIntegerParamDef DcmDspRoeMaxSupportedDIDLength [

Parameter Name	DcmDspRoeMaxSupportedDIDLength		
Parent Container	DcmDspRoe		
Description	Maximum DID response size in bytes allowed for a DID that is used for onComparisonOfValues or onChangeOfDataIdentifier.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01235] Definition of EcucFloatParamDef DcmDspRoeSchedulerRate [

Parameter Name	DcmDspRoeSchedulerRate		
Parent Container	DcmDspRoe		
Description	Calling rate of the RoE scheduler. The parameter value is defined in seconds and shall be a multiple of DcmTaskTime.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[-INF .. INF]		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01234] Definition of EcucBooleanParamDef DcmDspRoeStoreEventSupport [

Parameter Name	DcmDspRoeStoreEventSupport		
Parent Container	DcmDspRoe		
Description	Configures if ResponseOnEvent storageState is supported. If set to true, the use of the storageState bit is supported and if set to false the use of the storageState bit is not supported.		





Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

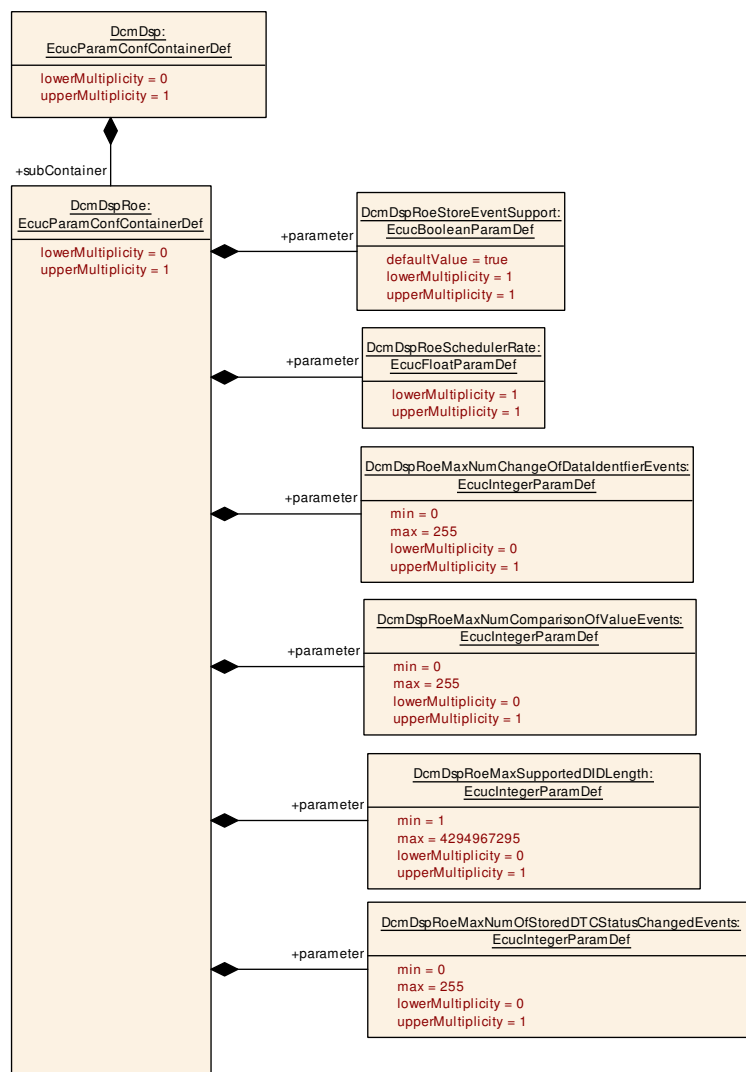


Figure 10.43: DcmDspRoe configuration overview

10.2.5.19 Routines

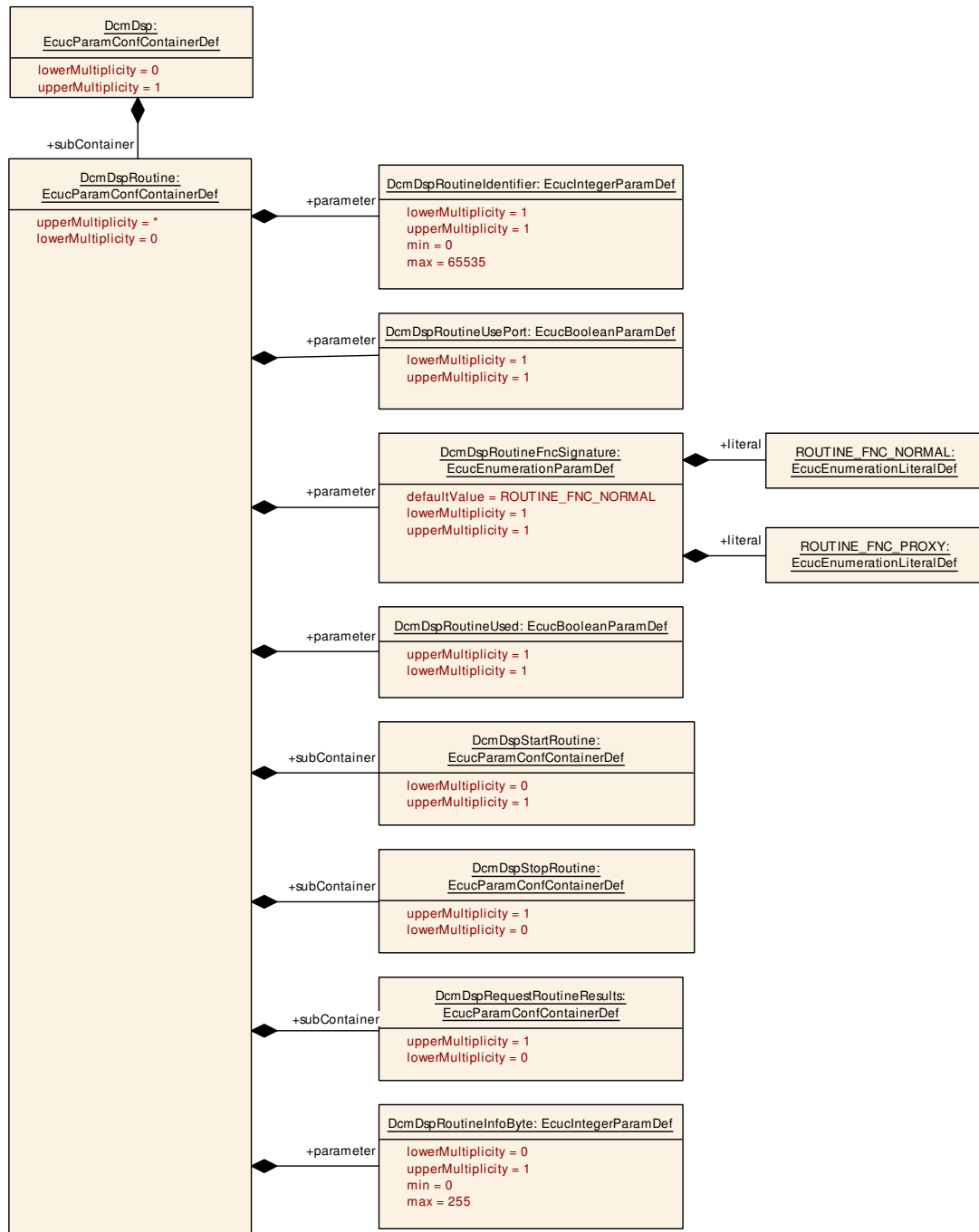


Figure 10.44: DcmDspRoutine configuration overview

10.2.5.19.1 DcmDspRoutine

[ECUC_Dcm_00640] Definition of **EcucParamConfContainerDef DcmDspRoutine**

[

Container Name	DcmDspRoutine
Parent Container	DcmDsp
Description	This container contains the configuration (parameters) for Routines
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspRoutineFncSignature	1	[ECUC_Dcm_01215]
DcmDspRoutineIdentifier	1	[ECUC_Dcm_00641]
DcmDspRoutineInfoByte	0..1	[ECUC_Dcm_01063]
DcmDspRoutineUsed	1	[ECUC_Dcm_00807]
DcmDspRoutineUsePort	1	[ECUC_Dcm_00724]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspRequestRoutineResults	0..1	Provides the configuration of RequestResult subservice for RoutineControl service. Existence indicates that the Request RoutineResults in the RoutineControl is supported.
DcmDspStartRoutine	0..1	Provides the configuration of Start subservice for RoutineControl service.
DcmDspStopRoutine	0..1	Provides the configuration of Stop subservice for RoutineControl service. Existence indicates that the StopRoutine in the Routine Control is supported.

]

[ECUC_Dcm_01215] Definition of EcucEnumerationParamDef DcmDspRoutineFncSignature

Status: DRAFT

[

Parameter Name	DcmDspRoutineFncSignature	
Parent Container	DcmDspRoutine	
Description	If the parameter is set to ROUTINE_FNC_NORMAL the routine control functions DcmDspStartRoutineFnc, DcmDspStopRoutineFnc or DcmDspRequestResultsRoutineFnc are defined with the signature for a cluster local user with separate arguments per signal. If the parameter is set to ROUTINE_FNC_PROXY the routine control functions DcmDspStartRoutineFnc, DcmDspStopRoutineFnc or DcmDspRequestResultsRoutineFnc with a fixed number of arguments passing a common buffer for all signals and a DataLength. Tags: atp.Status=draft	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	ROUTINE_FNC_NORMAL	Function calls with normal set of arguments used for Sw Cluster internal interfaces. Tags: atp.Status=draft
	ROUTINE_FNC_PROXY	Function call used with generic set of arguments used for Dcm low proxy. Tags: atp.Status=draft
Default value	ROUTINE_FNC_NORMAL	





Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00641] Definition of EcucIntegerParamDef DcmDspRoutineIdentifier

Parameter Name	DcmDspRoutineIdentifier		
Parent Container	DcmDspRoutine		
Description	2 bytes Identifier of the RID Within each DcmConfigSet all DcmDspRoutineIdentifier values shall be unique.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_01063] Definition of EcucIntegerParamDef DcmDspRoutineInfo Byte

Parameter Name	DcmDspRoutineInfoByte		
Parent Container	DcmDspRoutine		
Description	If configured it defines the static value used by the Dcm to report as routine info byte. An example of a possible usage are routines within the OBD range of RID 0xE000-0xE1FF. If the parameter is not configured, no Dcm created routine info byte is provided within the positive response of a RoutineControl Service.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_00807] Definition of EcucBooleanParamDef DcmDspRoutineUsed

Parameter Name	DcmDspRoutineUsed		
Parent Container	DcmDspRoutine		
Description	Allow to activate or deactivate the usage of a Routine, for multi purpose ECUs True = Routine is available False = Routine is not available		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Dependency			

[ECUC_Dcm_00724] Definition of EcucBooleanParamDef DcmDspRoutineUse Port

Parameter Name	DcmDspRoutineUsePort		
Parent Container	DcmDspRoutine		
Description	If this parameter is set to true, the DCM uses a port requiring a PortInterface Routine Services_{RoutineName}. The R-Port is named RoutineServices_{RoutineName} where {RoutineName} is the name of the container DcmDspRoutine. In that case, the configuration must not provide function names in DcmDspStartRoutineFnc, DcmDspStopRoutineFnc or DcmDsp RequestResultsRoutineFnc. If this is false, the DCM expects to find the names of the functions to be used in DcmDspStartRoutineFnc, DcmDspStopRoutineFnc or DcmDsp RequestResultsRoutineFnc.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

10.2.5.19.2 DcmDspRequestRoutineResults

[ECUC_Dcm_01023] Definition of EcucParamConfContainerDef DcmDspRequest RoutineResults

Container Name	DcmDspRequestRoutineResults
Parent Container	DcmDspRoutine
Description	Provides the configuration of RequestResult subservice for RoutineControl service. Existence indicates that the RequestRoutineResults in the RoutineControl is supported.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspRequestRoutineResultsConfirmationEnabled	0..1	[ECUC_Dcm_01091]
DcmDspRequestRoutineResultsConfirmationFnc	0..1	[ECUC_Dcm_01090]
DcmDspRequestRoutineResultsFnc	0..1	[ECUC_Dcm_00753]
DcmDspRoutineInterfaceArgumentIntegrity	0..1	[ECUC_Dcm_01216]
DcmDspRequestRoutineResultsCommonAuthorizationRef	0..1	[ECUC_Dcm_01054]
DcmDspRequestRoutineResultsRoleRef	0..32	[ECUC_Dcm_01146]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspRequestRoutineResultsIn	0..1	Provide description of input parameter of RequestResult subservice for RoutineControl service.
DcmDspRequestRoutineResultsOut	0..1	Provide description of output parameter of RequestResult subservice for RoutineControl service.

]

[ECUC_Dcm_01091] Definition of EcucBooleanParamDef DcmDspRequestRoutineResultsConfirmationEnabled [

Parameter Name	DcmDspRequestRoutineResultsConfirmationEnabled		
Parent Container	DcmDspRequestRoutineResults		
Description	Allows to enable/disable the confirmation function to indicate the transmission of a response to a RequestRoutineResults request		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01090] Definition of EcucFunctionNameDef DcmDspRequestRoutineResultsConfirmationFnc [

Parameter Name	DcmDspRequestRoutineResultsConfirmationFnc		
Parent Container	DcmDspRequestRoutineResults		
Description	C-function to call if a transmission confirmation is needed by the issuer (BSW module)		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00753] Definition of EcucFunctionNameDef DcmDspRequestRoutineResultsFnc [

Parameter Name	DcmDspRequestRoutineResultsFnc		
Parent Container	DcmDspRequestRoutineResults		
Description	Function name for request to application the results of a routine. (Routine_Request Results-function) This parameter is related to the interface Xxx_RequestResults.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01216] Definition of EcucBooleanParamDef DcmDspRoutineInterfaceArgumentIntegrity [

Parameter Name	DcmDspRoutineInterfaceArgumentIntegrity		
Parent Container	DcmDspRequestRoutineResults , DcmDspStartRoutine , DcmDspStopRoutine		
Description	Defines the value of ClientServerOperation.diagArgIntegrity for the created C/S interface of this routine		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01054] Definition of EcucReferenceDef DcmDspRequestRoutineResultsCommonAuthorizationRef [

Parameter Name	DcmDspRequestRoutineResultsCommonAuthorizationRef		
Parent Container	DcmDspRequestRoutineResults		
Description	Reference to DcmDspCommonAuthorization Common authorization configuration taken from the referenced DcmDspRequestRoutineResultsCommonAuthorizationRef. If there is no reference, no check on the commonly defined authorization conditions shall be done to get the routine result.		
Multiplicity	0..1		
Type	Reference to DcmDspCommonAuthorization		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01146] Definition of EcucReferenceDef DcmDspRequestRoutineResultsRoleRef [

Parameter Name	DcmDspRequestRoutineResultsRoleRef		
Parent Container	DcmDspRequestRoutineResults		
Description	Reference to DcmDspAuthenticationRow that defines a role in that this routine results can be read.		
Multiplicity	0..32		
Type	Reference to DcmDspAuthenticationRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

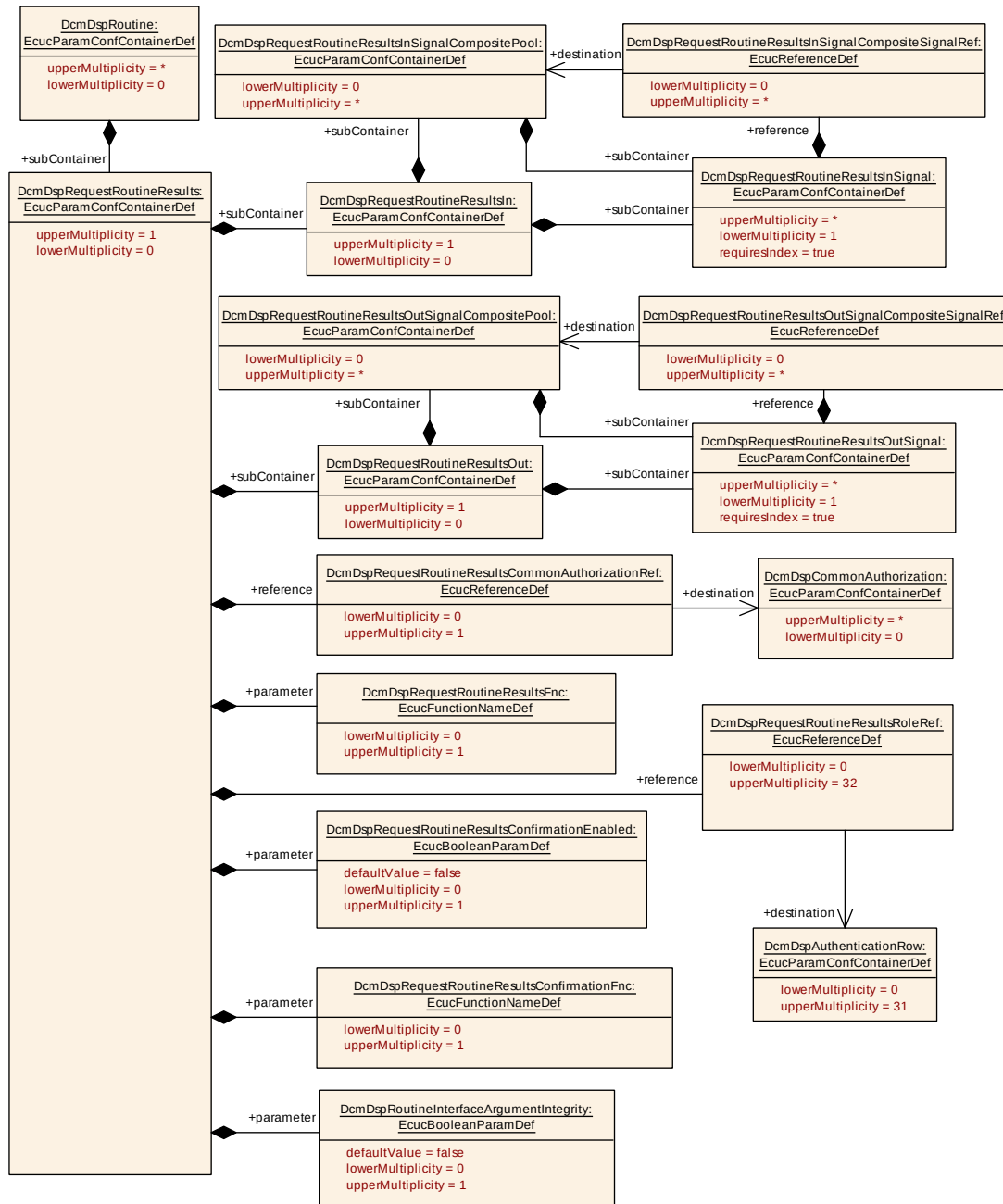


Figure 10.45: DcmDspRequestRoutineResults configuration overview

10.2.5.19.3 DcmDspRequestRoutineResultsIn

[ECUC_Dcm_01116] Definition of EcucParamConfContainerDef DcmDspRequestRoutineResultsIn

Container Name	DcmDspRequestRoutineResultsIn
Parent Container	DcmDspRequestRoutineResults
Description	Provide description of input parameter of RequestResult subservice for RoutineControl service.
Multiplicity	0..1
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspRequestRoutineResultsInSignal	1..*	Provides description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataIn_n elements in the XXX_RequestResult function call.
DcmDspRequestRoutineResultsInSignalCompositePool	0..*	Defines a collection of all contained signals for this root signal.

10.2.5.19.4 DcmDspRequestRoutineResultsInSignal

[ECUC_Dcm_01117] Definition of EcucParamConfContainerDef DcmDspRequestRoutineResultsInSignal

Container Name	DcmDspRequestRoutineResultsInSignal
Parent Container	DcmDspRequestRoutineResultsIn , DcmDspRequestRoutineResultsInSignalCompositePool
Description	Provides description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataIn_n elements in the XXX_RequestResult function call. Attributes: requiresIndex=true
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspRoutineParameterSize	0..1	[ECUC_Dcm_01119]
DcmDspRoutineSignalEndianness	0..1	[ECUC_Dcm_01121]
DcmDspRoutineSignalPos	1	[ECUC_Dcm_01118]
DcmDspRoutineSignalType	0..1	[ECUC_Dcm_01120]
DcmDspRequestRoutineResultsInSignalCompositeSignalRef	0..*	[ECUC_Dcm_01228]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspArgumentScaling	0..1	This container contains the configuration (arguments) of an alternative Diagnosis Representation. Out if this the scaling between Diagnosis and ECU internal representation and vice versa can be calculated.

]

[ECUC_Dcm_01119] Definition of EcucIntegerParamDef DcmDspRoutineParameterSize

Parameter Name	DcmDspRoutineParameterSize		
Parent Container	DcmDspRequestRoutineResultsInSignal		
Description	Provide the size of a RoutineControl parameter in bytes		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01121] Definition of EcucEnumerationParamDef DcmDspRoutineSignalEndianness

Parameter Name	DcmDspRoutineSignalEndianness		
Parent Container	DcmDspRequestRoutineResultsInSignal		
Description	Defines the endianness of the data belonging to a Routine In Signal for RequestResult subfunction.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BIG_ENDIAN		Most significant byte shall be stored at the lowest address.
	LITTLE_ENDIAN		Most significant byte shall be stored at the highest address
	OPAQUE		Opaque data endianness
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01118] Definition of EcucIntegerParamDef DcmDspRoutineSignalPos

Parameter Name	DcmDspRoutineSignalPos		
Parent Container	DcmDspRequestRoutineResultsInSignal		
Description	Provide the position of the signal in the RoutineControl request/response. The position is defined in bits.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_01120] Definition of EcucEnumerationParamDef DcmDspRoutineSignalType

Parameter Name	DcmDspRoutineSignalType	
Parent Container	DcmDspRequestRoutineResultsInSignal	
Description	Provide the type of the signal in the RoutineControl request/response.	
Multiplicity	0..1	
Type	EcucEnumerationParamDef	
Range	BOOLEAN	Type of the signal is boolean.
	DOUBLE	Type of the data is double (64 bit float)
	DOUBLE_N	Type of the data is double (64 bit float) array
	FLOAT	Type of the data is float.
	FLOAT_N	Type of the data is float array.
	SINT16	Type of the signal is sint16.
	SINT16_N	Type of the signal is sint16 array.
	SINT32	Type of the signal is sint32.
	SINT32_N	Type of the signal is sint32 array.
	SINT64	Type of the data is sint64
	SINT64_N	Type of the data is sint64 array
	SINT8	Type of the signal is sint8.
	SINT8_N	Type of the signal is sint8 array.
	UINT16	Type of the signal is uint16.
	UINT16_N	Type of the signal is uint16 array.
	UINT32	Type of the signal is uint32.
	UINT32_N	Type of the signal is uint32 array.
	UINT64	Type of the data is uint64.
	UINT64_N	Type of the data is uint64 array





	UINT8	Type of the signal is uint8.	
	UINT8_N	Type of the signal is uint8 array.	
	VARIABLE_LENGTH	Type of the signal is uint8[DcmDspRoutineParameterSize]. This is only valid for the last signal and when DcmDspRoutineSignalType is set to VARIABLE_LENGTH.	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

└

[ECUC_Dcm_01228] Definition of EcucReferenceDef DcmDspRequestRoutineResultsInSignalCompositeSignalRef

Parameter Name	DcmDspRequestRoutineResultsInSignalCompositeSignalRef		
Parent Container	DcmDspRequestRoutineResultsInSignal		
Description	Reference to the contained signal.		
Multiplicity	0..*		
Type	Reference to DcmDspRequestRoutineResultsInSignalCompositePool		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

10.2.5.19.5 DcmDspRequestRoutineResultsInSignalCompositePool

[ECUC_Dcm_01227] Definition of EcucParamConfContainerDef DcmDspRequestRoutineResultsInSignalCompositePool

Container Name	DcmDspRequestRoutineResultsInSignalCompositePool		
Parent Container	DcmDspRequestRoutineResultsIn		
Description	Defines a collection of all contained signals for this root signal.		
Multiplicity	0..*		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants





	Link time	–	
	Post-build time	–	
Configuration Parameters			

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspRequestRoutineResultsInSignal	1..*	Provides description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataIn_n elements in the XXX_RequestResult function call.

]

10.2.5.19.6 DcmDspRequestRoutineResultsOut

[ECUC_Dcm_00831] Definition of EcucParamConfContainerDef DcmDspRequestRoutineResultsOut [

Container Name	DcmDspRequestRoutineResultsOut
Parent Container	DcmDspRequestRoutineResults
Description	Provide description of output parameter of RequestResult subservice for Routine Control service.
Multiplicity	0..1
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspRequestRoutineResultsOutSignal	1..*	Provides description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataOutN elements in the XXX_RequestResult function call.
DcmDspRequestRoutineResultsOutSignalCompositePool	0..*	Defines a collection of all contained signals for this root signal.

]

10.2.5.19.7 DcmDspRequestRoutineResultsOutSignal

[ECUC_Dcm_00836] Definition of EcucParamConfContainerDef DcmDspRequestRoutineResultsOutSignal [

Container Name	DcmDspRequestRoutineResultsOutSignal
Parent Container	DcmDspRequestRoutineResultsOut , DcmDspRequestRoutineResultsOutSignalCompositePool
Description	Provides description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataOutN elements in the XXX_RequestResult function call. Attributes: requiresIndex=true
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspRoutineParameterSize	0..1	[ECUC_Dcm_00838]
DcmDspRoutineSignalEndianness	0..1	[ECUC_Dcm_01013]
DcmDspRoutineSignalPos	1	[ECUC_Dcm_00837]
DcmDspRoutineSignalType	0..1	[ECUC_Dcm_00881]
DcmDspRequestRoutineResultsOutSignalCompositeSignalRef	0..*	[ECUC_Dcm_01230]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspArgumentScaling	0..1	This container contains the configuration (arguments) of an alternative Diagnosis Representation. Out if this the scaling between Diagnosis and ECU internal representation and vice versa can be calculated.

[ECUC_Dcm_00838] Definition of EcucIntegerParamDef DcmDspRoutineParameterSize

Parameter Name	DcmDspRoutineParameterSize		
Parent Container	DcmDspRequestRoutineResultsOutSignal		
Description	Provide the size of a RoutineControl parameter in bytes		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_01013] Definition of EcucEnumerationParamDef DcmDspRoutineSignalEndianness [

Parameter Name	DcmDspRoutineSignalEndianness		
Parent Container	DcmDspRequestRoutineResultsOutSignal		
Description	Defines the endianness of the data belonging to a Routine Out Signal for Request Result subfunction.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BIG_ENDIAN	Most significant byte shall be stored at the lowest address.	
	LITTLE_ENDIAN	Most significant byte shall be stored at the highest address	
	OPAQUE	Opaque data endianness	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_00837] Definition of EcucIntegerParamDef DcmDspRoutineSignalPos [

Parameter Name	DcmDspRoutineSignalPos		
Parent Container	DcmDspRequestRoutineResultsOutSignal		
Description	Provide the position of the signal in the RoutineControl request/response. The position is defined in bits.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	—		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_00881] Definition of EcucEnumerationParamDef DcmDspRoutineSignalType

Parameter Name	DcmDspRoutineSignalType		
Parent Container	DcmDspRequestRoutineResultsOutSignal		
Description	Provide the type of the signal in the RoutineControl request/response.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BOOLEAN	Type of the signal is boolean.	
	DOUBLE	Type of the data is double (64 bit float)	
	DOUBLE_N	Type of the data is double (64 bit float) array	
	FLOAT	Type of the data is float.	
	FLOAT_N	Type of the data is float array.	
	SINT16	Type of the signal is sint16.	
	SINT16_N	Type of the signal is sint16 array.	
	SINT32	Type of the signal is sint32.	
	SINT32_N	Type of the signal is sint32 array.	
	SINT64	Type of the data is sint64	
	SINT64_N	Type of the data is sint64 array	
	SINT8	Type of the signal is sint8.	
	SINT8_N	Type of the signal is sint8 array.	
	UINT16	Type of the signal is uint16.	
	UINT16_N	Type of the signal is uint16 array.	
	UINT32	Type of the signal is uint32.	
	UINT32_N	Type of the signal is uint32 array.	
	UINT64	Type of the data is uint64.	
	UINT64_N	Type of the data is uint64 array	
	UINT8	Type of the signal is uint8.	
	UINT8_N	Type of the signal is uint8 array.	
	VARIABLE_LENGTH	Type of the signal is uint8[DcmDspRoutineParameterSize]. This is only valid for the last signal and when DcmDspRoutineSignalType is set to VARIABLE_LENGTH.	
	Post-Build Variant Value	false	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_01230] Definition of EcucReferenceDef DcmDspRequestRoutineResultsOutSignalCompositeSignalRef [

Parameter Name	DcmDspRequestRoutineResultsOutSignalCompositeSignalRef		
Parent Container	DcmDspRequestRoutineResultsOutSignal		
Description	Reference to the contained signal.		
Multiplicity	0..*		
Type	Reference to DcmDspRequestRoutineResultsOutSignalCompositePool		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.19.8 DcmDspRequestRoutineResultsOutSignalCompositePool

[ECUC_Dcm_01229] Definition of EcucParamConfContainerDef DcmDspRequestRoutineResultsOutSignalCompositePool [

Container Name	DcmDspRequestRoutineResultsOutSignalCompositePool		
Parent Container	DcmDspRequestRoutineResultsOut		
Description	Defines a collection of all contained signals for this root signal.		
Multiplicity	0..*		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspRequestRoutineResultsOutSignal	1..*	Provides description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataOutN elements in the XXX_RequestResult function call.

]

10.2.5.19.9 DcmDspStartRoutine

[ECUC_Dcm_01021] Definition of EcucParamConfContainerDef DcmDspStartRoutine [

Container Name	DcmDspStartRoutine
Parent Container	DcmDspRoutine
Description	Provides the configuration of Start subservice for RoutineControl service.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspRoutineInterfaceArgumentIntegrity	0..1	[ECUC_Dcm_01216]
DcmDspStartRoutineConfirmationEnabled	0..1	[ECUC_Dcm_01093]
DcmDspStartRoutineConfirmationFnc	0..1	[ECUC_Dcm_01094]
DcmDspStartRoutineFnc	0..1	[ECUC_Dcm_00664]
DcmDspStartRoutineCommonAuthorizationRef	0..1	[ECUC_Dcm_01052]
DcmDspStartRoutineRoleRef	0..32	[ECUC_Dcm_01144]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspStartRoutineIn	0..1	Provide description of input parameter of Start subservice for RoutineControl service
DcmDspStartRoutineOut	0..1	Provide description of output parameter of Start subservice for RoutineControl service.

]

For parameter table [\[ECUC_Dcm_01216\] DcmDspRoutineInterfaceArgumentIntegrity](#), see definition below container [DcmDspRequestRoutineResults](#).

[ECUC_Dcm_01093] Definition of EcucBooleanParamDef DcmDspStartRoutineConfirmationEnabled [

Parameter Name	DcmDspStartRoutineConfirmationEnabled		
Parent Container	DcmDspStartRoutine		
Description	Allows to enable/disable the confirmation function to indicate the transmission of a response to a StartRoutine request		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	





	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01094] Definition of EcucFunctionNameDef DcmDspStartRoutineConfirmationFnc

Parameter Name	DcmDspStartRoutineConfirmationFnc		
Parent Container	DcmDspStartRoutine		
Description	C-function to call if a transmission confirmation is needed by the issuer (BSW module)		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00664] Definition of EcucFunctionNameDef DcmDspStartRoutineFnc

Parameter Name	DcmDspStartRoutineFnc		
Parent Container	DcmDspStartRoutine		
Description	Function name for request to application to start a routine. (Routine_Start-function) This parameter is related to the interface Xxx_Start.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01052] Definition of EcucReferenceDef DcmDspStartRoutineCommonAuthorizationRef [

Parameter Name	DcmDspStartRoutineCommonAuthorizationRef		
Parent Container	DcmDspStartRoutine		
Description	Reference to DcmDspCommonAuthorization Common authorization configuration taken from the referenced DcmDspStartRoutineCommonAuthorizationRef. If there is no reference, no check on the commonly defined authorization conditions shall be done to start the routine.		
Multiplicity	0..1		
Type	Reference to DcmDspCommonAuthorization		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01144] Definition of EcucReferenceDef DcmDspStartRoutineRoleRef [

Parameter Name	DcmDspStartRoutineRoleRef		
Parent Container	DcmDspStartRoutine		
Description	Reference to DcmDspAuthenticationRow that defines a role in that this routine can be started.		
Multiplicity	0..32		
Type	Reference to DcmDspAuthenticationRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

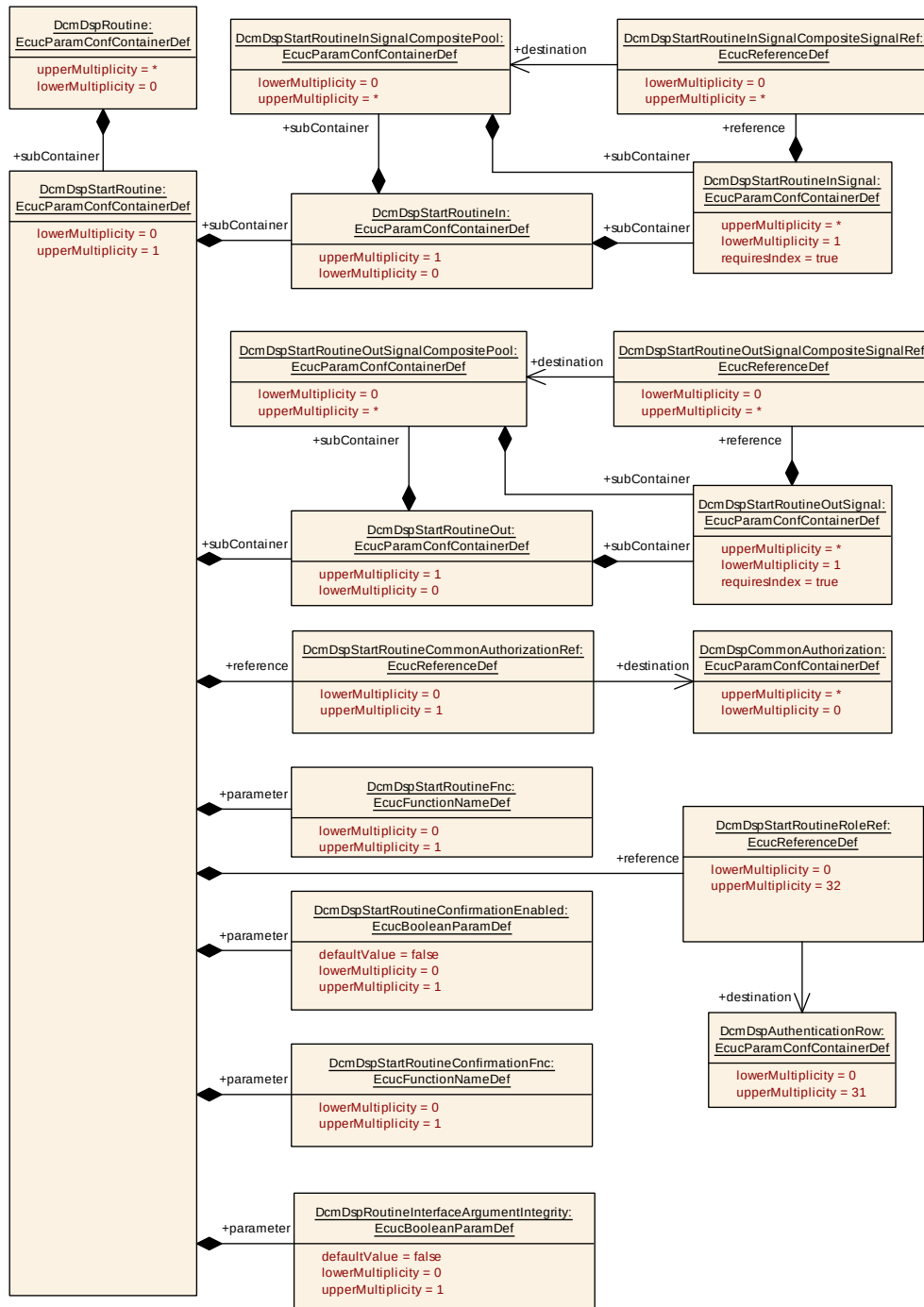


Figure 10.46: DcmDspStartRoutine configuration overview

10.2.5.19.10 DcmDspStartRoutineIn

[ECUC_Dcm_00834] Definition of EcucParamConfContainerDef DcmDspStartRoutineIn

Container Name	DcmDspStartRoutineIn
Parent Container	DcmDspStartRoutine
Description	Provide description of input parameter of Start subservice for RoutineControl service
Multiplicity	0..1
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspStartRoutineInSignal	1..*	Provide description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataInN elements in the XXX_Start function call.
DcmDspStartRoutineInSignal CompositePool	0..*	Defines a collection of all contained signals for this root signal.

]

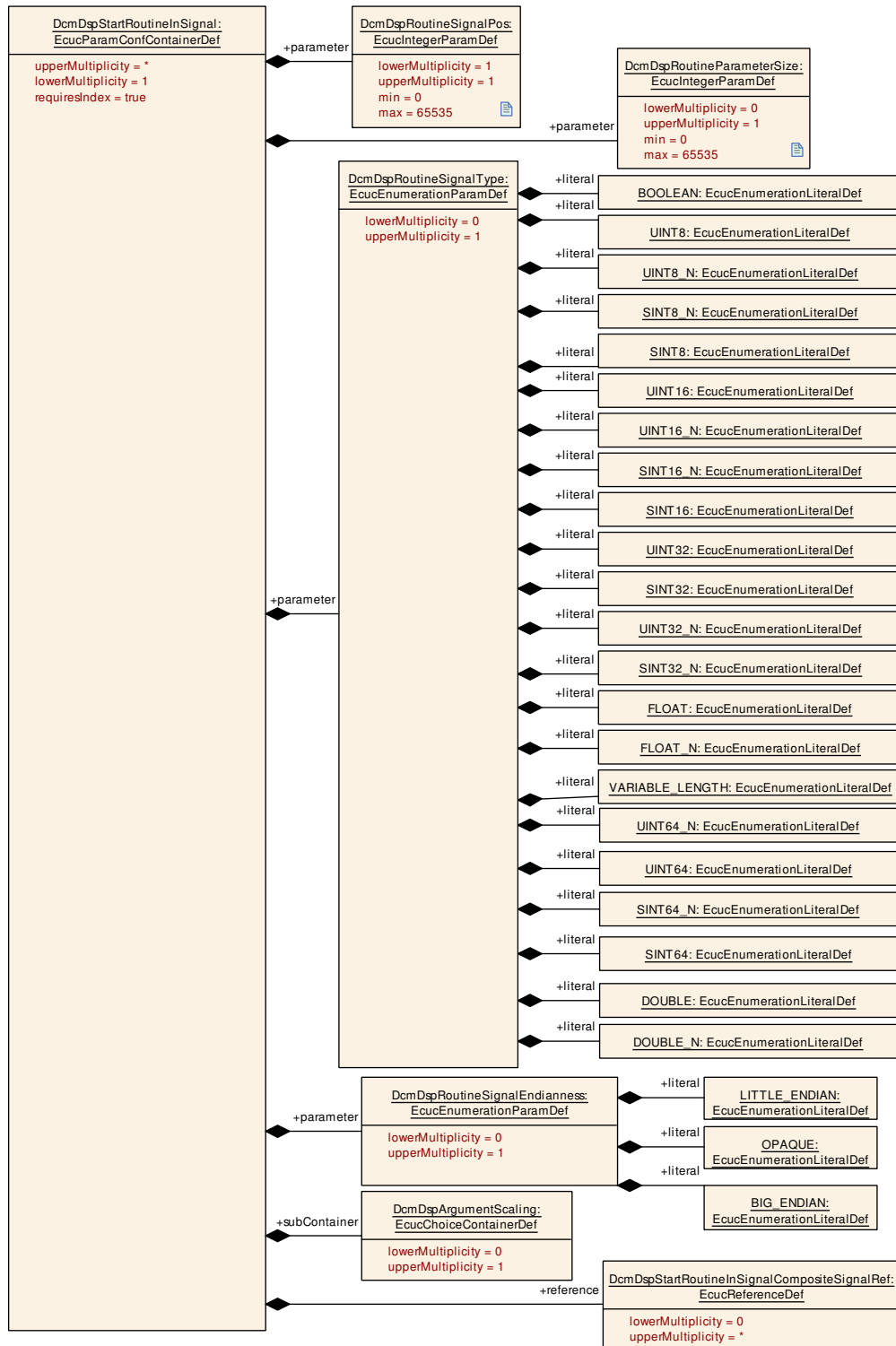


Figure 10.47: DcmDspStartRoutineIn configuration overview

10.2.5.19.11 DcmDspStartRoutineInSignal

[ECUC_Dcm_00845] Definition of EcucParamConfContainerDef DcmDspStartRoutineInSignal [

Container Name	DcmDspStartRoutineInSignal
Parent Container	DcmDspStartRoutineIn , DcmDspStartRoutineInSignalCompositePool
Description	Provide description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataInN elements in the XXX_Start function call. Attributes: requiresIndex=true
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspRoutineParameterSize	0..1	[ECUC_Dcm_00847]
DcmDspRoutineSignalEndianness	0..1	[ECUC_Dcm_01016]
DcmDspRoutineSignalPos	1	[ECUC_Dcm_00846]
DcmDspRoutineSignalType	0..1	[ECUC_Dcm_00884]
DcmDspStartRoutineInSignalCompositeSignalRef	0..*	[ECUC_Dcm_01220]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspArgumentScaling	0..1	This container contains the configuration (arguments) of an alternative Diagnosis Representation. Out if this the scaling between Diagnosis and ECU internal representation and vice versa can be calculated.

]

[ECUC_Dcm_00847] Definition of EcucIntegerParamDef DcmDspRoutineParameterSize [

Parameter Name	DcmDspRoutineParameterSize		
Parent Container	DcmDspStartRoutineInSignal		
Description	Provide the size of a RoutineControl parameter in bytes		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME





	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01016] Definition of EcucEnumerationParamDef DcmDspRoutineSignalEndianness [

Parameter Name	DcmDspRoutineSignalEndianness		
Parent Container	DcmDspStartRoutineInSignal		
Description	Defines the endianness of the data belonging to a Routine In Signal for Start subfunction.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BIG_ENDIAN	Most significant byte shall be stored at the lowest address.	
	LITTLE_ENDIAN	Most significant byte shall be stored at the highest address	
	OPAQUE	Opaque data endianness	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_00846] Definition of EcucIntegerParamDef DcmDspRoutineSignalPos [

Parameter Name	DcmDspRoutineSignalPos		
Parent Container	DcmDspStartRoutineInSignal		
Description	Provide the position of the signal in the RoutineControl request/response. The position is defined in bits.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00884] Definition of EcucEnumerationParamDef DcmDspRoutineSignalType

Parameter Name	DcmDspRoutineSignalType		
Parent Container	DcmDspStartRoutineInSignal		
Description	Provide the type of the signal in the RoutineControl request/response.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BOOLEAN	Type of the signal is boolean.	
	DOUBLE	Type of the data is double (64 bit float)	
	DOUBLE_N	Type of the data is double (64 bit float) array	
	FLOAT	Type of the data is float.	
	FLOAT_N	Type of the data is float array.	
	SINT16	Type of the signal is sint16.	
	SINT16_N	Type of the signal is sint16 array.	
	SINT32	Type of the signal is sint32.	
	SINT32_N	Type of the signal is sint32 array.	
	SINT64	Type of the data is sint64	
	SINT64_N	Type of the data is sint64 array	
	SINT8	Type of the signal is sint8.	
	SINT8_N	Type of the signal is sint8 array.	
	UINT16	Type of the signal is uint16.	
	UINT16_N	Type of the signal is uint16 array.	
	UINT32	Type of the signal is uint32.	
	UINT32_N	Type of the signal is uint32 array.	
	UINT64	Type of the data is uint64.	
	UINT64_N	Type of the data is uint64 array	
	UINT8	Type of the signal is uint8.	
	UINT8_N	Type of the signal is uint8 array.	
	VARIABLE_LENGTH	Type of the signal is uint8[DcmDspRoutineParameterSize]. This is only valid for the last signal and when DcmDspRoutineSignalType is set to VARIABLE_LENGTH.	
	Post-Build Variant Value	false	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_01220] Definition of EcucReferenceDef DcmDspStartRoutineInSignalCompositeSignalRef

Parameter Name	DcmDspStartRoutineInSignalCompositeSignalRef		
Parent Container	DcmDspStartRoutineInSignal		
Description	Reference to the contained signal.		
Multiplicity	0..*		
Type	Reference to DcmDspStartRoutineInSignalCompositePool		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

10.2.5.19.12 DcmDspStartRoutineInSignalCompositePool

[ECUC_Dcm_01219] Definition of EcucParamConfContainerDef DcmDspStartRoutineInSignalCompositePool

Container Name	DcmDspStartRoutineInSignalCompositePool		
Parent Container	DcmDspStartRoutineIn		
Description	Defines a collection of all contained signals for this root signal.		
Multiplicity	0..*		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspStartRoutineInSignal	1..*	Provide description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataInN elements in the XXX_Start function call.

10.2.5.19.13 DcmDspStartRoutineOut

[ECUC_Dcm_00835] Definition of EcucParamConfContainerDef DcmDspStartRoutineOut

Container Name	DcmDspStartRoutineOut
Parent Container	DcmDspStartRoutine
Description	Provide description of output parameter of Start subservice for RoutineControl service.
Multiplicity	0..1
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspStartRoutineOutSignal	1..*	Provide description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataOutN elements in the XXX_Start function call.
DcmDspStartRoutineOutSignalCompositePool	0..*	Defines a collection of all contained signals for this root signal.

]

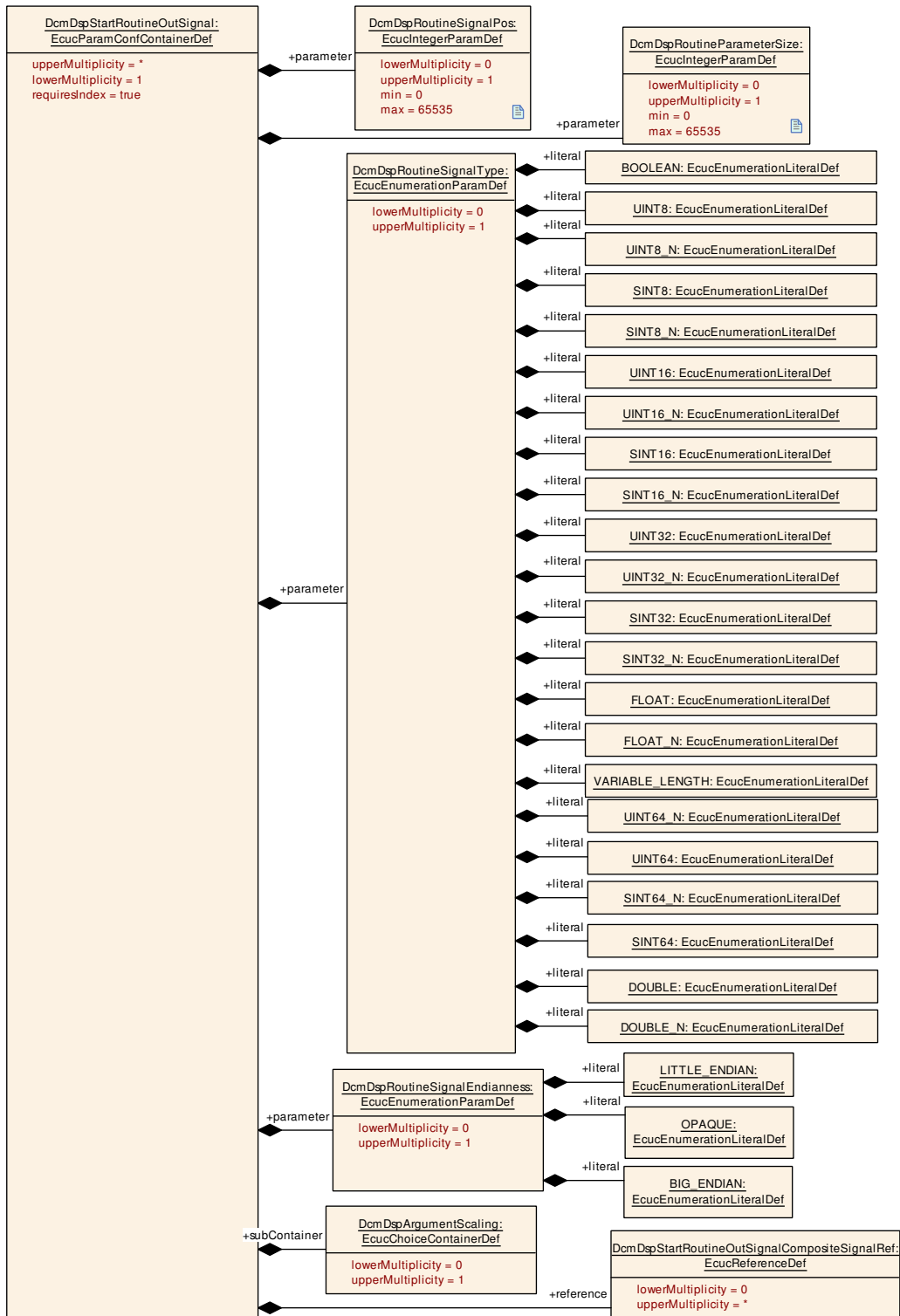


Figure 10.48: DcmDspStartRoutineOut configuration overview

10.2.5.19.14 DcmDspStartRoutineOutSignal

[ECUC_Dcm_00848] Definition of EcucParamConfContainerDef DcmDspStartRoutineOutSignal

Container Name	DcmDspStartRoutineOutSignal
Parent Container	DcmDspStartRoutineOut , DcmDspStartRoutineOutSignalCompositePool
Description	Provide description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataOutN elements in the XXX_Start function call. Attributes: requiresIndex=true
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspRoutineParameterSize	0..1	[ECUC_Dcm_00850]
DcmDspRoutineSignalEndianness	0..1	[ECUC_Dcm_01017]
DcmDspRoutineSignalPos	0..1	[ECUC_Dcm_00867]
DcmDspRoutineSignalType	0..1	[ECUC_Dcm_00885]
DcmDspStartRoutineOutSignalCompositeSignalRef	0..*	[ECUC_Dcm_01222]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspArgumentScaling	0..1	This container contains the configuration (arguments) of an alternative Diagnosis Representation. Out if this the scaling between Diagnosis and ECU internal representation and vice versa can be calculated.

]

[ECUC_Dcm_00850] Definition of EcucIntegerParamDef DcmDspRoutineParameterSize

Parameter Name	DcmDspRoutineParameterSize		
Parent Container	DcmDspStartRoutineOutSignal		
Description	Provide the size of a RoutineControl parameter in bytes		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME





	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01017] Definition of EcucEnumerationParamDef DcmDspRoutineSignalEndianness [

Parameter Name	DcmDspRoutineSignalEndianness		
Parent Container	DcmDspStartRoutineOutSignal		
Description	Defines the endianness of the data belonging to a Routine Out Signal for Start subfunction.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BIG_ENDIAN	Most significant byte shall be stored at the lowest address.	
	LITTLE_ENDIAN	Most significant byte shall be stored at the highest address	
	OPAQUE	Opaque data endianness	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_00867] Definition of EcucIntegerParamDef DcmDspRoutineSignalPos [

Parameter Name	DcmDspRoutineSignalPos		
Parent Container	DcmDspStartRoutineOutSignal		
Description	Provide the position of the signal in the RoutineControl request/response. The position is defined in bits.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	—		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	





Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00885] Definition of EcucEnumerationParamDef DcmDspRoutineSignalType

Parameter Name	DcmDspRoutineSignalType		
Parent Container	DcmDspStartRoutineOutSignal		
Description	Provide the type of the signal in the RoutineControl request/response.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BOOLEAN	Type of the signal is boolean.	
	DOUBLE	Type of the data is double (64 bit float)	
	DOUBLE_N	Type of the data is double (64 bit float) array	
	FLOAT	Type of the data is float.	
	FLOAT_N	Type of the data is float array.	
	SINT16	Type of the signal is sint16.	
	SINT16_N	Type of the signal is sint16 array.	
	SINT32	Type of the signal is sint32.	
	SINT32_N	Type of the signal is sint32 array.	
	SINT64	Type of the data is sint64	
	SINT64_N	Type of the data is sint64 array	
	SINT8	Type of the signal is sint8.	
	SINT8_N	Type of the signal is sint8 array.	
	UINT16	Type of the signal is uint16.	
	UINT16_N	Type of the signal is uint16 array.	
	UINT32	Type of the signal is uint32.	
	UINT32_N	Type of the signal is uint32 array.	
	UINT64	Type of the data is uint64.	
	UINT64_N	Type of the data is uint64 array	
	UINT8	Type of the signal is uint8.	
	UINT8_N	Type of the signal is uint8 array.	
	VARIABLE_LENGTH	Type of the signal is uint8[DcmDspRoutineParameterSize]. This is only valid for the last signal and when DcmDspRoutineSignalType is set to VARIABLE_LENGTH.	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	





Dependency	

]

[ECUC_Dcm_01222] Definition of EcucReferenceDef DcmDspStartRoutineOutSignalCompositeSignalRef [

Parameter Name	DcmDspStartRoutineOutSignalCompositeSignalRef		
Parent Container	DcmDspStartRoutineOutSignal		
Description	Reference to the contained signal.		
Multiplicity	0..*		
Type	Reference to DcmDspStartRoutineOutSignalCompositePool		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.19.15 DcmDspStopRoutine

[ECUC_Dcm_01022] Definition of EcucParamConfContainerDef DcmDspStopRoutine [

Container Name	DcmDspStopRoutine
Parent Container	DcmDspRoutine
Description	Provides the configuration of Stop subservice for RoutineControl service. Existence indicates that the StopRoutine in the RoutineControl is supported.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspRoutineInterfaceArgumentIntegrity	0..1	[ECUC_Dcm_01216]
DcmDspStopRoutineConfirmationEnabled	0..1	[ECUC_Dcm_01095]
DcmDspStopRoutineConfirmationFnc	0..1	[ECUC_Dcm_01096]
DcmDspStopRoutineFnc	0..1	[ECUC_Dcm_00752]
DcmDspStopRoutineCommonAuthorizationRef	0..1	[ECUC_Dcm_01053]
DcmDspStopRoutineRoleRef	0..32	[ECUC_Dcm_01145]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspStopRoutineIn	0..1	Provide description of input parameter of Stop subservice for RoutineControl service.
DcmDspStopRoutineOut	0..1	Provide description of output parameter of Stop subservice for RoutineControl service.

]

For parameter table [\[ECUC_Dcm_01216\] DcmDspRoutineInterfaceArgumentIntegrity](#), see definition below container [DcmDspRequestRoutineResults](#).

[ECUC_Dcm_01095] Definition of EcucBooleanParamDef DcmDspStopRoutineConfirmationEnabled [

Parameter Name	DcmDspStopRoutineConfirmationEnabled		
Parent Container	DcmDspStopRoutine		
Description	Allows to enable/disable the confirmation function to indicate the transmission of a response to a StopRoutine request		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01096] Definition of EcucFunctionNameDef DcmDspStopRoutineConfirmationFnc [

Parameter Name	DcmDspStopRoutineConfirmationFnc		
Parent Container	DcmDspStopRoutine		
Description	C-function to call if a transmission confirmation is needed by the issuer (BSW module)		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants





	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00752] Definition of EcucFunctionNameDef DcmDspStopRoutineFnc

Parameter Name	DcmDspStopRoutineFnc		
Parent Container	DcmDspStopRoutine		
Description	Function name for request to application to stop a routine. (Routine_Stop-function) This parameter is related to the interface Xxx_Stop.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01053] Definition of EcucReferenceDef DcmDspStopRoutineCommonAuthorizationRef

Parameter Name	DcmDspStopRoutineCommonAuthorizationRef		
Parent Container	DcmDspStopRoutine		
Description	Reference to DcmDspCommonAuthorization Common authorization configuration taken from the referenced DcmDspStopRoutineCommonAuthorizationRef. If there is no reference, no check on the commonly defined authorization conditions shall be done to stop the routine.		
Multiplicity	0..1		
Type	Reference to DcmDspCommonAuthorization		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	





Dependency	

]

[ECUC_Dcm_01145] Definition of EcucReferenceDef DcmDspStopRoutineRoleRef [

Parameter Name	DcmDspStopRoutineRoleRef		
Parent Container	DcmDspStopRoutine		
Description	Reference to DcmDspAuthenticationRow that defines a role in that this routine can be stopped.		
Multiplicity	0..32		
Type	Reference to DcmDspAuthenticationRow		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

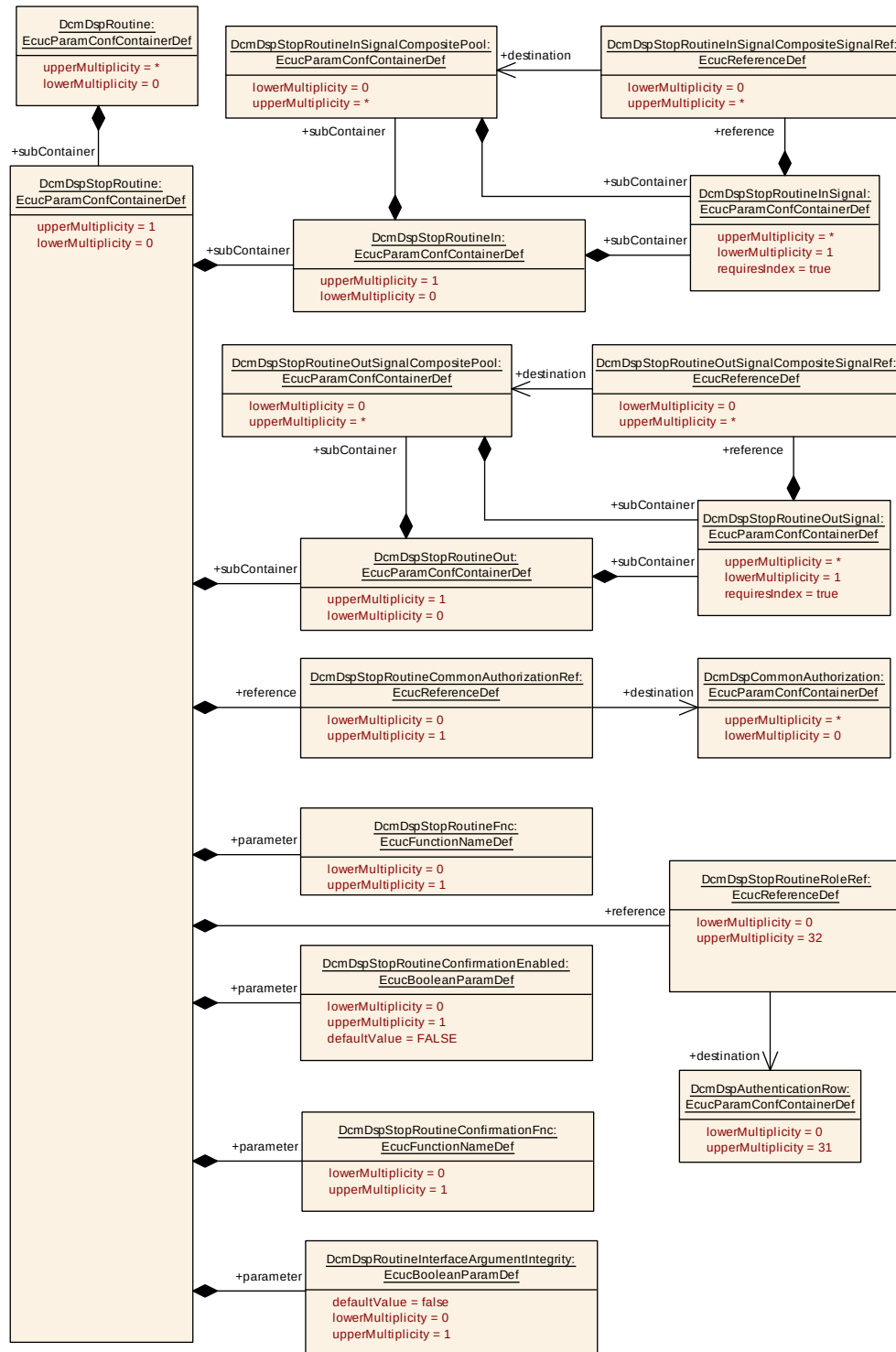


Figure 10.49: DcmDspStopRoutine configuration overview

10.2.5.19.16 DcmDspStopRoutineIn

[ECUC_Dcm_00832] Definition of EcucParamConfContainerDef DcmDspStopRoutineIn [

Container Name	DcmDspStopRoutineIn
Parent Container	DcmDspStopRoutine
Description	Provide description of input parameter of Stop subservice for RoutineControl service.
Multiplicity	0..1
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspStopRoutineInSignal	1..*	Provide description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataInN elements in the XXX_Stop function call.
DcmDspStopRoutineInSignal CompositePool	0..*	Defines a collection of all contained signals for this root signal.

]

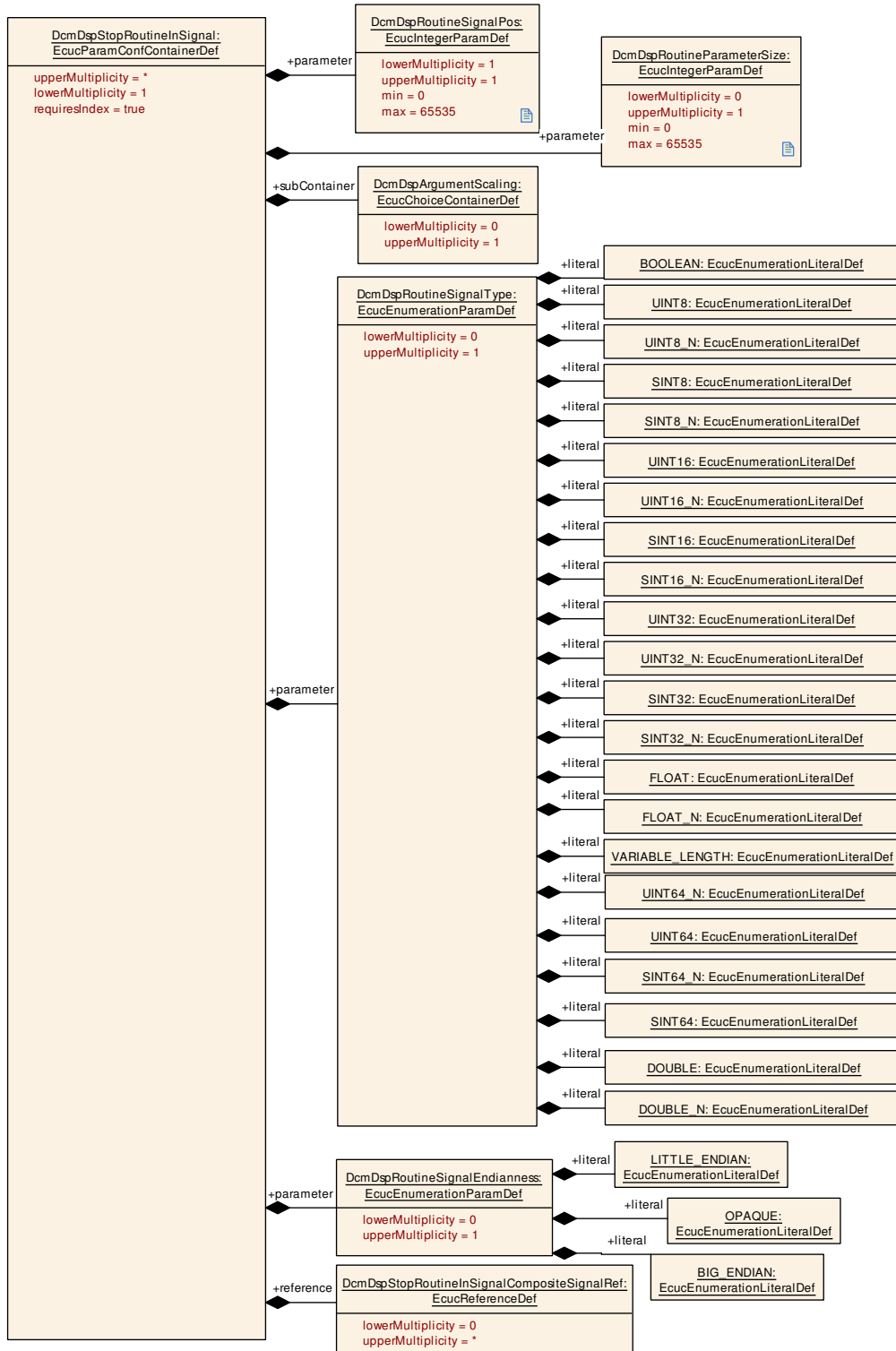


Figure 10.50: DcmDspRoutineStopIn configuration overview

10.2.5.19.17 DcmDspStartRoutineOutSignalCompositePool

[ECUC_Dcm_01221] Definition of EcucParamConfContainerDef DcmDspStartRoutineOutSignalCompositePool [

Container Name	DcmDspStartRoutineOutSignalCompositePool		
Parent Container	DcmDspStartRoutineOut		
Description	Defines a collection of all contained signals for this root signal.		
Multiplicity	0..*		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspStartRoutineOutSignal	1..*	Provide description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataOutN elements in the XXX_Start function call.

]

10.2.5.19.18 DcmDspStopRoutineInSignal

[ECUC_Dcm_00839] Definition of EcucParamConfContainerDef DcmDspStopRoutineInSignal [

Container Name	DcmDspStopRoutineInSignal
Parent Container	DcmDspStopRoutineIn , DcmDspStopRoutineInSignalCompositePool
Description	Provide description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataInN elements in the XXX_Stop function call. Attributes: requiresIndex=true
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspRoutineParameterSize	0..1	[ECUC_Dcm_00841]
DcmDspRoutineSignalEndianness	0..1	[ECUC_Dcm_01014]
DcmDspRoutineSignalPos	1	[ECUC_Dcm_00840]
DcmDspRoutineSignalType	0..1	[ECUC_Dcm_00882]
DcmDspStopRoutineInSignalCompositeSignalRef	0..*	[ECUC_Dcm_01224]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspArgumentScaling	0..1	This container contains the configuration (arguments) of an alternative Diagnosis Representation. Out if this the scaling between Diagnosis and ECU internal representation and vice versa can be calculated.

]

[ECUC_Dcm_00841] Definition of EcucIntegerParamDef DcmDspRoutineParameterSize

Parameter Name	DcmDspRoutineParameterSize		
Parent Container	DcmDspStopRoutineInSignal		
Description	Provide the size of a RoutineControl parameter in bytes		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01014] Definition of EcucEnumerationParamDef DcmDspRoutineSignalEndianness

Parameter Name	DcmDspRoutineSignalEndianness		
Parent Container	DcmDspStopRoutineInSignal		
Description	Defines the endianness of the data belonging to a Routine In Signal for Stop subfunction.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BIG_ENDIAN		Most significant byte shall be stored at the lowest address.
	LITTLE_ENDIAN		Most significant byte shall be stored at the highest address
	OPAQUE		Opaque data endianness
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME





	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00840] Definition of EcucIntegerParamDef DcmDspRoutineSignalPos

Parameter Name	DcmDspRoutineSignalPos		
Parent Container	DcmDspStopRoutineInSignal		
Description	Provide the position of the signal in the RoutineControl request/response. The position is defined in bits.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00882] Definition of EcucEnumerationParamDef DcmDspRoutineSignalType

Parameter Name	DcmDspRoutineSignalType	
Parent Container	DcmDspStopRoutineInSignal	
Description	Provide the type of the signal in the RoutineControl request/response.	
Multiplicity	0..1	
Type	EcucEnumerationParamDef	
Range	BOOLEAN	Type of the signal is boolean.
	DOUBLE	Type of the data is double (64 bit float)
	DOUBLE_N	Type of the data is double (64 bit float) array
	FLOAT	Type of the data is float.
	FLOAT_N	Type of the data is float array.
	SINT16	Type of the signal is sint16.
	SINT16_N	Type of the signal is sint16 array.
	SINT32	Type of the signal is sint32.
	SINT32_N	Type of the signal is sint32 array.
	SINT64	Type of the data is sint64
	SINT64_N	Type of the data is sint64 array





	SINT8	Type of the signal is sint8.	
	SINT8_N	Type of the signal is sint8 array.	
	UINT16	Type of the signal is uint16.	
	UINT16_N	Type of the signal is uint16 array.	
	UINT32	Type of the signal is uint32.	
	UINT32_N	Type of the signal is uint32 array.	
	UINT64	Type of the data is uint64.	
	UINT64_N	Type of the data is uint64 array	
	UINT8	Type of the signal is uint8.	
	UINT8_N	Type of the signal is uint8 array.	
	VARIABLE_LENGTH	Type of the signal is uint8[DcmDspRoutineParameterSize]. This is only valid for the last signal and when DcmDspRoutineSignalType is set to VARIABLE_LENGTH.	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_01224] Definition of EcucReferenceDef DcmDspStopRoutineInSignalCompositeSignalRef

Parameter Name	DcmDspStopRoutineInSignalCompositeSignalRef		
Parent Container	DcmDspStopRoutineInSignal		
Description	Reference to the contained signal.		
Multiplicity	0..*		
Type	Reference to DcmDspStopRoutineInSignalCompositePool		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.19.19 DcmDspStopRoutineInSignalCompositePool

[ECUC_Dcm_01223] Definition of EcucParamConfContainerDef DcmDspStopRoutineInSignalCompositePool

Container Name	DcmDspStopRoutineInSignalCompositePool		
Parent Container	DcmDspStopRoutineIn		
Description	Defines a collection of all contained signals for this root signal.		
Multiplicity	0..*		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspStopRoutineInSignal	1..*	Provide description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataInN elements in the XXX_Stop function call.

10.2.5.19.20 DcmDspStopRoutineOut

[ECUC_Dcm_00833] Definition of EcucParamConfContainerDef DcmDspStopRoutineOut

Container Name	DcmDspStopRoutineOut
Parent Container	DcmDspStopRoutine
Description	Provide description of output parameter of Stop subservice for RoutineControl service.
Multiplicity	0..1
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspStopRoutineOutSignal	1..*	Provide description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataOutN elements in the XXX_Stop function call.
DcmDspStopRoutineOutSignalCompositePool	0..*	Defines a collection of all contained signals for this root signal.

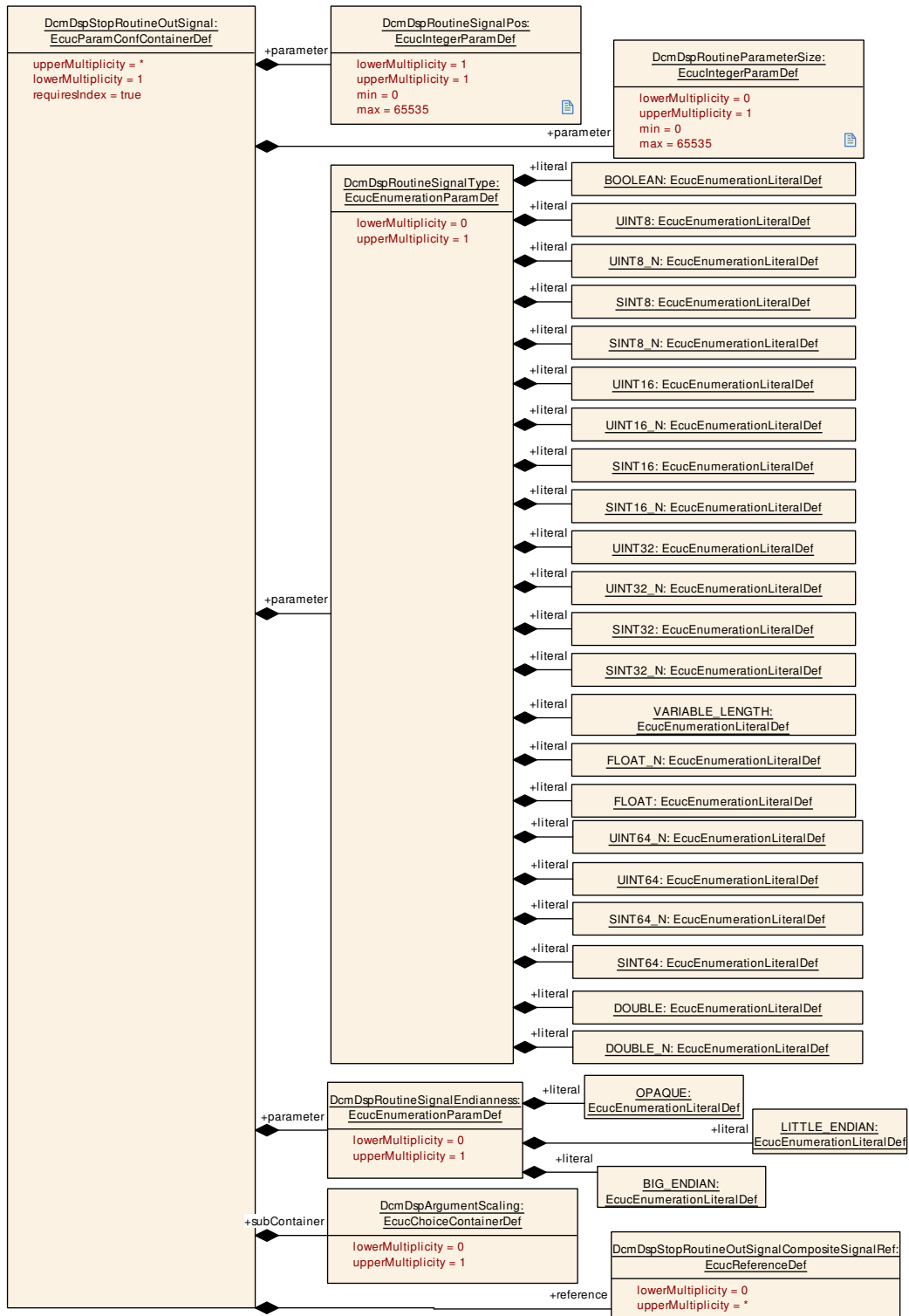


Figure 10.51: DcmDspStopRoutineOut configuration overview

10.2.5.19.21 DcmDspStopRoutineOutSignal

[ECUC_Dcm_00842] Definition of EcucParamConfContainerDef DcmDspStopRoutineOutSignal

Container Name	DcmDspStopRoutineOutSignal
Parent Container	DcmDspStopRoutineOut , DcmDspStopRoutineOutSignalCompositePool
Description	Provide description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataOutN elements in the XXX_Stop function call. Attributes: requiresIndex=true
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspRoutineParameterSize	0..1	[ECUC_Dcm_00844]
DcmDspRoutineSignalEndianness	0..1	[ECUC_Dcm_01015]
DcmDspRoutineSignalPos	1	[ECUC_Dcm_00843]
DcmDspRoutineSignalType	0..1	[ECUC_Dcm_00883]
DcmDspStopRoutineOutSignalCompositeSignalRef	0..*	[ECUC_Dcm_01226]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspArgumentScaling	0..1	This container contains the configuration (arguments) of an alternative Diagnosis Representation. Out if this the scaling between Diagnosis and ECU internal representation and vice versa can be calculated.

]

[ECUC_Dcm_00844] Definition of EcucIntegerParamDef DcmDspRoutineParameterSize

Parameter Name	DcmDspRoutineParameterSize		
Parent Container	DcmDspStopRoutineOutSignal		
Description	Provide the size of a RoutineControl parameter in bytes		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	—		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME





	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01015] Definition of EcucEnumerationParamDef DcmDspRoutineSignalEndianness

Parameter Name	DcmDspRoutineSignalEndianness		
Parent Container	DcmDspStopRoutineOutSignal		
Description	Defines the endianness of the data belonging to a Routine Out Signal for Stop subfunction.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BIG_ENDIAN	Most significant byte shall be stored at the lowest address.	
	LITTLE_ENDIAN	Most significant byte shall be stored at the highest address	
	OPAQUE	Opaque data endianness	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

└

[ECUC_Dcm_00843] Definition of EcucIntegerParamDef DcmDspRoutineSignalPos

Parameter Name	DcmDspRoutineSignalPos		
Parent Container	DcmDspStopRoutineOutSignal		
Description	Provide the position of the signal in the RoutineControl request/response. The position is defined in bits.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00883] Definition of EcucEnumerationParamDef DcmDspRoutineSignalType

Parameter Name	DcmDspRoutineSignalType		
Parent Container	DcmDspStopRoutineOutSignal		
Description	Provide the type of the signal in the RoutineControl request/response.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BOOLEAN	Type of the signal is boolean.	
	DOUBLE	Type of the data is double (64 bit float)	
	DOUBLE_N	Type of the data is double (64 bit float) array	
	FLOAT	Type of the data is float.	
	FLOAT_N	Type of the data is float array.	
	SINT16	Type of the signal is sint16.	
	SINT16_N	Type of the signal is sint16 array.	
	SINT32	Type of the signal is sint32.	
	SINT32_N	Type of the signal is sint32 array.	
	SINT64	Type of the data is sint64	
	SINT64_N	Type of the data is sint64 array	
	SINT8	Type of the signal is sint8.	
	SINT8_N	Type of the signal is sint8 array.	
	UINT16	Type of the signal is uint16.	
	UINT16_N	Type of the signal is uint16 array.	
	UINT32	Type of the signal is uint32.	
	UINT32_N	Type of the signal is uint32 array.	
	UINT64	Type of the data is uint64.	
	UINT64_N	Type of the data is uint64 array	
	UINT8	Type of the signal is uint8.	
	UINT8_N	Type of the signal is uint8 array.	
	VARIABLE_LENGTH	Type of the signal is uint8[DcmDspRoutineParameterSize]. This is only valid for the last signal and when DcmDspRoutineSignalType is set to VARIABLE_LENGTH.	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_01226] Definition of EcucReferenceDef DcmDspStopRoutineOutSignalCompositeSignalRef [

Parameter Name	DcmDspStopRoutineOutSignalCompositeSignalRef		
Parent Container	DcmDspStopRoutineOutSignal		
Description	Reference to the contained signal.		
Multiplicity	0..*		
Type	Reference to DcmDspStopRoutineOutSignalCompositePool		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.19.22 DcmDspStopRoutineOutSignalCompositePool

[ECUC_Dcm_01225] Definition of EcucParamConfContainerDef DcmDspStopRoutineOutSignalCompositePool [

Container Name	DcmDspStopRoutineOutSignalCompositePool		
Parent Container	DcmDspStopRoutineOut		
Description	Defines a collection of all contained signals for this root signal.		
Multiplicity	0..*		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspStopRoutineOutSignal	1..*	Provide description of a routine signal used in RoutineControl service. The ordering defined via the index attribute of the subcontainers in this list represents the order of the dataOutN elements in the XXX_Stop function call.

]

10.2.5.20 Session Security and Modes

10.2.5.20.1 DcmDspSecurity

[ECUC_Dcm_00764] Definition of EcucParamConfContainerDef DcmDspSecurity

Container Name	DcmDspSecurity		
Parent Container	DcmDsp		
Description	This container contains the configuration (DSP parameter) for security level configuration (per security level) Description This container contains Rows of DcmDsp SecurityRow		
Multiplicity	1		
Configuration Parameters			
Included Parameters			
Parameter Name	Multiplicity	ECUC ID	
DcmDspSecurityMaxAttemptCounterReadoutTime	1	[ECUC_Dcm_01101]	
DcmDspSecurityResetAttemptCounterOnTimeout	0..1	[ECUC_Dcm_01210]	
Included Containers			
Container Name	Multiplicity	Dependency	
DcmDspSecurityRow	0..31	Definition of a single Row of configuration for security level configuration (per security level) The name of this container is used to define the name of the R-Port through which the DCM accesses the interface SecurityAccess_{SecurityLevel}. The R-Port is named SecurityAccess_{SecurityLevel} where {SecurityLevel} is the name of the container DcmDspSecurity Row. If there is no reference, no check of security level shall be done.	

[ECUC_Dcm_01101] Definition of EcucFloatParamDef DcmDspSecurityMaxAttemptCounterReadoutTime

Parameter Name	DcmDspSecurityMaxAttemptCounterReadoutTime		
Parent Container	DcmDspSecurity		
Description	Delay, in seconds, from startup (measured from the first call of the Dcm_Main Function()), allowed for all AttemptCounter values to be obtained from the Application. Must be a multiple of the DcmTaskTime. min: A value equal to the DcmTaskTime		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	]0 .. 65535[		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_01210] Definition of EcucBooleanParamDef DcmDspSecurityReset AttemptCounterOnTimeout [

Parameter Name	DcmDspSecurityResetAttemptCounterOnTimeout		
Parent Container	DcmDspSecurity		
Description	This configuration parameter shall control the resetting behavior of SecurityAccess AttemptCounter upon Delay Timer expiration. If this parameter is false, then it attempts the SecurityAccess AttemptCounter not to reset when the delay timer expires. And if the parameter is true, then it attempts the SecurityAccess AttemptCounter to reset when the delay timer expires.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

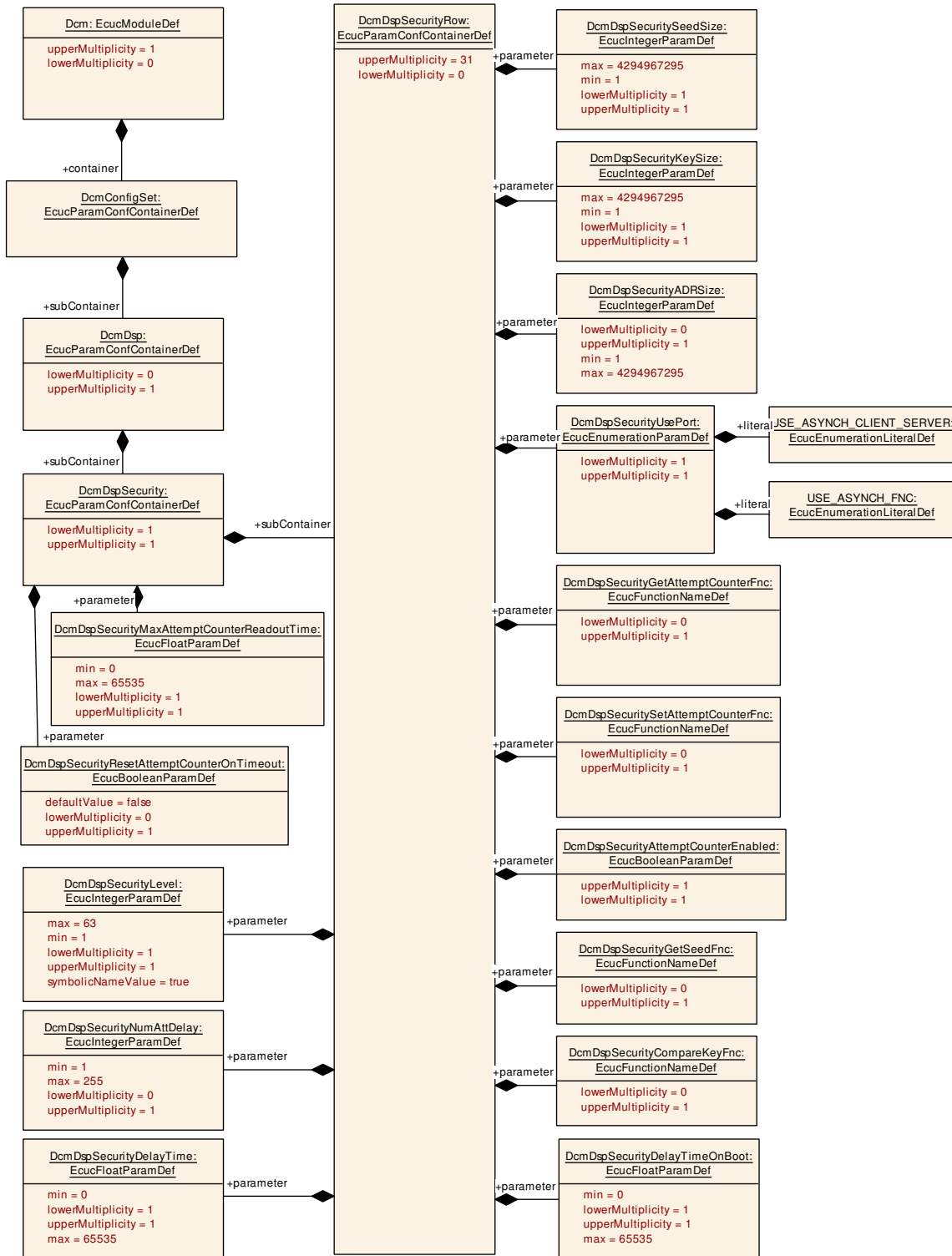


Figure 10.52: DcmDspSecurity configuration overview

10.2.5.20.2 DcmDspSecurityRow

[ECUC_Dcm_00759] Definition of EcucParamConfContainerDef DcmDspSecurityRow

Container Name	DcmDspSecurityRow
Parent Container	DcmDspSecurity
Description	Definition of a single Row of configuration for security level configuration (per security level) The name of this container is used to define the name of the R-Port through which the DCM accesses the interface SecurityAccess_{SecurityLevel}. The R-Port is named SecurityAccess_{SecurityLevel} where {SecurityLevel} is the name of the container DcmDspSecurityRow. If there is no reference, no check of security level shall be done.
Multiplicity	0..31
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspSecurityADRSIZE	0..1	[ECUC_Dcm_00725]
DcmDspSecurityAttemptCounterEnabled	1	[ECUC_Dcm_01050]
DcmDspSecurityCompareKeyFnc	0..1	[ECUC_Dcm_00969]
DcmDspSecurityDelayTime	1	[ECUC_Dcm_00757]
DcmDspSecurityDelayTimeOnBoot	1	[ECUC_Dcm_00726]
DcmDspSecurityGetAttemptCounterFnc	0..1	[ECUC_Dcm_01048]
DcmDspSecurityGetSeedFnc	0..1	[ECUC_Dcm_00968]
DcmDspSecurityKeySize	1	[ECUC_Dcm_00760]
DcmDspSecurityLevel	1	[ECUC_Dcm_00754]
DcmDspSecurityNumAttDelay	0..1	[ECUC_Dcm_00762]
DcmDspSecuritySeedSize	1	[ECUC_Dcm_00755]
DcmDspSecuritySetAttemptCounterFnc	0..1	[ECUC_Dcm_01049]
DcmDspSecurityUsePort	1	[ECUC_Dcm_00967]

No Included Containers

]

[ECUC_Dcm_00725] Definition of EcucIntegerParamDef DcmDspSecurityADRSIZE

Parameter Name	DcmDspSecurityADRSIZE
Parent Container	DcmDspSecurityRow
Description	Size in bytes of the AccessDataRecord used in GetSeed
Multiplicity	0..1
Type	EcucIntegerParamDef
Range	1 .. 4294967295
Default value	–
Post-Build Variant Multiplicity	false
Post-Build Variant Value	false





Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01050] Definition of EcucBooleanParamDef DcmDspSecurityAttemptCounterEnabled

Parameter Name	DcmDspSecurityAttemptCounterEnabled		
Parent Container	DcmDspSecurityRow		
Description	This configuration parameter controls the existence of the APIs to set / get the attempt counter values towards application (Xxx_SetSecurityAttemptCounter() / Xxx_GetSecurityAttemptCounter()). In case of enabled, the security attempt counter values are passed to application, whenever there is a change in the value. This allows storing the values in nonvolatile RAM and restoring them at ECU startup.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00969] Definition of EcucFunctionNameDef DcmDspSecurityCompareKeyFnc

Parameter Name	DcmDspSecurityCompareKeyFnc		
Parent Container	DcmDspSecurityRow		
Description	Function name to request the result of a key comparison. Parameter is only relevant if DcmDspSecurityUsePort=="USE_ASYNC_FNC". This parameter is related to the interface Xxx_CompareKey.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME





	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00757] Definition of EcucFloatParamDef DcmDspSecurityDelay Time

Parameter Name	DcmDspSecurityDelayTime		
Parent Container	DcmDspSecurityRow		
Description	Delay time after failed security access in seconds. This is started after DcmDspSecurityNumAttDelay number of failed security accesses. min: A negative value is not allowed.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 65535]		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00726] Definition of EcucFloatParamDef DcmDspSecurityDelay TimeOnBoot

Parameter Name	DcmDspSecurityDelayTimeOnBoot		
Parent Container	DcmDspSecurityRow		
Description	Value of the delay timer in case of 'power on' in seconds. This delay indicates the time at ECU boot power-on time during which the Dcm does not accept a security access. min: A negative value is not allowed.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 65535]		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01048] Definition of EcucFunctionNameDef DcmDspSecurityGetAttemptCounterFnc

Parameter Name	DcmDspSecurityGetAttemptCounterFnc		
Parent Container	DcmDspSecurityRow		
Description	Function name to request the value of an attempt counter. This parameter is related to the interface Xxx_GetSecurityAttemptCounter.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00968] Definition of EcucFunctionNameDef DcmDspSecurityGetSeedFnc

Parameter Name	DcmDspSecurityGetSeedFnc		
Parent Container	DcmDspSecurityRow		
Description	Callout function name used to request a seed. This parameter is related to the interface Xxx_GetSeed.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00760] Definition of EcucIntegerParamDef DcmDspSecurityKey Size

Parameter Name	DcmDspSecurityKeySize		
Parent Container	DcmDspSecurityRow		
Description	size of the security key (in Bytes).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_00754] Definition of EcucIntegerParamDef DcmDspSecurityLevel

Parameter Name	DcmDspSecurityLevel		
Parent Container	DcmDspSecurityRow		
Description	Value of Security level. The locked state cannot be configured explicitly. 1,2,3...63: configuration dependent - Conversion formula to calculate SecurityLevel out of tester requested SecurityAccessType parameter: $\text{SecurityLevel} = (\text{SecurityAccessType} (\text{requestSeed}) + 1) / 2$ Type: Dcm_SecLevelType		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	1 .. 63		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00762] Definition of EcucIntegerParamDef DcmDspSecurityNumAtt Delay

Parameter Name	DcmDspSecurityNumAttDelay		
Parent Container	DcmDspSecurityRow		
Description	Number of failed security accesses after which the delay time is activated		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	–		





Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00755] Definition of EcucIntegerParamDef DcmDspSecuritySeed Size

Parameter Name	DcmDspSecuritySeedSize		
Parent Container	DcmDspSecurityRow		
Description	size of the security seed (in Bytes).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01049] Definition of EcucFunctionNameDef DcmDspSecuritySetAttemptCounterFnc

Parameter Name	DcmDspSecuritySetAttemptCounterFnc		
Parent Container	DcmDspSecurityRow		
Description	Function name to set the value of an attempt counter. This parameter is related to the interface Xxx_SetSecurityAttemptCounter.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	





Dependency	

└

[ECUC_Dcm_00967] Definition of EcucEnumerationParamDef DcmDspSecurityUsePort

Parameter Name	DcmDspSecurityUsePort		
Parent Container	DcmDspSecurityRow		
Description	Defines which kind of interface shall be used for security access.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	USE_ASYNC_CLIENT_SERVER	The DCM will access the data using an R-Port requiring a asynchronous ClientServerInterface SecurityAccess_{SecurityLevel}. The R-Port is described in DcmDspSecurityRow description.	
	USE_ASYNC_FNC	The DCM will access the data using the functions that are defined in the parameters DcmDspSecurityGetSeedFnc and DcmDspSecurityCompareKeyFnc as well as the functions defined in DcmDspSecurityGetAttemptCounterFnc and DcmDspSecuritySetAttemptCounterFnc, if enabled by the parameter DcmDspSecurityAttemptCounterEnabled. DCM_E_PENDING return is allowed and Op Status is existing as IN parameter.	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

10.2.5.20.3 DcmDspSession

[ECUC_Dcm_00769] Definition of EcucParamConfContainerDef DcmDspSession

Container Name	DcmDspSession
Parent Container	DcmDsp
Description	Parent container holding single rows to configure particular sessions
Multiplicity	1
Configuration Parameters	
No Included Parameters	

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspSessionRow	0..31	This container holds all parameters needed to configure a single session

└

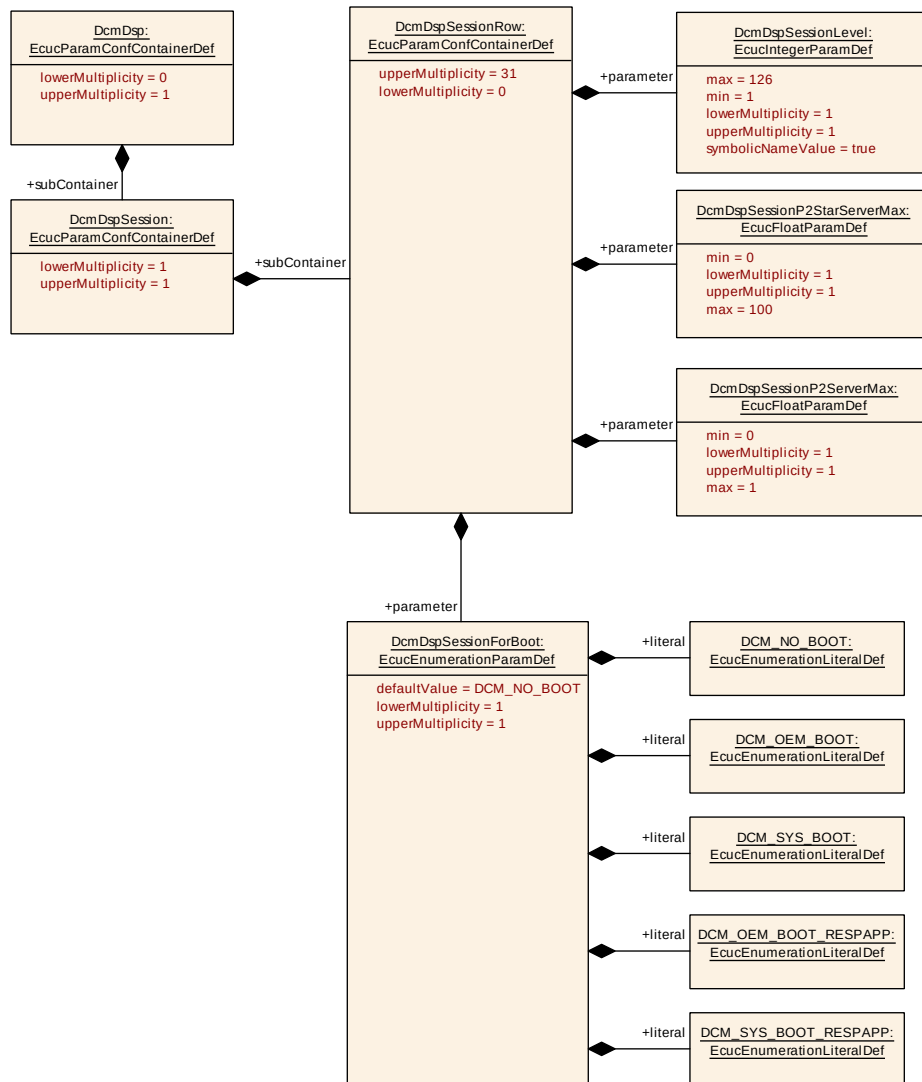


Figure 10.53: DcmDspSession configuration overview

10.2.5.20.4 DcmDspSessionRow

[ECUC_Dcm_00767] Definition of EcucParamConfContainerDef DcmDspSession Row

Container Name	DcmDspSessionRow
Parent Container	DcmDspSession
Description	This container holds all parameters needed to configure a single session
Multiplicity	0..31
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspSessionForBoot	1	[ECUC_Dcm_00815]
DcmDspSessionLevel	1	[ECUC_Dcm_00765]
DcmDspSessionP2ServerMax	1	[ECUC_Dcm_00766]
DcmDspSessionP2StarServerMax	1	[ECUC_Dcm_00768]

No Included Containers

]

[ECUC_Dcm_00815] Definition of EcucEnumerationParamDef DcmDspSessionForBoot [

Parameter Name	DcmDspSessionForBoot		
Parent Container	DcmDspSessionRow		
Description	This parameter defines whether this diagnostic session allows to jump to Bootloader (OEM Bootloader or System Supplier Bootloader) and determines, from which unit the final response will be sent. If this diagnostic session doesn't allow to jump to Bootloader the value DCM_NO_BOOT shall be chosen.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DCM_NO_BOOT	This diagnostic session doesn't allow to jump to Bootloader.	
	DCM_OEM_BOOT	This diagnostic session allows to jump to OEM Bootloader and bootloader sends final response.	
	DCM_OEM_BOOT_RESPAPP	This diagnostic session allows to jump to OEM Bootloader and application sends final response.	
	DCM_SYS_BOOT	This diagnostic session allows to jump to System Supplier Bootloader and bootloader sends final response.	
	DCM_SYS_BOOT_RESPAPP	This diagnostic session allows to jump to System Supplier Bootloader and application sends final response.	
Default value	DCM_NO_BOOT		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_00765] Definition of EcucIntegerParamDef DcmDspSessionLevel [

Parameter Name	DcmDspSessionLevel		
Parent Container	DcmDspSessionRow		
Description	subFunction value of the DiagnosticSession. 0, 127 and all values above 127 are reserved by ISO		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	1 .. 126		
Default value	—		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	—	
	Post-build time	—	
Dependency			

]

[ECUC_Dcm_00766] Definition of EcucFloatParamDef DcmDspSessionP2ServerMax [

Parameter Name	DcmDspSessionP2ServerMax		
Parent Container	DcmDspSessionRow		
Description	This is the session value for P2ServerMax in seconds (per Session). The AUTOSAR configuration standard is to use SI units, so this parameter is defined as a float value in seconds. DCM configuration tools must convert this float value to the appropriate value format for the use in the software implementation of DCM. This value is reported to the tester within the response to the 'Session Control' service.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 1]		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00768] Definition of EcucFloatParamDef DcmDspSessionP2StarServerMax [

Parameter Name	DcmDspSessionP2StarServerMax		
Parent Container	DcmDspSessionRow		
Description	This is the session value for P2*ServerMax in seconds (per Session). The AUTOSAR configuration standard is to use SI units, so this parameter is defined as a float value in seconds. DCM configuration tools must convert this float value to the appropriate value format for the use in the software implementation of DCM. This value is reported to the tester within the response to the 'Session Control' service.		
Multiplicity	1		





Type	EcucFloatParamDef		
Range	[0 .. 100]		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

10.2.5.20.5 DcmModeCondition

[ECUC_Dcm_00928] Definition of EcucParamConfContainerDef DcmModeCondition

Container Name	DcmModeCondition
Parent Container	DcmProcessingConditions
Description	This container contains the configuration of a mode condition or an environmental conditions which can be used as argument in DcmModeRules. One DcmModeCondition shall contain either one DcmSwcModeRef or one DcmBswModeRef or one DcmSwcSRDataElementRef. Please note that the Dcm acts as well as mode manager. Therefore the references DcmSwcModeRef or one DcmBswModeRef. might point to provided ModeDeclaration GroupPrototypes of the Dcm itself as well as to provided ModeDeclarationGroup Prototypes of other Bsw Modules or software components. In case of a configured DcmSwcModeRef or DcmBswModeRef only the DcmCondition Type DCM_EQUALS or DCM_EQUALS_NOT are applicable. In case of DcmSwcSRDataElementRef all literals of DcmConditionType are possible.
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmConditionType	1	[ECUC_Dcm_00929]
DcmBswModeRef	0..1	[ECUC_Dcm_00931]
DcmModeConditionCertificateCompareElementRef	0..1	[ECUC_Dcm_01179]
DcmSwcModeRef	0..1	[ECUC_Dcm_00930]
DcmSwcSRDataElementRef	0..1	[ECUC_Dcm_01037]

Included Containers		
Container Name	Multiplicity	Dependency
DcmSwcDataElementValue	1	This container contains the configuration of a compare value.

[ECUC_Dcm_00929] Definition of EcucEnumerationParamDef DcmCondition Type

Parameter Name	DcmConditionType		
Parent Container	DcmModeCondition		
Description	This parameter specifies what kind of comparison that is made for the evaluation of the mode condition.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DCM_EQUALS	—	
	DCM_EQUALS_NOT	—	
	DCM_GREATER_OR_EQUAL	—	
	DCM_GREATER_THAN	—	
	DCM_LESS_OR_EQUAL	—	
	DCM_LESS_THAN	—	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	—	
	Post-build time	—	
Dependency			

[ECUC_Dcm_00931] Definition of EcucInstanceReferenceDef DcmBswModeRef

Parameter Name	DcmBswModeRef		
Parent Container	DcmModeCondition		
Description	This parameter references a mode of a ModeDeclarationGroupPrototype provided by a Basic Software Module used for the condition. Please note that such ModeDeclarationGroupPrototype are owned by a Basic Software Module Description in the role providedModeGroup.		
Multiplicity	0..1		
Type	Instance reference to MODE-DECLARATION context: MODE-DECLARATION-GROUP-PROTOTYPE		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_01179] Definition of EcucReferenceDef DcmModeConditionCertificateCompareElementRef

Parameter Name	DcmModeConditionCertificateCompareElementRef		
Parent Container	DcmModeCondition		
Description	Reference to a certificate data element that provides the compare value.		
Multiplicity	0..1		
Type	Symbolic name reference to KeyMCertificateElement		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00930] Definition of EcucInstanceReferenceDef DcmSwcModeRef

Parameter Name	DcmSwcModeRef		
Parent Container	DcmModeCondition		
Description	This parameter references a mode in a particular mode request port of a software component that is used for the condition.		
Multiplicity	0..1		
Type	Instance reference to MODE-DECLARATION context: ROOT-SW-COMPOSITION-PROTOTYPE SW-COMPONENT-PROTOTYPE P-PORT-PROTOTYPE MODE-DECLARATION-GROUP-PROTOTYPE		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_01037] Definition of EcucChoiceReferenceDef DcmSwcSRDataElementRef

Parameter Name	DcmSwcSRDataElementRef		
Parent Container	DcmModeCondition		
Description	Reference to environmental conditions. It is possible to reference a S/R Receiver-Port to read physical values and compare (equal, greater, less,...) them with a configured value that is defined by DcmSwcDataElementValue.		





Multiplicity	0..1		
Type	Choice reference to [DcmDspExternalSRDataElementClass , DcmDspPidService01ExternalSRDataElementClass]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

┌

10.2.5.20.6 DcmSwcDataElementValue

[ECUC_Dcm_01123] Definition of EcucChoiceContainerDef DcmSwcDataElementValue ┌

Choice Container Name	DcmSwcDataElementValue
Parent Container	DcmModeCondition
Description	This container contains the configuration of a compare value.
Multiplicity	1

No Included Parameters

Container Choices		
Container Name	Multiplicity	Dependency
DcmSwcDataElementArray	0..1	This container contains the configuration of a array compare value.
DcmSwcDataElementPrimitive	0..1	This container contains the configuration of a primitive compare value.

└

10.2.5.20.7 DcmSwcDataElementPrimitive

[ECUC_Dcm_01124] Definition of EcucParamConfContainerDef DcmSwcDataElementPrimitive ┌

Container Name	DcmSwcDataElementPrimitive
Parent Container	DcmSwcDataElementValue
Description	This container contains the configuration of a primitive compare value.





Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmSwcDataElementPrimitiveValue	1	[ECUC_Dcm_01126]

No Included Containers

]

[ECUC_Dcm_01126] Definition of EcucIntegerParamDef DcmSwcDataElementPrimitiveValue [

Parameter Name	DcmSwcDataElementPrimitiveValue		
Parent Container	DcmSwcDataElementPrimitive		
Description	Primitive compare value.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 18446744073709551615		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.20.8 DcmSwcDataElementArray

[ECUC_Dcm_01125] Definition of EcucParamConfContainerDef DcmSwcDataElementArray [

Container Name	DcmSwcDataElementArray
Parent Container	DcmSwcDataElementValue
Description	This container contains the configuration of a array compare value.
Multiplicity	0..1
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Dependency
DcmSwcDataElementArrayElement	0..*	This container contains the configuration of a array element compare value.

]

10.2.5.20.9 DcmSwcDataElementArrayElement

[ECUC_Dcm_01129] Definition of EcucParamConfContainerDef DcmSwcDataElementArrayElement [

Container Name	DcmSwcDataElementArrayElement
Parent Container	DcmSwcDataElementArray
Description	This container contains the configuration of a array element compare value.
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmSwcDataElementArrayElementIndex	1	[ECUC_Dcm_01127]
DcmSwcDataElementArrayElementValue	1	[ECUC_Dcm_01128]

No Included Containers

]

[ECUC_Dcm_01127] Definition of EcucIntegerParamDef DcmSwcDataElementArrayElementIndex [

Parameter Name	DcmSwcDataElementArrayElementIndex		
Parent Container	DcmSwcDataElementArrayElement		
Description	Index to an element of the compare value array.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 18446744073709551615		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01128] Definition of EcucIntegerParamDef DcmSwcDataElementArrayElementValue [

Parameter Name	DcmSwcDataElementArrayElementValue		
Parent Container	DcmSwcDataElementArrayElement		
Description	Value of an array element compare value.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 18446744073709551615		
Default value	–		





Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

10.2.5.20.10 DcmModeRule

[ECUC_Dcm_00925] Definition of EcucParamConfContainerDef DcmModeRule [

Container Name	DcmModeRule		
Parent Container	DcmProcessingConditions		
Description	This container contains the configuration of a mode rule which represents a logical expression with DcmModeConditions or other DcmModeRules as arguments. All arguments are processed with the operator defined by DcmLogicalOperator, for instance: Argument_A AND Argument_B AND Argument_C		
Multiplicity	1..*		
Configuration Parameters			

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmLogicalOperator	0..1	[ECUC_Dcm_00926]
DcmModeRuleNrcValue	0..1	[ECUC_Dcm_00949]
DcmArgumentRef	1..*	[ECUC_Dcm_00927]

No Included Containers

]

[ECUC_Dcm_00926] Definition of EcucEnumerationParamDef DcmLogicalOperator [

Parameter Name	DcmLogicalOperator		
Parent Container	DcmModeRule		
Description	This parameter specifies the logical operator to be used in the logical expression. If the expression only consists of a single condition this parameter shall not be used.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DCM_AND	–	
	DCM_OR	–	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	





Value Configuration Class	Post-build time	–	
	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_00949] Definition of EcucIntegerParamDef DcmModeRuleNrcValue

Parameter Name	DcmModeRuleNrcValue		
Parent Container	DcmModeRule		
Description	Optional parameter which defines the NRC to be sent in case the mode rule condition is not valid.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_00927] Definition of EcucChoiceReferenceDef DcmArgumentRef

Parameter Name	DcmArgumentRef		
Parent Container	DcmModeRule		
Description	This is a choice reference either to a mode condition or a an other mode rule serving as sub-expression. Attributes: requiresIndex=true		
Multiplicity	1..*		
Type	Choice reference to [DcmModeCondition , DcmModeRule]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

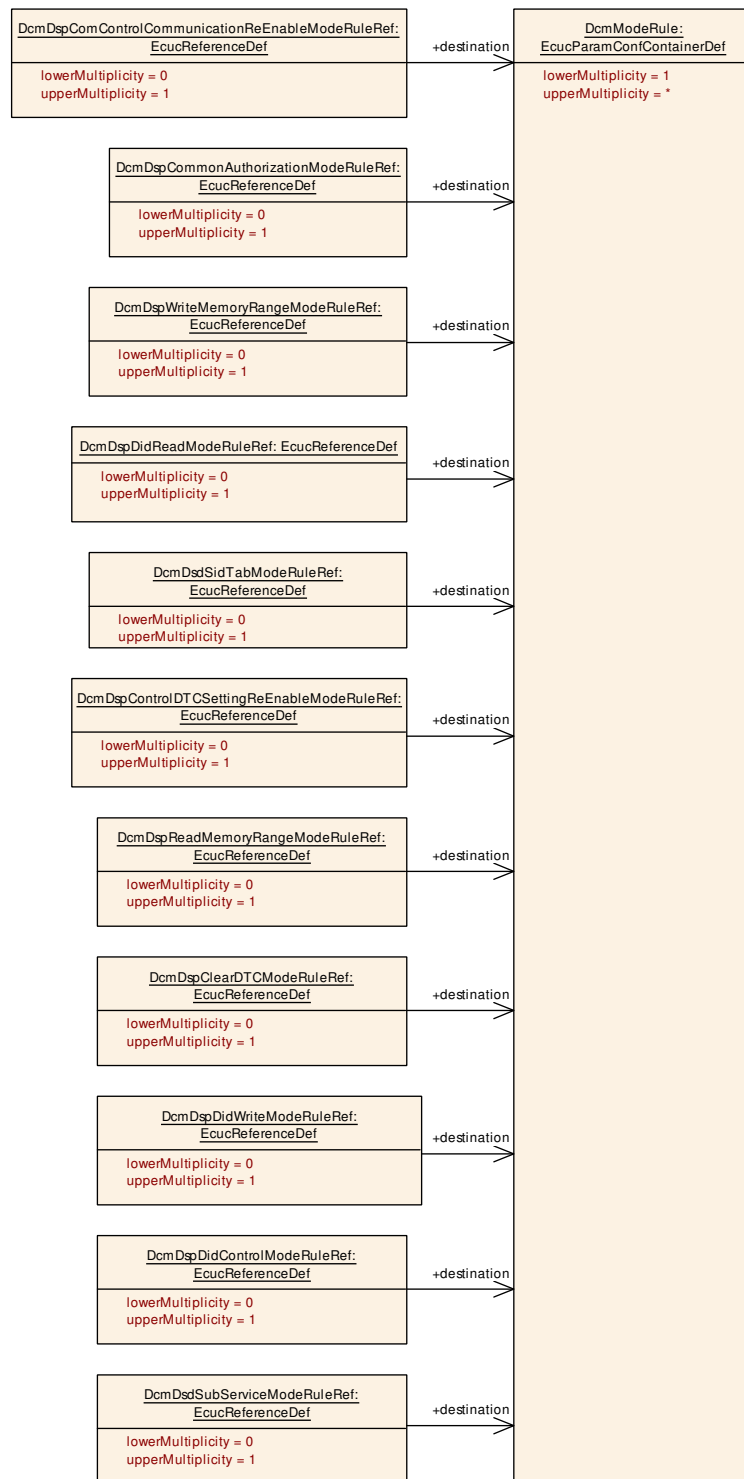


Figure 10.54: DcmModeRuleUsage configuration overview

10.2.5.21 DcmDspVehInfo

[ECUC_Dcm_00630] Definition of EcucParamConfContainerDef DcmDspVehInfo

[

Container Name	DcmDspVehInfo
Parent Container	DcmDsp
Description	This container contains the configuration (parameters) for one single VehicleInfoType of service \$09
Multiplicity	0..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspVehInfoInfoType	1	[ECUC_Dcm_00631]
DcmDspVehInfoNODIProvResp	0..1	[ECUC_Dcm_01051]

Included Containers		
Container Name	Multiplicity	Dependency
DcmDspVehInfoData	1..*	Data Item of an InfoType; ShortName is post-fix of the port interface name.

]

[ECUC_Dcm_00631] Definition of EcucIntegerParamDef DcmDspVehInfoInfoType [

Parameter Name	DcmDspVehInfoInfoType		
Parent Container	DcmDspVehInfo		
Description	value of InfoType. Within each DcmConfigSet all DcmDspVehInfoInfoType values shall be unique.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01051] Definition of EcucBooleanParamDef DcmDspVehInfoNODIProvResp [

Parameter Name	DcmDspVehInfoNODIProvResp		
Parent Container	DcmDspVehInfo		
Description	Indicate the Dcm, which side is responsible to fill the number of data items (NODI), Dcm or the provider of the InfoType data. In case the responsibility is on provider side, only one DcmDspVehInfoData container is allowed. • true: Provider is responsible for providing the number of data items parameter • false or not existing: Dcm is responsible for providing the number of data items parameter		





Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

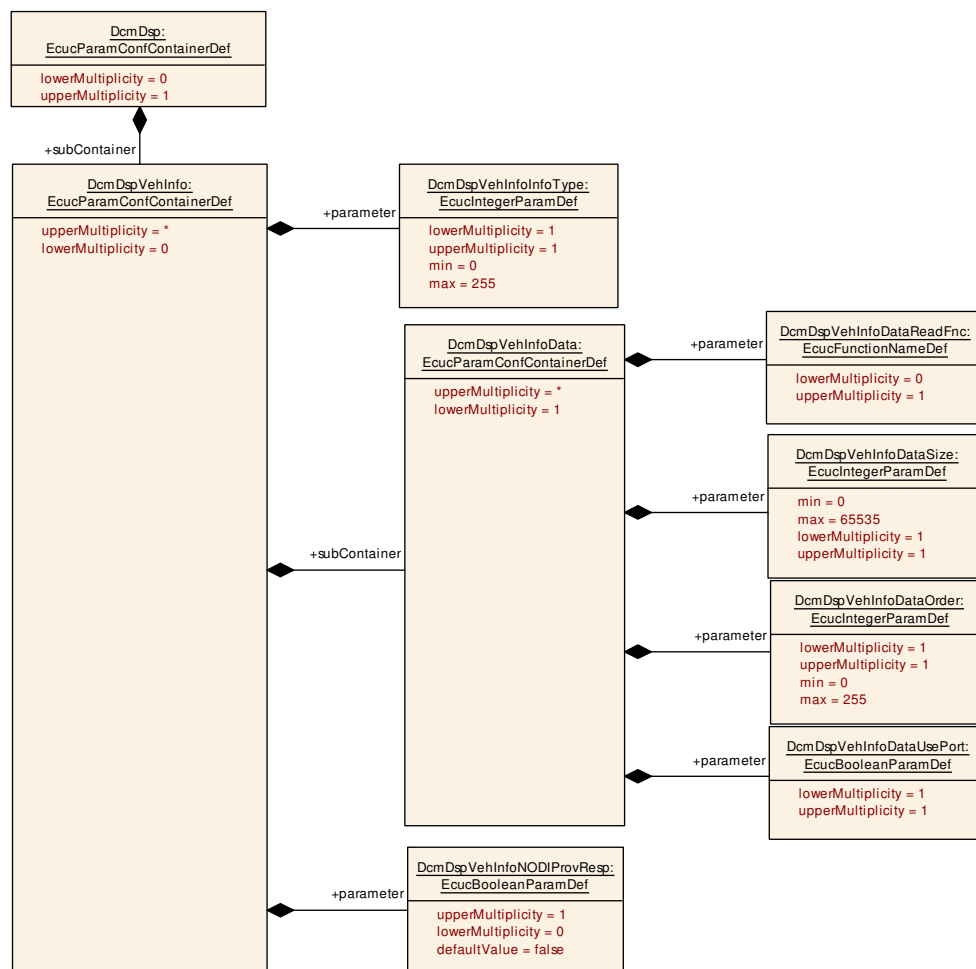


Figure 10.55: DcmDspVehInfo configuration overview

10.2.5.22 DcmDspVehInfoData

[ECUC_Dcm_00888] Definition of EcucParamConfContainerDef DcmDspVehInfo Data

Container Name	DcmDspVehInfoData
Parent Container	DcmDspVehInfo
Description	Data Item of an InfoType; ShortName is post-fix of the port interface name.
Multiplicity	1..*
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspVehInfoDataOrder	1	[ECUC_Dcm_00891]
DcmDspVehInfoDataReadFnc	0..1	[ECUC_Dcm_00889]
DcmDspVehInfoDataSize	1	[ECUC_Dcm_00890]
DcmDspVehInfoDataUsePort	1	[ECUC_Dcm_00727]

No Included Containers

]

[ECUC_Dcm_00891] Definition of EcucIntegerParamDef DcmDspVehInfoDataOrder [

Parameter Name	DcmDspVehInfoDataOrder		
Parent Container	DcmDspVehInfoData		
Description	Defines the order of the data item in the InfoType; values: 0..255; first data item having the order number 0; the next 1 and so on. The configuration of order needs to be unique per InfoType.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00889] Definition of EcucFunctionNameDef DcmDspVehInfoDataReadFnc [

Parameter Name	DcmDspVehInfoDataReadFnc		
Parent Container	DcmDspVehInfoData		
Description	Callout function name for reading InfoType data item. Only required in case parameter 'DcmDspVehInfoDataUsePort' is set to 'false'		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		





Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00890] Definition of EcucIntegerParamDef DcmDspVehInfoData Size

Parameter Name	DcmDspVehInfoDataSize		
Parent Container	DcmDspVehInfoData		
Description	Size in bytes of the InfoType data item.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00727] Definition of EcucBooleanParamDef DcmDspVehInfoData UsePort

Parameter Name	DcmDspVehInfoDataUsePort		
Parent Container	DcmDspVehInfoData		
Description	When this parameter is set to true the DCM will access the Data using an R-Port requiring a PortInterface IInfotypeServices_{VehInfoData}. The R-Port is named InfotypeServices_{VehInfoData} where {VEHINFODATA} is the name of the container DcmDspVehInfoData. In that case, the DcmDspVehInfoDataReadFnc is ignored and the RTE APIs are used. When this parameter is set to false, the DCM calls the function defined in DcmDspVehInfoDataReadFnc.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	





Dependency	

]

10.2.5.23 DcmDspPeriodicTransmission

[ECUC_Dcm_00957] Definition of EcucParamConfContainerDef DcmDspPeriodicTransmission [

Container Name	DcmDspPeriodicTransmission
Parent Container	DcmDsp
Description	This container contains the configuration (parameters) for Periodic Transmission Scheduler.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspMaxPeriodicDidScheduler	1	[ECUC_Dcm_00962]
DcmDspPeriodicTransmissionFastRate	0..1	[ECUC_Dcm_00960]
DcmDspPeriodicTransmissionMaxPeriodicFast Transmissions	0..1	[ECUC_Dcm_01182]
DcmDspPeriodicTransmissionMaxPeriodicMedium Transmissions	0..1	[ECUC_Dcm_01181]
DcmDspPeriodicTransmissionMaxPeriodicSlow Transmissions	0..1	[ECUC_Dcm_01183]
DcmDspPeriodicTransmissionMediumRate	0..1	[ECUC_Dcm_00959]
DcmDspPeriodicTransmissionSchedulerType	1	[ECUC_Dcm_01180]
DcmDspPeriodicTransmissionSlowRate	0..1	[ECUC_Dcm_00958]

No Included Containers

]

[ECUC_Dcm_00962] Definition of EcucIntegerParamDef DcmDspMaxPeriodicDidScheduler [

Parameter Name	DcmDspMaxPeriodicDidScheduler	
Parent Container	DcmDspPeriodicTransmission	
Description	Defines the maximum number of periodicDataIdentifiers that can be scheduled concurrently.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	1 .. 255	
Default value	–	
Post-Build Variant Value	false	





Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00960] Definition of EcucFloatParamDef DcmDspPeriodicTransmissionFastRate

Parameter Name	DcmDspPeriodicTransmissionFastRate		
Parent Container	DcmDspPeriodicTransmission		
Description	This parameter give the transmission rate of the requested periodicDataIdentifiers to be used if the parameter transmissionMode given in the ReadDataByPeriodicID request is equal to 0x03 ("sendAtFastRate"). This parameter value in seconds have to be configured as a multiple of DcmTaskTime. min: A negative value and zero is not allowed.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[1E-4 .. 1]		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01182] Definition of EcucIntegerParamDef DcmDspPeriodicTransmissionMaxPeriodicFastTransmissions

Parameter Name	DcmDspPeriodicTransmissionMaxPeriodicFastTransmissions		
Parent Container	DcmDspPeriodicTransmission		
Description	Defines the maximum number of periodic connections used for fast transmission.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	





Dependency	

]

[ECUC_Dcm_01181] Definition of EcucIntegerParamDef DcmDspPeriodicTransmissionMaxPeriodicMediumTransmissions [

Parameter Name	DcmDspPeriodicTransmissionMaxPeriodicMediumTransmissions		
Parent Container	DcmDspPeriodicTransmission		
Description	Defines the maximum number of periodic connections used for medium transmission.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01183] Definition of EcucIntegerParamDef DcmDspPeriodicTransmissionMaxPeriodicSlowTransmissions [

Parameter Name	DcmDspPeriodicTransmissionMaxPeriodicSlowTransmissions		
Parent Container	DcmDspPeriodicTransmission		
Description	Defines the maximum number of periodic connections used for slow transmission.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_00959] Definition of EcucFloatParamDef DcmDspPeriodicTransmissionMediumRate

Parameter Name	DcmDspPeriodicTransmissionMediumRate		
Parent Container	DcmDspPeriodicTransmission		
Description	This parameter give the transmission rate of the requested periodicDataIdentifiers to be used if the parameter transmissionMode given in the ReadDataByPeriodicID request is equal to 0x02 ("sendAtMediumRate"). This parameter value in seconds have to be configured as a multiple of DcmTaskTime. min: A negative value and zero is not allowed.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[1E-4 .. 1]		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

[ECUC_Dcm_01180] Definition of EcucEnumerationParamDef DcmDspPeriodicTransmissionSchedulerType

Parameter Name	DcmDspPeriodicTransmissionSchedulerType		
Parent Container	DcmDspPeriodicTransmission		
Description	Defines the used scheduler type according to ISO 14229-1:2018 for read data by periodic identifier.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	SCHEDULER_TYPE1	Selects the scheduler type#1 of ISO 14229-1:2018	
	SCHEDULER_TYPE2	Selects the scheduler type#2 of ISO 14229-1:2018	
Default value	SCHEDULER_TYPE1		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	—	
	Post-build time	—	
Dependency			

[ECUC_Dcm_00958] Definition of EcucFloatParamDef DcmDspPeriodicTransmissionSlowRate [

Parameter Name	DcmDspPeriodicTransmissionSlowRate		
Parent Container	DcmDspPeriodicTransmission		
Description	This parameter give the transmission rate of the requested periodicDataIdentifiers to be used if the parameter transmissionMode given in the ReadDataByPeriodicID request is equal to 0x01 ("sendAtSlowRate"). This parameter value in seconds have to be configured as a multiple of DcmTaskTime. min: A negative value and zero is not allowed.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[1E-4 .. 1]		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

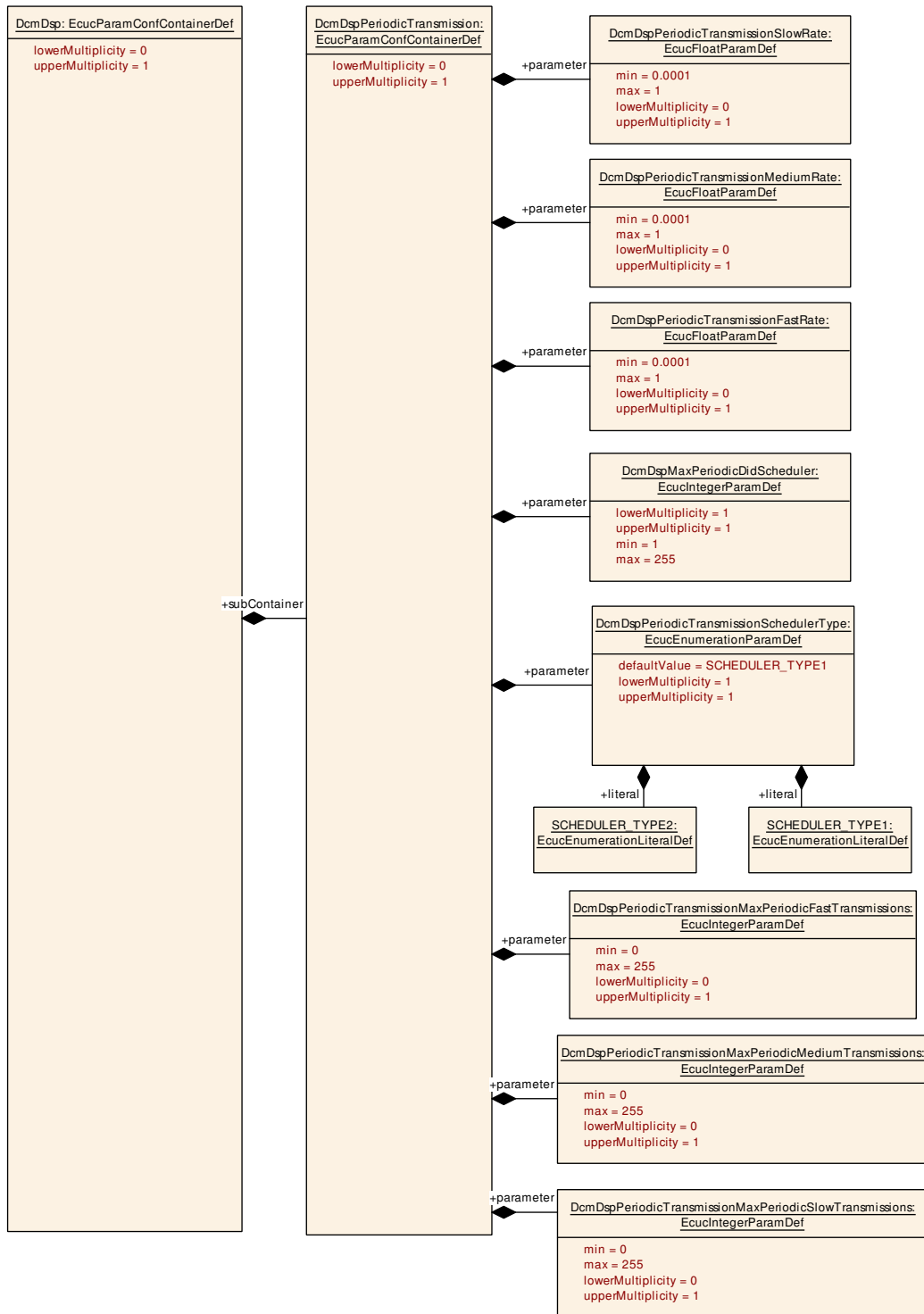


Figure 10.56: DcmDspPeriodicTransmission configuration overview

10.2.5.24 DcmDspClearDTC

[ECUC_Dcm_01064] Definition of EcucParamConfContainerDef DcmDspClearDTC

Container Name	DcmDspClearDTC
Parent Container	DcmDsp
Description	This container contains the configuration for the Clear DTC service.
Multiplicity	0..1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDspClearDTCCheckFnc	0..1	[ECUC_Dcm_01066]
DcmDspClearDTCModeRuleRef	0..1	[ECUC_Dcm_01065]

No Included Containers

]

[[ECUC_Dcm_01066](#)] Definition of EcucFunctionNameDef DcmDspClearDTC-CheckFnc [

Parameter Name	DcmDspClearDTCCheckFnc		
Parent Container	DcmDspClearDTC		
Description	Callback function for condition check, manufacturer / supplier specific checks on the groupOfDTC, which is requested to clear. This parameter is related to the interface : Xxx_ClearDTCCheckFnc.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[[ECUC_Dcm_01065](#)] Definition of EcucReferenceDef DcmDspClearDTCModeRuleRef [

Parameter Name	DcmDspClearDTCModeRuleRef
Parent Container	DcmDspClearDTC
Description	Reference to DcmModeRule Mode rule which controls to clear the DTCs. If there is no reference, no check of the mode rule shall be done.
Multiplicity	0..1
Type	Reference to DcmModeRule





Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

]

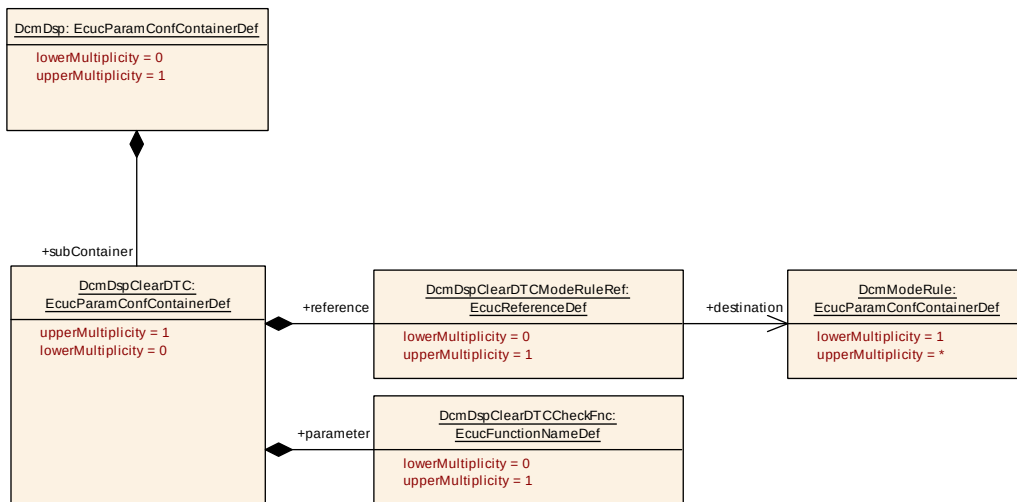


Figure 10.57: DcmDspClearDTC configuration overview

10.2.6 DcmGeneral

[ECUC_Dcm_00822] Definition of EcucParamConfContainerDef DcmGeneral [

Container Name	DcmGeneral
Parent Container	Dcm
Description	Contains general configuration parameters valid for the entire Dcm module.
Multiplicity	1
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmDDDIDStorage	0..1	[ECUC_Dcm_00971]
DcmDevErrorDetect	1	[ECUC_Dcm_00823]
DcmEnableSecurityEventReporting	1	[ECUC_Dcm_01187]





Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmHeaderFileInclusion	0..*	[ECUC_Dcm_01019]
DcmRespondAllRequest	1	[ECUC_Dcm_00600]
DcmS3ServerTimeoutOverwrite	0..1	[ECUC_Dcm_01240]
DcmTaskTime	1	[ECUC_Dcm_00820]
DcmVersionInfoApi	1	[ECUC_Dcm_00821]
DcmVinRef	0..1	[ECUC_Dcm_00984]

Included Containers		
Container Name	Multiplicity	Dependency
DcmSecureCoding	0..1	Contains references to diagnostic elements that are used for the secure coding process.
DcmSecurityEventRefs	0..1	Container for the references to IdsMEvent elements representing the security events that the Dcm module shall report to the IdsM in case the corresponding security related event occurs (and if DcmEnableSecurityEventReporting is set to "true"). The standardized security events in this container can be extended by vendor-specific security events. Tags: atp.Status=draft

└

[ECUC_Dcm_00971] Definition of EcucBooleanParamDef DcmDDDIDStorage ┌

Parameter Name	DcmDDDIDStorage		
Parent Container	DcmGeneral		
Description	This configuration switch defines, whether DDDID definition is stored non-volatile or not. true: DDDID are stored non-volatile false: DDDID are only maintained volatile		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00823] Definition of EcucBooleanParamDef DcmDevErrorDetect [

Parameter Name	DcmDevErrorDetect		
Parent Container	DcmGeneral		
Description	Switches the development error detection and notification on or off. • true: detection and notification is enabled. • false: detection and notification is disabled.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01187] Definition of EcucBooleanParamDef DcmEnableSecurityEventReporting

Status: DRAFT

[

Parameter Name	DcmEnableSecurityEventReporting		
Parent Container	DcmGeneral		
Description	Switches the reporting of security events to the IdsM: - true: reporting is enabled. - false: reporting is disabled. Tags: atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01019] Definition of EcucStringParamDef DcmHeaderFileInclusion [

Parameter Name	DcmHeaderFileInclusion		
Parent Container	DcmGeneral		
Description	Name of the header file(s) to be included by the Dcm module containing the used C-callback declarations.		
Multiplicity	0..*		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	[a-zA-Z0-9_]([a-zA-Z0-9_])*		





Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00600] Definition of EcucBooleanParamDef DcmRespondAllRequest

Parameter Name	DcmRespondAllRequest		
Parent Container	DcmGeneral		
Description	If set to FALSE the Dcm will not respond to diagnostic request that contains a service ID which is in the range from 0x40 to 0x7F or in the range from 0xC0 to 0xFF (Response IDs).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01240] Definition of EcucFloatParamDef DcmS3ServerTimeoutOverwrite

Parameter Name	DcmS3ServerTimeoutOverwrite		
Parent Container	DcmGeneral		
Description	Time for the server to keep a diagnostic session other than the defaultSession active while not receiving any diagnostic request message. This parameter overwrites the 5s recommend value of the S3server timeout in ISO 14229-2. The parameter value is defined in seconds.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[5 .. INF[		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants



△

	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00820] Definition of EcucFloatParamDef DcmTaskTime [

Parameter Name	DcmTaskTime		
Parent Container	DcmGeneral		
Description	Allow to configure the time for the periodic cyclic task. Please note: This configuration value shall be equal to the value in the RTE module. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as float value in seconds. Dcm configuration tools must convert this float value to the appropriate value format for the use in the software implementation of Dcm. min: A negative value and zero is not allowed. upperMultiplicity: Exactly one TaskTime must be specified per configuration. lowerMultiplicity: Exactly one TaskTime must be specified per configuration.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	]0 .. INF[		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	X	VARIANT-LINK-TIME
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00821] Definition of EcucBooleanParamDef DcmVersionInfoApi [

Parameter Name	DcmVersionInfoApi		
Parent Container	DcmGeneral		
Description	Preprocessor switch to enable or disable the output Version info of the functionality.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_00984] Definition of EcucReferenceDef DcmVinRef [

Parameter Name	DcmVinRef		
Parent Container	DcmGeneral		
Description	Reference to the Did containing the VIN Information. This parameter is needed for function Dcm_GetVin		
Multiplicity	0..1		
Type	Reference to DcmDspDid		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

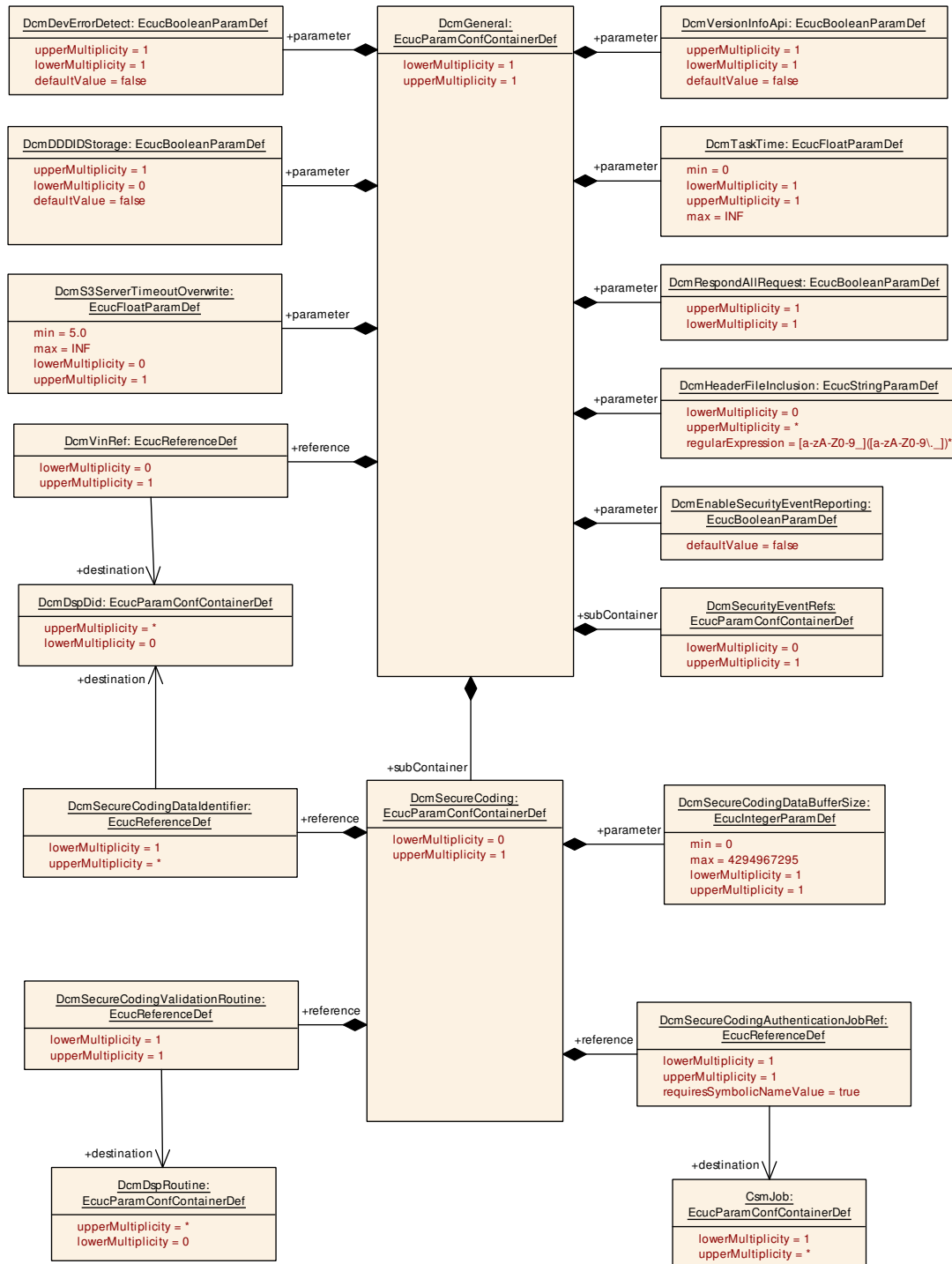


Figure 10.58: DcmGeneral configuration overview

10.2.6.1 DcmSecurityEventRefs

[ECUC_Dcm_01188] Definition of EcucParamConfContainerDef DcmSecurityEventRefs

Status: DRAFT

[

Container Name	DcmSecurityEventRefs		
Parent Container	DcmGeneral		
Description	Container for the references to IdsMEvent elements representing the security events that the Dcm module shall report to the IdsM in case the corresponding security related event occurs (and if DcmEnableSecurityEventReporting is set to "true"). The standardized security events in this container can be extended by vendor-specific security events. Tags: atp.Status=draft		
Multiplicity	0..1		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
SEV_UDS_AUTHENTICATION_FAILED	0..1	[ECUC_Dcm_01255]
SEV_UDS_AUTHENTICATION_NEEDED	0..1	[ECUC_Dcm_01251]
SEV_UDS_AUTHENTICATION_SUCCESSFUL	0..1	[ECUC_Dcm_01254]
SEV_UDS_CLEAR_DTC_FAILED	0..1	[ECUC_Dcm_01267]
SEV_UDS_CLEAR_DTC_SUCCESSFUL	0..1	[ECUC_Dcm_01266]
SEV_UDS_COMMUNICATION_CONTROL_FAILED	0..1	[ECUC_Dcm_01265]
SEV_UDS_COMMUNICATION_CONTROL_SUCCESSFUL	0..1	[ECUC_Dcm_01264]
SEV_UDS_CONTROL_DTC_SETTING_FAILED	0..1	[ECUC_Dcm_01269]
SEV_UDS_CONTROL_DTC_SETTING_SUCCESSFUL	0..1	[ECUC_Dcm_01268]
SEV_UDS_ECU_RESET_FAILED	0..1	[ECUC_Dcm_01271]
SEV_UDS_ECU_RESET_SUCCESSFUL	0..1	[ECUC_Dcm_01270]
SEV_UDS_IO_CONTROL_FAILED	0..1	[ECUC_Dcm_01275]
SEV_UDS_IO_CONTROL_SUCCESSFUL	0..1	[ECUC_Dcm_01274]
SEV_UDS_REQUEST_FILE_TRANSFER_FAILED	0..1	[ECUC_Dcm_01263]
SEV_UDS_REQUEST_FILE_TRANSFER_SUCCESSFUL	0..1	[ECUC_Dcm_01262]
SEV_UDS_REQUEST_UP_DOWNLOAD_FAILED	0..1	[ECUC_Dcm_01261]
SEV_UDS_REQUEST_UP_DOWNLOAD_SUCCESSFUL	0..1	[ECUC_Dcm_01260]
SEV_UDS_ROUTINE_CONTROL_FAILED	0..1	[ECUC_Dcm_01273]
SEV_UDS_ROUTINE_CONTROL_SUCCESSFUL	0..1	[ECUC_Dcm_01272]
SEV_UDS_SECURITY_ACCESS_FAILED	0..1	[ECUC_Dcm_01253]
SEV_UDS_SECURITY_ACCESS_NEEDED	0..1	[ECUC_Dcm_01250]
SEV_UDS_SECURITY_ACCESS_SUCCESSFUL	0..1	[ECUC_Dcm_01252]
SEV_UDS_WRITE_DATA_FAILED	0..1	[ECUC_Dcm_01257]





Included Parameters		
Parameter Name	Multiplicity	ECUC ID
SEV_UDS_WRITE_DATA_SUCCESSFUL	0..1	[ECUC_Dcm_01256]
SEV_UDS_WRITE_MEMORY_FAILED	0..1	[ECUC_Dcm_01259]
SEV_UDS_WRITE_MEMORY_SUCCESSFUL	0..1	[ECUC_Dcm_01258]

No Included Containers

]

[ECUC_Dcm_01255] Definition of EcucReferenceDef SEV_UDS_AUTHENTICATION_FAILED [

Parameter Name	SEV_UDS_AUTHENTICATION_FAILED		
Parent Container	DcmSecurityEventRefs		
Description	Authentication (via Authentication SID 0x29) failed		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01251] Definition of EcucReferenceDef SEV_UDS_AUTHENTICATION_NEEDED [

Parameter Name	SEV_UDS_AUTHENTICATION_NEEDED		
Parent Container	DcmSecurityEventRefs		
Description	A diagnostic request was received while the required authentication to execute this service is not given. NRC 0x34 (authenticationRequired) was returned.		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	





Dependency	

]

[ECUC_Dcm_01254] Definition of EcucReferenceDef SEV_UDS_AUTHENTICATION_SUCCESSFUL [

Parameter Name	SEV_UDS_AUTHENTICATION_SUCCESSFUL		
Parent Container	DcmSecurityEventRefs		
Description	Succesfully authenticated (via Authentication SID 0x29)		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01267] Definition of EcucReferenceDef SEV_UDS_CLEAR_DTC_FAILED [

Parameter Name	SEV_UDS_CLEAR_DTC_FAILED		
Parent Container	DcmSecurityEventRefs		
Description	Clearing DTC information has been requested by SID 0x14 ClearDiagnostic Information, but failed		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01266] Definition of EcucReferenceDef SEV_UDS_CLEAR_DTC_SUCCESSFUL [

Parameter Name	SEV_UDS_CLEAR_DTC_SUCCESSFUL		
Parent Container	DcmSecurityEventRefs		
Description	DTC information has been cleared by SID 0x14 ClearDiagnosticInformation		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01265] Definition of EcucReferenceDef SEV_UDS_COMMUNICATION_CONTROL_FAILED [

Parameter Name	SEV_UDS_COMMUNICATION_CONTROL_FAILED		
Parent Container	DcmSecurityEventRefs		
Description	The control of a communication has been requested by service SID 0x28 CommunicationControl, but failed		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01264] Definition of EcucReferenceDef SEV_UDS_COMMUNICATION_CONTROL_SUCCESSFUL [

Parameter Name	SEV_UDS_COMMUNICATION_CONTROL_SUCCESSFUL		
Parent Container	DcmSecurityEventRefs		
Description	The control of a communication has been requested by service SID 0x28 CommunicationControl successfully		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		





Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01269] Definition of EcucReferenceDef SEV_UDS_CONTROL_DTC_SETTING_FAILED [

Parameter Name	SEV_UDS_CONTROL_DTC_SETTING_FAILED		
Parent Container	DcmSecurityEventRefs		
Description	Control of DTC setting has been requested by service SID 0x85 ControlDTCSetting, but failed		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01268] Definition of EcucReferenceDef SEV_UDS_CONTROL_DTC_SETTING_SUCCESSFUL [

Parameter Name	SEV_UDS_CONTROL_DTC_SETTING_SUCCESSFUL		
Parent Container	DcmSecurityEventRefs		
Description	The control of a DTC setting has been requested by service SID 0x85 ControlDTCSetting successfully		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	





	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01271] Definition of EcucReferenceDef SEV_UDS_ECU_RESET_FAILED [

Parameter Name	SEV_UDS_ECU_RESET_FAILED		
Parent Container	DcmSecurityEventRefs		
Description	ECU Reset has been requested by SID 0x11 ECUReset, but failed		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01270] Definition of EcucReferenceDef SEV_UDS_ECU_RESET_SUCCESSFUL [

Parameter Name	SEV_UDS_ECU_RESET_SUCCESSFUL		
Parent Container	DcmSecurityEventRefs		
Description	ECU has been reset by SID 0x11 ECUReset		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01275] Definition of EcucReferenceDef SEV_UDS_IO_CONTROL_FAILED

Parameter Name	SEV_UDS_IO_CONTROL_FAILED		
Parent Container	DcmSecurityEventRefs		
Description	IOControl operation has been requested by service SID 0x2F InputOutputControlBy Identifier, but failed		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_01274] Definition of EcucReferenceDef SEV_UDS_IO_CONTROL_SUCCESSFUL

Parameter Name	SEV_UDS_IO_CONTROL_SUCCESSFUL		
Parent Container	DcmSecurityEventRefs		
Description	IOControl operation has been requested by service SID 0x2F InputOutputControlBy Identifier successfully		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_01263] Definition of EcucReferenceDef SEV_UDS_REQUEST_FILE_TRANSFER_FAILED

Parameter Name	SEV_UDS_REQUEST_FILE_TRANSFER_FAILED		
Parent Container	DcmSecurityEventRefs		
Description	A file transfer sequence has been requested with SID 0x38, but failed		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		





Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01262] Definition of EcucReferenceDef SEV_UDS_REQUEST_FILE_TRANSFER_SUCCESSFUL [

Parameter Name	SEV_UDS_REQUEST_FILE_TRANSFER_SUCCESSFUL		
Parent Container	DcmSecurityEventRefs		
Description	A file transfer sequence has been requested successfully with SID 0x38		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01261] Definition of EcucReferenceDef SEV_UDS_REQUEST_UP_DOWNLOAD_FAILED [

Parameter Name	SEV_UDS_REQUEST_UP_DOWNLOAD_FAILED		
Parent Container	DcmSecurityEventRefs		
Description	An upload / download sequence has been requested with SID 0x34 or SID 0x35, but failed		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	





	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01260] Definition of EcucReferenceDef SEV_UDS_REQUEST_UP_DOWNLOAD_SUCCESSFUL ┌

Parameter Name	SEV_UDS_REQUEST_UP_DOWNLOAD_SUCCESSFUL		
Parent Container	DcmSecurityEventRefs		
Description	An upload / download sequence has been requested successfully with SID 0x34 or SID 0x35		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01273] Definition of EcucReferenceDef SEV_UDS_ROUTINE_CONTROL_FAILED ┌

Parameter Name	SEV_UDS_ROUTINE_CONTROL_FAILED		
Parent Container	DcmSecurityEventRefs		
Description	The control of a routine has been requested by service SID 0x31 RoutineControl, but failed		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01272] Definition of EcucReferenceDef SEV_UDS_ROUTINE_CONTROL_SUCCESSFUL [

Parameter Name	SEV_UDS_ROUTINE_CONTROL_SUCCESSFUL		
Parent Container	DcmSecurityEventRefs		
Description	The control of a routine has been requested by service SID 0x31 RoutineControl successfully		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01253] Definition of EcucReferenceDef SEV_UDS_SECURITY_ACCESS_FAILED [

Parameter Name	SEV_UDS_SECURITY_ACCESS_FAILED		
Parent Container	DcmSecurityEventRefs		
Description	Unlocking of the ECU (via Security Access SID 0x27) failed		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01250] Definition of EcucReferenceDef SEV_UDS_SECURITY_ACCESS_NEEDED [

Parameter Name	SEV_UDS_SECURITY_ACCESS_NEEDED		
Parent Container	DcmSecurityEventRefs		
Description	Tester has sent a diagnostic request without meeting the server's security level requirements for that service. NRC 0x33 (securityAccessDenied) was returned.		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		





Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01252] Definition of EcucReferenceDef SEV_UDS_SECURITY_ACCESS_SUCCESSFUL [

Parameter Name	SEV_UDS_SECURITY_ACCESS_SUCCESSFUL		
Parent Container	DcmSecurityEventRefs		
Description	Successful unlocked the ECU (via Security Access SID 0x27)		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01257] Definition of EcucReferenceDef SEV_UDS_WRITE_DATA_FAILED [

Parameter Name	SEV_UDS_WRITE_DATA_FAILED		
Parent Container	DcmSecurityEventRefs		
Description	Change of Diagnostic data identifier has been requested by SID 0x2E WriteDataBy Identifier, but failed		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	





	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01256] Definition of EcucReferenceDef SEV_UDS_WRITE_DATA_SUCCESSFUL

Parameter Name	SEV_UDS_WRITE_DATA_SUCCESSFUL		
Parent Container	DcmSecurityEventRefs		
Description	Diagnostic data identifier has been written by SID 0x2E WriteDataByIdentifier		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01259] Definition of EcucReferenceDef SEV_UDS_WRITE_MEMORY_FAILED

Parameter Name	SEV_UDS_WRITE_MEMORY_FAILED		
Parent Container	DcmSecurityEventRefs		
Description	Writing of Data into memory has been requested by SID 0x3D WriteMemoryBy Address, but failed		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

└

[ECUC_Dcm_01258] Definition of EcucReferenceDef SEV_UDS_WRITE_MEMORY_SUCCESSFUL

Parameter Name	SEV_UDS_WRITE_MEMORY_SUCCESSFUL		
Parent Container	DcmSecurityEventRefs		
Description	Data has been written into memory by SID 0x3D WriteMemoryByAddress		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

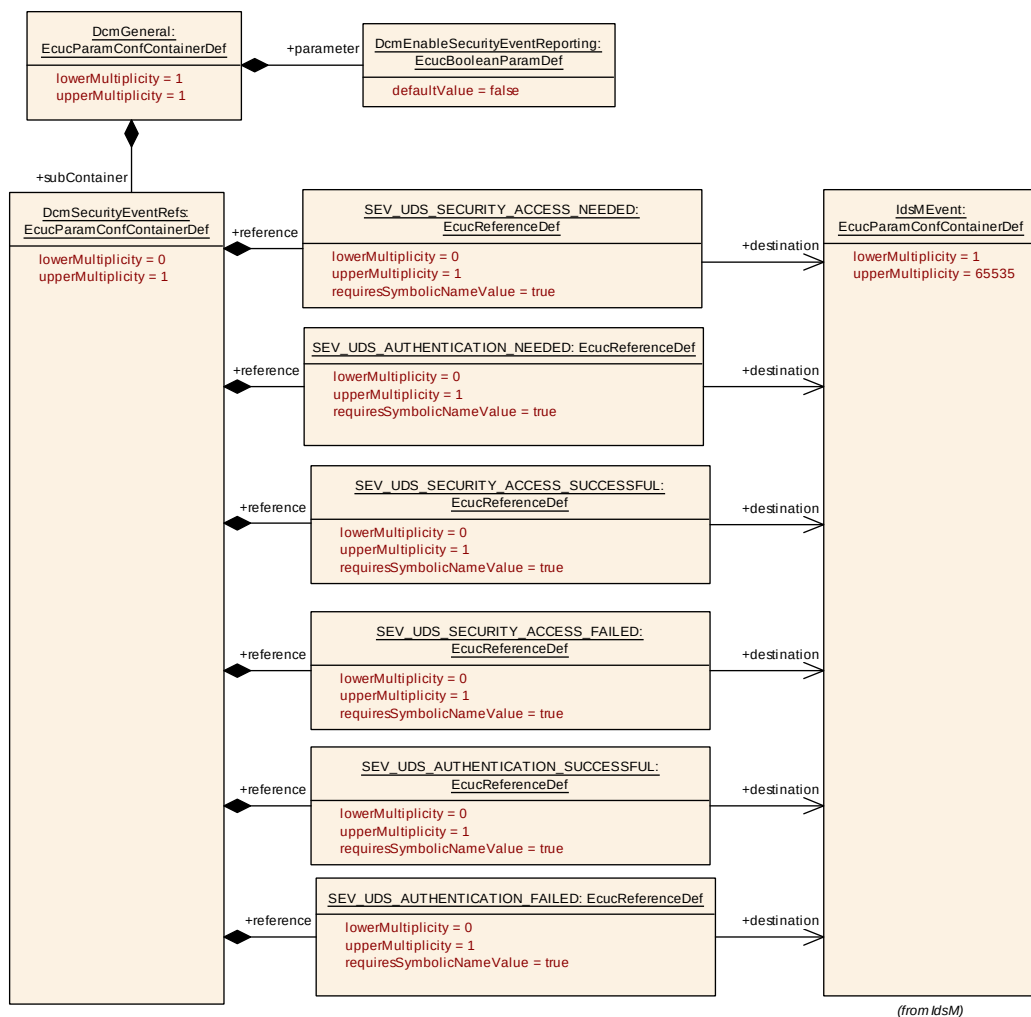


Figure 10.59: DcmSecurityEventRefs1 configuration overview

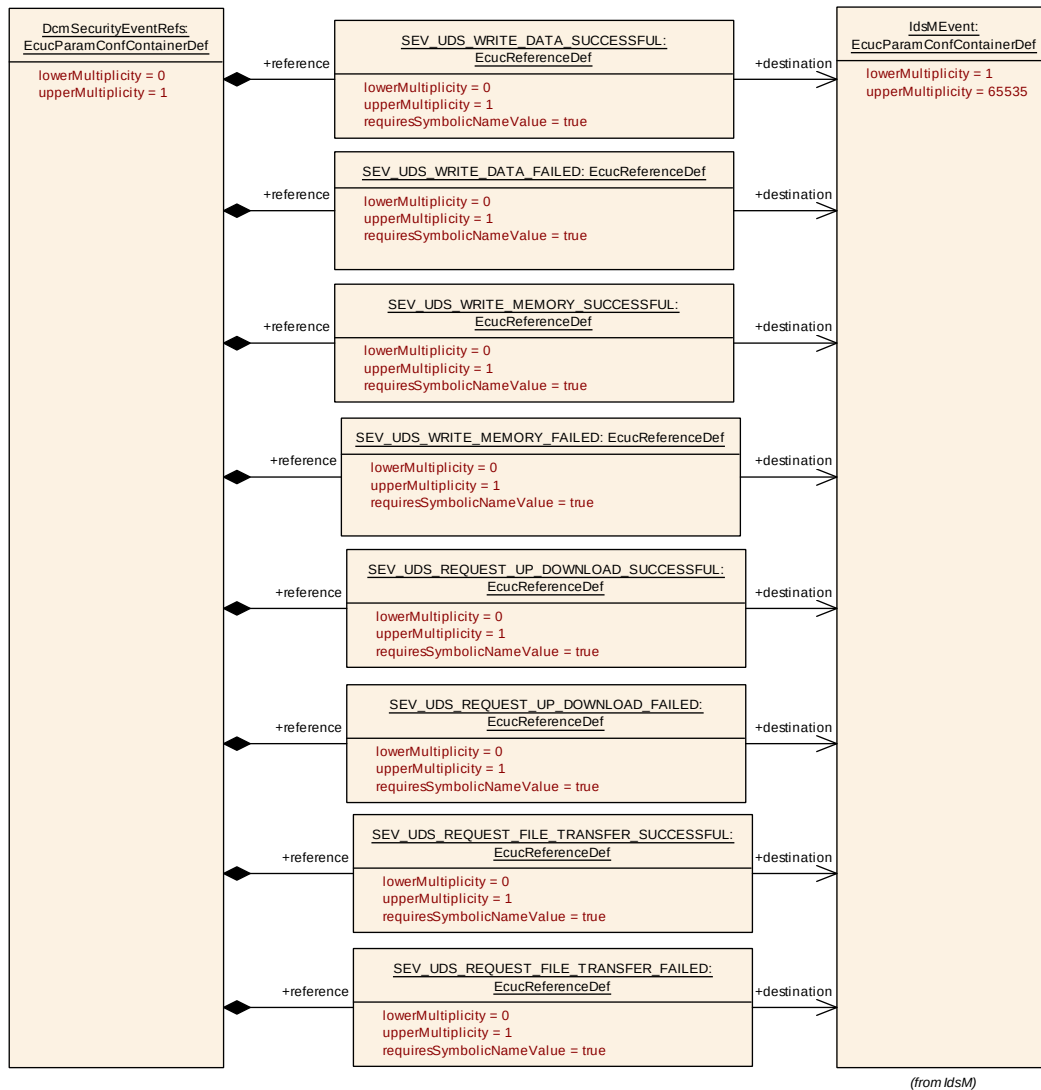


Figure 10.60: DcmSecurityEventRefs2 configuration overview

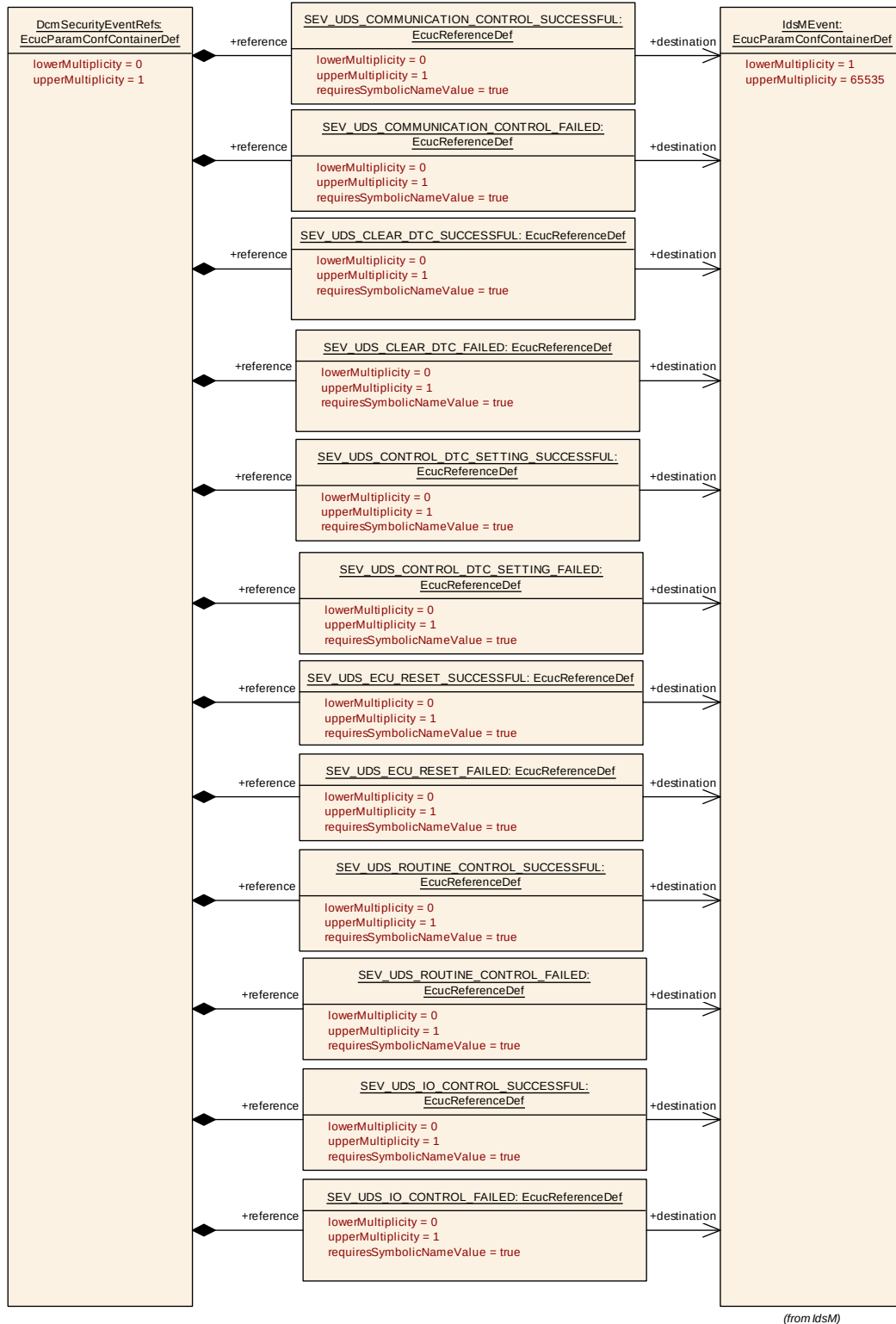


Figure 10.61: DcmSecurityEventRefs3 configuration overview

10.2.6.2 DcmSecureCoding

[ECUC_Dcm_01245] Definition of EcucParamConfContainerDef DcmSecureCoding

Container Name	DcmSecureCoding		
Parent Container	DcmGeneral		
Description	Contains references to diagnostic elements that are used for the secure coding process.		
Multiplicity	0..1		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
DcmSecureCodingDataBufferSize	1	[ECUC_Dcm_01247]
DcmSecureCodingAuthenticationJobRef	1	[ECUC_Dcm_01249]
DcmSecureCodingDataIdentifier	1..*	[ECUC_Dcm_01246]
DcmSecureCodingValidationRoutine	1	[ECUC_Dcm_01248]

No Included Containers

[ECUC_Dcm_01247] Definition of EcucIntegerParamDef DcmSecureCodingDataBufferSize

Parameter Name	DcmSecureCodingDataBufferSize		
Parent Container	DcmSecureCoding		
Description	Provides the size of the Dcm internal buffer to store data during the secure coding sequence.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

[ECUC_Dcm_01249] Definition of EcucReferenceDef DcmSecureCodingAuthenticationJobRef

Parameter Name	DcmSecureCodingAuthenticationJobRef		
Parent Container	DcmSecureCoding		
Description	Reference to a CsmJob used to verify the authentication of the secure coding data in the Csm.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01246] Definition of EcucReferenceDef DcmSecureCodingDataIdentifier

Parameter Name	DcmSecureCodingDataIdentifier		
Parent Container	DcmSecureCoding		
Description	Refers to a writeable data identifier that requires the secure coding process to write the data.		
Multiplicity	1..*		
Type	Reference to DcmDspDid		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Dependency			

]

[ECUC_Dcm_01248] Definition of EcucReferenceDef DcmSecureCodingValidationRoutine

Parameter Name	DcmSecureCodingValidationRoutine		
Parent Container	DcmSecureCoding		
Description	Refers to a routine control that triggers the secure coding validation and data update process.		
Multiplicity	1		
Type	Reference to DcmDspRoutine		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	





	Post-build time	–	
Dependency			

└

10.3 Protocol Configuration Example

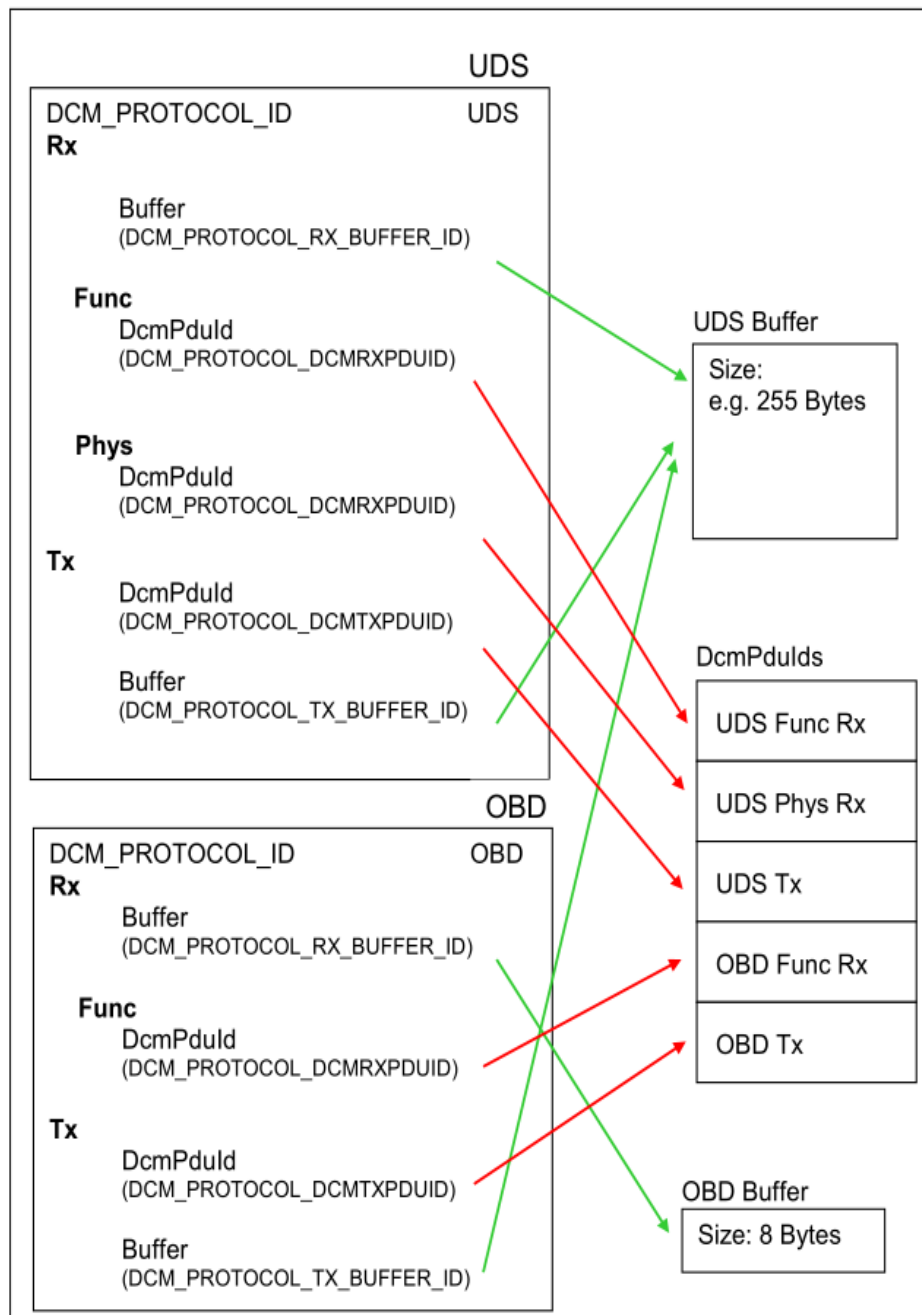


Figure 10.62: Examples of protocol configuration with focus on buffer / DcmPduld settings

Above example shows protocol configuration at the use cases examples [OBD](#) and [UDS](#) (used for customer enhanced diagnosis). It is assumed that for [UDS](#) communication, there are functional and physical requests. There will be separate `DcmPduRxIds` for functional and physical reception.

Concerning buffer configuration it is proposed to use a separate buffer for the functional requests. This in correspondence to support the keep alive logic with functional addressed `TesterPresent` commands.

It is also proposed to use a separate receive buffer for the [OBD](#) commands. This in reference to support the protocol switch functionality.

It is allowed to share for both protocols the transmit buffer. Please note: The [DcmDsl-ProtocolRx](#) has two possible configurations:

- functional
- physical

The physical shall have a 1:1 (or 1:0) dependency to the [DcmDslMain-Connection](#). (which means: [DcmDslProtocolRxPduRef](#) in combination `DCM_PROTOCOL_RX_ADDR_TYP = physical` can exist only once per "Module") The functional shall have a 1:n dependency to the [DcmDslMain-Connection](#). (which means: [DcmDslProtocolRxPduRef](#) in combination `DCM_PROTOCOL_RX_ADDR_TYP = functional` can exist several times per "Module") The [DcmDslProtocolTx](#) shall exist only once per "Module"

10.4 Published Information

For details refer to [8] Chapter 10.3 "*Published Information*".

A Change history of AUTOSAR traceable items

Please note that the lists in this chapter also include traceable items that have been removed from the specification in a later version. These items do not appear as hyperlinks in the document.

A.1 Traceable item history of this document according to AUTOSAR Release R25-11

A.1.1 Added Specification Items in R25-11

Number	Heading
[ECUC_Dcm_01276]	Definition of EcucReferenceDef DcmDspAuthenticationMileageInformation
[ECUC_Dcm_01277]	Definition of EcucIntegerParamDef DcmDspAuthenticationMileageOffset Limit
[SWS_Dcm_00484]	Maximum number of DIDs in a OBD DID request
[SWS_Dcm_01135]	Routine info byte support
[SWS_Dcm_01566]	Authentication distance supervision
[SWS_Dcm_91126]	Definition of configurable interface Xxx_GetInfotypeValueData for two byte data length

Table A.1: Added Specification Items in R25-11

A.1.2 Changed Specification Items in R25-11

Number	Heading
[ECUC_Dcm_00638]	Definition of EcucIntegerParamDef DcmDspMaxDidToRead
[ECUC_Dcm_00826]	Definition of EcucIntegerParamDef DcmDslProtocolRxConnectionId
[ECUC_Dcm_00881]	Definition of EcucEnumerationParamDef DcmDspRoutineSignalType
[ECUC_Dcm_00882]	Definition of EcucEnumerationParamDef DcmDspRoutineSignalType
[ECUC_Dcm_00883]	Definition of EcucEnumerationParamDef DcmDspRoutineSignalType
[ECUC_Dcm_00884]	Definition of EcucEnumerationParamDef DcmDspRoutineSignalType
[ECUC_Dcm_00885]	Definition of EcucEnumerationParamDef DcmDspRoutineSignalType
[ECUC_Dcm_00890]	Definition of EcucIntegerParamDef DcmDspVehInfoDataSize
[ECUC_Dcm_00935]	Definition of EcucParamConfContainerDef DcmDspControlDTCSetting
[ECUC_Dcm_00985]	Definition of EcucEnumerationParamDef DcmDspDataType
[ECUC_Dcm_01018]	Definition of EcucEnumerationParamDef DcmDspPidDataType
[ECUC_Dcm_01063]	Definition of EcucIntegerParamDef DcmDspRoutineInfoByte
[ECUC_Dcm_01078]	Definition of EcucIntegerParamDef DcmDspRequestControlInfoByte





Number	Heading
[ECUC_Dcm_01083]	Definition of EcucReferenceDef DcmDemClientRef
[ECUC_Dcm_01088]	Definition of EcucReferenceDef DcmDspReadMemoryRangeSessionLevelRef
[ECUC_Dcm_01090]	Definition of EcucFunctionNameDef DcmDspRequestRoutineResultsConfirmationFnc
[ECUC_Dcm_01091]	Definition of EcucBooleanParamDef DcmDspRequestRoutineResultsConfirmationEnabled
[ECUC_Dcm_01120]	Definition of EcucEnumerationParamDef DcmDspRoutineSignalType
[ECUC_Dcm_01151]	Definition of EcucParamConfContainerDef DcmDspAuthentication
[ECUC_Dcm_01162]	Definition of EcucParamConfContainerDef DcmDspAuthenticationConnection
[SWS_Dcm_00093]	Definition of API function Dcm_TpRxIndication
[SWS_Dcm_00253]	Support of UDS Service ReadDataByIdentifier (0x22)
[SWS_Dcm_00333]	Definition of imported datatypes of module Dcm
[SWS_Dcm_00351]	Definition of API function Dcm_TpTxConfirmation
[SWS_Dcm_00688]	Definition of ClientServerInterface InfotypeServices_{VehInfoData}
[SWS_Dcm_00948]	
[SWS_Dcm_00980]	Definition of ImplementationDataType Dcm_NegativeResponseCodeType
[SWS_Dcm_01018]	Definition of ImplementationDataType Dcm_RequestDataOut_{Routine}_{Signal}PrimitivType
[SWS_Dcm_01020]	Definition of ImplementationDataType Dcm_StartDataIn_{Routine}_{Signal}PrimitivType
[SWS_Dcm_01021]	Definition of ImplementationDataType Dcm_StartDataOut_{Routine}_{Signal}PrimitivType
[SWS_Dcm_01024]	Definition of ImplementationDataType Dcm_StopDataIn_{Routine}_{Signal}PrimitivType
[SWS_Dcm_01025]	Definition of ImplementationDataType Dcm_StopDataOut_{Routine}_{Signal}PrimitivType
[SWS_Dcm_01083]	
[SWS_Dcm_01088]	
[SWS_Dcm_01090]	
[SWS_Dcm_01091]	
[SWS_Dcm_01335]	Maximum number of DIDs in non-OBd DID request
[SWS_Dcm_01392]	
[SWS_Dcm_01393]	
[SWS_Dcm_01446]	
[SWS_Dcm_01447]	
[SWS_Dcm_01448]	
[SWS_Dcm_01449]	
[SWS_Dcm_01450]	
[SWS_Dcm_01451]	





Number	Heading
[SWS_Dcm_01452]	
[SWS_Dcm_01453]	
[SWS_Dcm_01454]	
[SWS_Dcm_01455]	
[SWS_Dcm_01456]	
[SWS_Dcm_01457]	
[SWS_Dcm_01458]	
[SWS_Dcm_01462]	Required configuration for bidirectional authentication
[SWS_Dcm_91002]	Definition of optional interfaces requested by module Dcm
[SWS_Dcm_91040]	Definition of ImplementationDataType Dcm_RequestData Out_{Routine}_{Signal}Type
[SWS_Dcm_91041]	Definition of ImplementationDataType Dcm_RequestData Out_{Routine}_{Signal}ArrayType
[SWS_Dcm_91042]	Definition of ImplementationDataType Dcm_StartData In_{Routine}_{Signal}Type
[SWS_Dcm_91043]	Definition of ImplementationDataType Dcm_StartData In_{Routine}_{Signal}ArrayType
[SWS_Dcm_91044]	Definition of ImplementationDataType Dcm_StartData Out_{Routine}_{Signal}Type
[SWS_Dcm_91045]	Definition of ImplementationDataType Dcm_StartData Out_{Routine}_{Signal}ArrayType
[SWS_Dcm_91046]	Definition of ImplementationDataType Dcm_StopData In_{Routine}_{Signal}Type
[SWS_Dcm_91047]	Definition of ImplementationDataType Dcm_StopData In_{Routine}_{Signal}ArrayType
[SWS_Dcm_91048]	Definition of ImplementationDataType Dcm_StopData Out_{Routine}_{Signal}Type
[SWS_Dcm_91049]	Definition of ImplementationDataType Dcm_StopData Out_{Routine}_{Signal}ArrayType
[SWS_Dcm_91051]	Definition of ImplementationDataType Dcm_DataElement_{Data}Type
[SWS_Dcm_91052]	Definition of ImplementationDataType Dcm_RequestData In_{Routine}_{Signal}Type
[SWS_Dcm_91054]	Definition of ImplementationDataType Dcm_RequestData In_{Routine}_{Signal}PrimitiveType
[SWS_Dcm_91055]	Definition of ImplementationDataType Dcm_RequestData In_{Routine}_{Signal}ArrayType
[SWS_Dcm_91062]	Definition of ImplementationDataType Dcm_DataElement_{Data}_Primitive Type
[SWS_Dcm_91063]	Definition of ImplementationDataType Dcm_DataElement_{Data}_ArrayType
[SWS_Dcm_91064]	Definition of ImplementationDataType Dcm_ResponseDataArrayType
[SWS_Dcm_91065]	Definition of ClientServerInterface UploadDownloadServices
[SWS_Dcm_91066]	Definition of ImplementationDataType Dcm_FileAndDirNameType





Number	Heading
[SWS_Dcm_91078]	Definition of configurable interface Dcm_ProcessRequestAddFile
[SWS_Dcm_91079]	Definition of configurable interface Dcm_ProcessRequestDeleteFile
[SWS_Dcm_91080]	Definition of configurable interface Dcm_ProcessRequestReplaceFile
[SWS_Dcm_91081]	Definition of configurable interface Dcm_ProcessRequestReadFile
[SWS_Dcm_91082]	Definition of configurable interface Dcm_ProcessRequestReadDir
[SWS_Dcm_91083]	Definition of configurable interface Dcm_WriteFile
[SWS_Dcm_91084]	Definition of Port UploadDownloadServices required by module Dcm
[SWS_Dcm_91085]	Definition of configurable interface Dcm_ReadFileOrDir

Table A.2: Changed Specification Items in R25-11

A.1.3 Deleted Specification Items in R25-11

Number	Heading
[ECUC_Dcm_00700]	Definition of EcucEnumerationParamDef DcmDslProtocolTransType
[ECUC_Dcm_00965]	Definition of EcucBooleanParamDef DcmSupportDTCSettingControlOption Record
[ECUC_Dcm_01166]	Definition of EcucReferenceDef DcmDspAuthenticationPublicKeyElement Ref
[ECUC_Dcm_01176]	Definition of EcucReferenceDef DcmDspAuthenticationCertificatePublicKey StoreJobRef
[ECUC_Dcm_01178]	Definition of EcucReferenceDef DcmDspAuthenticationECUCertificateKey ElementRef
[SWS_Dcm_00162]	
[SWS_Dcm_00701]	
[SWS_Dcm_00829]	
[SWS_Dcm_00852]	
[SWS_Dcm_00976]	Definition of datatype Dcm_StatusType
[SWS_Dcm_01564]	

Table A.3: Deleted Specification Items in R25-11

A.1.4 Added Constraints in R25-11

Number	Heading
[SWS_Dcm_CONSTR_06037]	Restrictions on size parameter for 64 Bit arrays





Number	Heading
[SWS_Dcm_CONSTR_06042]	Restrictions on size parameter for 64 Bit arrays
[SWS_Dcm_CONSTR_06114]	Complete distance offset configuration
[SWS_Dcm_CONSTR_06115]	Limiting data configuration for distance information

Table A.4: Added Constraints in R25-11

A.1.5 Changed Constraints in R25-11

Number	Heading
[SWS_Dcm_CONSTR_06002]	Existence of size parameter
[SWS_Dcm_CONSTR_06008]	Define the usage of <code>DcmDspRoutineParameterSize</code> parameter
[SWS_Dcm_CONSTR_06012]	Existence of size parameter

Table A.5: Changed Constraints in R25-11

A.1.6 Deleted Constraints in R25-11

none

A.2 Traceable item history of this document according to AUTOSAR Release R24-11

A.2.1 Added Specification Items in R24-11

Number	Heading
[ECUC_Dcm_01250]	Definition of EcucReferenceDef SEV_UDS_SECURITY_ACCESS_NEEDED
[ECUC_Dcm_01251]	Definition of EcucReferenceDef SEV_UDS_AUTHENTICATION_NEEDED
[ECUC_Dcm_01252]	Definition of EcucReferenceDef SEV_UDS_SECURITY_ACCESS_SUCCESSFUL
[ECUC_Dcm_01253]	Definition of EcucReferenceDef SEV_UDS_SECURITY_ACCESS_FAILED
[ECUC_Dcm_01254]	Definition of EcucReferenceDef SEV_UDS_AUTHENTICATION_SUCCESSFUL
[ECUC_Dcm_01255]	Definition of EcucReferenceDef SEV_UDS_AUTHENTICATION_FAILED
[ECUC_Dcm_01256]	Definition of EcucReferenceDef SEV_UDS_WRITE_DATA_SUCCESSFUL
[ECUC_Dcm_01257]	Definition of EcucReferenceDef SEV_UDS_WRITE_DATA_FAILED
[ECUC_Dcm_01258]	Definition of EcucReferenceDef SEV_UDS_WRITE_MEMORY_SUCCESSFUL
[ECUC_Dcm_01259]	Definition of EcucReferenceDef SEV_UDS_WRITE_MEMORY_FAILED
[ECUC_Dcm_01260]	Definition of EcucReferenceDef SEV_UDS_REQUEST_UP_DOWNLOAD_SUCCESSFUL
[ECUC_Dcm_01261]	Definition of EcucReferenceDef SEV_UDS_REQUEST_UP_DOWNLOAD_FAILED
[ECUC_Dcm_01262]	Definition of EcucReferenceDef SEV_UDS_REQUEST_FILE_TRANSFER_SUCCESSFUL
[ECUC_Dcm_01263]	Definition of EcucReferenceDef SEV_UDS_REQUEST_FILE_TRANSFER_FAILED
[ECUC_Dcm_01264]	Definition of EcucReferenceDef SEV_UDS_COMMUNICATION_CONTROL_SUCCESSFUL
[ECUC_Dcm_01265]	Definition of EcucReferenceDef SEV_UDS_COMMUNICATION_CONTROL_FAILED
[ECUC_Dcm_01266]	Definition of EcucReferenceDef SEV_UDS_CLEAR_DTC_SUCCESSFUL
[ECUC_Dcm_01267]	Definition of EcucReferenceDef SEV_UDS_CLEAR_DTC_FAILED
[ECUC_Dcm_01268]	Definition of EcucReferenceDef SEV_UDS_CONTROL_DTC_SETTING_SUCCESSFUL
[ECUC_Dcm_01269]	Definition of EcucReferenceDef SEV_UDS_CONTROL_DTC_SETTING_FAILED
[ECUC_Dcm_01270]	Definition of EcucReferenceDef SEV_UDS_ECU_RESET_SUCCESSFUL
[ECUC_Dcm_01271]	Definition of EcucReferenceDef SEV_UDS_ECU_RESET_FAILED
[ECUC_Dcm_01272]	Definition of EcucReferenceDef SEV_UDS_ROUTINE_CONTROL_SUCCESSFUL
[ECUC_Dcm_01273]	Definition of EcucReferenceDef SEV_UDS_ROUTINE_CONTROL_FAILED
[ECUC_Dcm_01274]	Definition of EcucReferenceDef SEV_UDS_IO_CONTROL_SUCCESSFUL





Number	Heading
[ECUC_Dcm_01275]	Definition of EcucReferenceDef SEV_UDS_IO_CONTROL_FAILED
[SWS_Dcm_01702]	Reporting security access denied to IdsM
[SWS_Dcm_01703]	Security event context data definition: SEV_UDS_SECURITY_ACCESS_NEEDED
[SWS_Dcm_01704]	Reporting authentication required to IdsM
[SWS_Dcm_01705]	Security event context data definition: SEV_UDS_AUTHENTICATION_NEEDED
[SWS_Dcm_01706]	Reporting successful security access to IdsM
[SWS_Dcm_01707]	Security event context data definition: SEV_UDS_SECURITY_ACCESS_SUCCESSFUL
[SWS_Dcm_01708]	Reporting failed security access to IdsM
[SWS_Dcm_01709]	Security event context data definition: SEV_UDS_SECURITY_ACCESS_FAILED
[SWS_Dcm_01710]	Reporting successful authentication to IdsM
[SWS_Dcm_01711]	Security event context data definition: SEV_UDS_AUTHENTICATION_SUCCESSFUL
[SWS_Dcm_01712]	Reporting failed authentication to IdsM
[SWS_Dcm_01713]	Security event context data definition: SEV_UDS_AUTHENTICATION_FAILED
[SWS_Dcm_01714]	Reporting successful writing of data to IdsM
[SWS_Dcm_01715]	Security event context data definition: SEV_UDS_WRITE_DATA_SUCCESSFUL
[SWS_Dcm_01716]	Reporting writing of data failed to IdsM
[SWS_Dcm_01717]	Security event context data definition: SEV_UDS_WRITE_DATA_FAILED
[SWS_Dcm_01718]	Reporting successful up download to IdsM
[SWS_Dcm_01719]	Security event context data definition: SEV_UDS_REQUEST_UP_DOWNLOAD_SUCCESSFUL
[SWS_Dcm_01720]	Reporting up download failed to IdsM
[SWS_Dcm_01721]	Security event context data definition: SEV_UDS_REQUEST_UP_DOWNLOAD_FAILED
[SWS_Dcm_01722]	Reporting successful file transfer to IdsM
[SWS_Dcm_01723]	Security event context data definition: SEV_UDS_REQUEST_FILE_TRANSFER_SUCCESSFUL
[SWS_Dcm_01724]	Reporting file transfer failed to IdsM
[SWS_Dcm_01725]	Security event context data definition: SEV_UDS_REQUEST_FILE_TRANSFER_FAILED
[SWS_Dcm_01726]	Reporting successful communication control to IdsM
[SWS_Dcm_01727]	Security event context data definition: SEV_UDS_COMMUNICATION_CONTROL_SUCCESSFUL
[SWS_Dcm_01728]	Reporting communication control failed to IdsM
[SWS_Dcm_01729]	Security event context data definition: SEV_UDS_COMMUNICATION_CONTROL_FAILED





Number	Heading
[SWS_Dcm_01730]	Reporting successful clearing of dtc to IdsM
[SWS_Dcm_01731]	Security event context data definition: SEV_UDS_CLEAR_DTC_SUCCESSFUL
[SWS_Dcm_01732]	Reporting clearing of dtc failed to IdsM
[SWS_Dcm_01733]	Security event context data definition: SEV_UDS_CLEAR_DTC_FAILED
[SWS_Dcm_01734]	Reporting successful control of dtc to IdsM
[SWS_Dcm_01735]	Security event context data definition: SEV_UDS_CONTROL_DTC_SETTING_SUCCESSFUL
[SWS_Dcm_01736]	Reporting control of dtc failed to IdsM
[SWS_Dcm_01737]	Security event context data definition: SEV_UDS_CONTROL_DTC_SETTING_FAILED
[SWS_Dcm_01738]	Reporting successful reset of ECU to IdsM
[SWS_Dcm_01739]	Security event context data definition: SEV_UDS_ECU_RESET_SUCCESSFUL
[SWS_Dcm_01740]	Reporting reset of ECU failed to IdsM
[SWS_Dcm_01741]	Security event context data definition: SEV_UDS_ECU_RESET_FAILED
[SWS_Dcm_01742]	Reporting successful routine control to IdsM
[SWS_Dcm_01743]	Security event context data definition: SEV_UDS_ROUTINE_CONTROL_SUCCESSFUL
[SWS_Dcm_01744]	Reporting routine control failed to IdsM
[SWS_Dcm_01745]	Security event context data definition: SEV_UDS_ROUTINE_CONTROL_FAILED
[SWS_Dcm_01746]	Reporting successful IO control to IdsM
[SWS_Dcm_01747]	Security event context data definition: SEV_UDS_IO_CONTROL_SUCCESSFUL
[SWS_Dcm_01748]	Reporting IO control failed to IdsM
[SWS_Dcm_01749]	Security event context data definition: SEV_UDS_IO_CONTROL_FAILED
[SWS_Dcm_01750]	Reporting successful Memory Write operation to IdsM
[SWS_Dcm_01751]	Security event context data definition: SEV_UDS_WRITE_MEMORY_SUCCESSFUL
[SWS_Dcm_01752]	Reporting Write Memory failed to IdsM
[SWS_Dcm_01753]	Security event context data definition: SEV_UDS_WRITE_MEMORY_FAILED

Table A.6: Added Specification Items in R24-11

A.2.2 Changed Specification Items in R24-11

Number	Heading
[ECUC_Dcm_00690]	Definition of EcucParamConfContainerDef DcmDsl
[ECUC_Dcm_00784]	Definition of EcucParamConfContainerDef DcmDspMemory
[ECUC_Dcm_00858]	Definition of EcucParamConfContainerDef DcmDspRoe
[ECUC_Dcm_00911]	Definition of EcucParamConfContainerDef DcmDspMemoryIdInfo
[ECUC_Dcm_01162]	Definition of EcucParamConfContainerDef DcmDspAuthentication Connection
[ECUC_Dcm_01166]	Definition of EcucReferenceDef DcmDspAuthenticationPublicKeyElement Ref
[ECUC_Dcm_01168]	Definition of EcucReferenceDef DcmDspAuthenticationWhiteListServices ElementRef
[ECUC_Dcm_01176]	Definition of EcucReferenceDef DcmDspAuthenticationCertificatePublicKey StoreJobRef
[ECUC_Dcm_01177]	Definition of EcucReferenceDef DcmDspAuthenticationECUCertificateRef
[ECUC_Dcm_01188]	Definition of EcucParamConfContainerDef DcmSecurityEventRefs
[ECUC_Dcm_01237]	Definition of EcucIntegerParamDef DcmDspRoeMaxNumComparisonOf ValueEvents
[ECUC_Dcm_01238]	Definition of EcucIntegerParamDef DcmDspRoeMaxSupportedDIDLength
[ECUC_Dcm_01239]	Definition of EcucIntegerParamDef DcmDspRoeMaxNumOfStored DTCStatusChangedEvents
[SWS_Dcm_00250]	
[SWS_Dcm_00481]	
[SWS_Dcm_00482]	
[SWS_Dcm_00483]	
[SWS_Dcm_00539]	Definition of configurable interface Dcm_ReadMemory
[SWS_Dcm_00594]	
[SWS_Dcm_00755]	Definition of configurable interface Dcm_ProcessRequestTransferExit
[SWS_Dcm_00763]	Definition of configurable interface <Module>_<DiagnosticService>
[SWS_Dcm_00796]	Definition of configurable interface Xxx_ReadDataLength
[SWS_Dcm_00804]	Definition of configurable interface Xxx_ReadDidData
[SWS_Dcm_00988]	Definition of datatype Dcm_ProgConditionsType
[SWS_Dcm_01151]	Definition of configurable interface Xxx_GetSeed
[SWS_Dcm_01182]	
[SWS_Dcm_01183]	
[SWS_Dcm_01204]	Definition of configurable interface Xxx_Stop
[SWS_Dcm_01429]	
[SWS_Dcm_01473]	Check certificate content
[SWS_Dcm_01478]	Mode declaration group per connection
[SWS_Dcm_01489]	No persistency for deauthenticated roles by SWC





Number	Heading
[SWS_Dcm_01508]	Sending positive response on verifyCertificateBidirectional
[SWS_Dcm_01517]	White list access
[SWS_Dcm_01518]	White list size check
[SWS_Dcm_01528]	Transition into authenticated state
[SWS_Dcm_01540]	Diagnostic service execution rights verification
[SWS_Dcm_01545]	Read user defined memory by status mask authentication check
[SWS_Dcm_01546]	Read user defined memory snapshot record authentication check
[SWS_Dcm_01547]	Read user defined memory extended data record authentication check
[SWS_Dcm_01590]	Security events for Dcm
[SWS_Dcm_01604]	Delay parallel processing on the same application interface change
[SWS_Dcm_01630]	Maximum number of comparison of value DIDs per scheduler
[SWS_Dcm_01632]	Maximum number of changed DTCs per scheduler
[SWS_Dcm_01641]	Support of UDS Services
[SWS_Dcm_01656]	Subfunction 0x42 response values
[SWS_Dcm_91002]	Definition of optional interfaces requested by module Dcm
[SWS_Dcm_91003]	Definition of configurable interface Xxx_GetSeed
[SWS_Dcm_91013]	Definition of configurable interface Xxx_RequestResults
[SWS_Dcm_91069]	Definition of API function Dcm_SetDeauthenticatedRole
[SWS_Dcm_91071]	Definition of configurable interface Dcm_ProcessTransferDataWrite
[SWS_Dcm_91073]	Definition of Port Authentication_{Connection} provided by module Dcm
[SWS_Dcm_91091]	Definition of configurable interface Xxx_ReadData
[SWS_Dcm_91092]	Definition of configurable interface Xxx_WriteData
[SWS_Dcm_91093]	Definition of configurable interface Xxx_GetScalingInformation
[SWS_Dcm_91094]	Definition of configurable interface Xxx_GetScalingInformation
[SWS_Dcm_91095]	Definition of configurable interface Xxx_WriteData
[SWS_Dcm_91097]	Definition of configurable interface Xxx_ReadDataLength
[SWS_Dcm_91098]	Definition of configurable interface Xxx_ReturnControlToECU
[SWS_Dcm_91099]	Definition of configurable interface Xxx_ResetToDefault
[SWS_Dcm_91100]	Definition of configurable interface Xxx_ResetToDefault
[SWS_Dcm_91102]	Definition of configurable interface Xxx_FreezeCurrentState
[SWS_Dcm_91103]	Definition of configurable interface Xxx_ShortTermAdjustment

Table A.7: Changed Specification Items in R24-11

A.2.3 Deleted Specification Items in R24-11

Number	Heading
[ECUC_Dcm_00679]	Definition of EcucParamConfContainerDef DcmDslCallbackDCMRequestService
[ECUC_Dcm_01189]	Definition of EcucReferenceDef SEV_UDS_WRITE_INVALID_DATA
[ECUC_Dcm_01190]	Definition of EcucReferenceDef SEV_UDS_SECURITY_ACCESS_DENIED
[ECUC_Dcm_01191]	Definition of EcucReferenceDef SEV_UDS_COMMUNICATION_CONTROL_SWITCHED_OFF
[ECUC_Dcm_01192]	Definition of EcucReferenceDef SEV_UDS_SERVICE_NOT_SUPPORTED
[ECUC_Dcm_01193]	Definition of EcucReferenceDef SEV_UDS_SUBFUNCTION_NOT_SUPPORTED
[ECUC_Dcm_01194]	Definition of EcucReferenceDef SEV_UDS_INCORRECT_MESSAGE_LENGTH_OR_FORMAT
[ECUC_Dcm_01195]	Definition of EcucReferenceDef SEV_UDS_REQUEST_SEQUENCE_ERROR
[ECUC_Dcm_01196]	Definition of EcucReferenceDef SEV_UDS_ECU_UNLOCK_SUCCESSFUL
[ECUC_Dcm_01197]	Definition of EcucReferenceDef SEV_UDS_AUTHENTICATION_SUCCESSFUL
[ECUC_Dcm_01198]	Definition of EcucReferenceDef SEV_UDS_CLEAR_DTC_SUCCESSFUL
[ECUC_Dcm_01199]	Definition of EcucReferenceDef SEV_UDS_ECU_RESET
[ECUC_Dcm_01200]	Definition of EcucReferenceDef SEV_UDS_WRITE_DATA
[ECUC_Dcm_01201]	Definition of EcucReferenceDef SEV_UDS_REQUEST_DOWNLOAD
[ECUC_Dcm_01202]	Definition of EcucReferenceDef SEV_UDS_DTC_SETTING_SWITCHED_OFF
[ECUC_Dcm_01203]	Definition of EcucReferenceDef SEV_UDS_REQUEST_OUT_OF_RANGE
[ECUC_Dcm_01204]	Definition of EcucReferenceDef SEV_UDS_REQUESTED_ACTIONS_REQUIRES_AUTHENTICATION
[ECUC_Dcm_01205]	Definition of EcucReferenceDef SEV_UDS_SECURITY_ACCESS_NUMBER_OF_ATTEMPTS_EXCEEDED
[ECUC_Dcm_01206]	Definition of EcucReferenceDef SEV_UDS_SECURITY_ACCESS_INVALID_KEY
[ECUC_Dcm_01207]	Definition of EcucReferenceDef SEV_UDS_SECURITY_ACCESS_REQUIRED_TIME_DELAY_NOT_EXPIRED
[ECUC_Dcm_01208]	Definition of EcucReferenceDef SEV_UDS_NUMBER_OF_FAILED_AUTHENTICATION_ATTEMPTS_EXCEEDED
[ECUC_Dcm_01209]	Definition of EcucReferenceDef SEV_UDS_CERTIFICATE_FAILURE
[SWS_Dcm_00036]	
[SWS_Dcm_00144]	
[SWS_Dcm_00145]	
[SWS_Dcm_00146]	
[SWS_Dcm_00147]	
[SWS_Dcm_00459]	Callback notification for preempted protocols





Number	Heading
[SWS_Dcm_00624]	
[SWS_Dcm_00625]	
[SWS_Dcm_00668]	
[SWS_Dcm_00669]	
[SWS_Dcm_00670]	
[SWS_Dcm_00671]	
[SWS_Dcm_00672]	
[SWS_Dcm_00673]	
[SWS_Dcm_00674]	
[SWS_Dcm_00692]	Definition of ClientServerInterface CallbackDCMRequestServices
[SWS_Dcm_00728]	
[SWS_Dcm_00775]	
[SWS_Dcm_00943]	
[SWS_Dcm_00945]	
[SWS_Dcm_00949]	
[SWS_Dcm_01033]	Definition of Port CallbackDCMRequestServices_{Name} required by module Dcm
[SWS_Dcm_01190]	
[SWS_Dcm_01194]	
[SWS_Dcm_01339]	Definition of configurable interface Xxx_StartProtocol
[SWS_Dcm_01340]	Definition of configurable interface Xxx_StopProtocol
[SWS_Dcm_01394]	

Table A.8: Deleted Specification Items in R24-11

A.2.4 Added Constraints in R24-11

none

A.2.5 Changed Constraints in R24-11

Number	Heading
[SWS_Dcm_CONSTR_06002]	Existence of size parameter
[SWS_Dcm_CONSTR_06008]	Define the usage of DcmDspRoutineParameterSize parameter





Number	Heading
[SWS_Dcm_CONSTR_06105]	
[SWS_Dcm_CONSTR_06106]	
[SWS_Dcm_CONSTR_06107]	

Table A.9: Changed Constraints in R24-11

A.2.6 Deleted Constraints in R24-11

none

A.3 Traceable item history of this document according to AUTOSAR Release R23-11

A.3.1 Added Specification Items in R23-11

Number	Heading
[SWS_Dcm_01641]	Support of UDS Services
[SWS_Dcm_01642]	Subfunction 0x01, 0x07 and 0x12 response values
[SWS_Dcm_01643]	Dem_SetDTCFilter() parameters values for subfunctions 0x01, 0x07 and 0x12
[SWS_Dcm_01644]	Subfunction 0x02, 0x0A, 0x13, 0x15 and 0x17 response values
[SWS_Dcm_01645]	Dem_SetDTCFilter() parameters values for subfunctions 0x02, 0x0A, 0x13, 0x15 and 0x17
[SWS_Dcm_01646]	Subfunction 0x08 response values
[SWS_Dcm_01647]	Dem_SetDTCFilter() parameters values for Subfunction 0x08
[SWS_Dcm_01648]	Subfunction 0x09 response values
[SWS_Dcm_01649]	Dem_SelectDTC() parameters values for subfunctions 0x06 and 0x19
[SWS_Dcm_01650]	Dem_SelectExtendedDataRecord() parameters values for subfunctions 0x06 and 0x19
[SWS_Dcm_01651]	Subfunction 0x05 response values
[SWS_Dcm_01652]	Subfunctions 0x0B, 0x0C, 0x0D and 0x0E response values
[SWS_Dcm_01653]	Dem_GetDTCByOccurrenceTime() parameters values for subfunctions 0x0B, 0x0C, 0x0D and 0x0E
[SWS_Dcm_01654]	Subfunction 0x14 response values





Number	Heading
[SWS_Dcm_01655]	Dem_GetNextFilteredDTCAndFDC() parameters values for subfunctions 0x14
[SWS_Dcm_01656]	Subfunction 0x42 response values
[SWS_Dcm_01657]	Dem_GetNextFilteredDTCAndSeverity() parameters values for subfunctions 0x42
[SWS_Dcm_01658]	Dem_GetNextFilteredDTCAndSeverity return values
[SWS_Dcm_01659]	Subfunction 0x55 response values
[SWS_Dcm_01660]	Dem_SetDTCFilter() parameters values for subfunction 0x55
[SWS_Dcm_01661]	InputOutputControlParameter definitions
[SWS_Dcm_01662]	NRC managed by callout function for service 0x34
[SWS_Dcm_01663]	NRC managed by callout function for service 0x35
[SWS_Dcm_01664]	NRC managed by callout function for service 0x37
[SWS_Dcm_01665]	NRC managed by callout function for service 0x36
[SWS_Dcm_01666]	Concurrent tester present support
[SWS_Dcm_01667]	Concurrent tester present from different connections
[SWS_Dcm_01668]	S3Server session timer support
[SWS_Dcm_01669]	Sessions and connections
[SWS_Dcm_01670]	Change to default session upon S3Server elapses
[SWS_Dcm_01671]	Message reception according to ISO 14229-2
[SWS_Dcm_01672]	Mapping of T_DataSOM.ind
[SWS_Dcm_01673]	Mapping of T_Data.ind
[SWS_Dcm_01674]	Multi client handling
[SWS_Dcm_01675]	Protocol resource availability
[SWS_Dcm_01676]	Decline a second request with NRC 0x21
[SWS_Dcm_01677]	Ignoring a second request
[SWS_Dcm_01678]	Multiple requests on the same connection
[SWS_Dcm_01679]	Default S3 timeout value
[SWS_Dcm_01680]	Overwritten S3 timeout value
[SWS_Dcm_01681]	Support of subfunction transmitCertificate
[SWS_Dcm_01682]	Verify certificateEvaluationId
[SWS_Dcm_01683]	Certificate storage parsing and verification
[SWS_Dcm_01684]	Starting the secure coding sequence
[SWS_Dcm_01685]	Write Secure Coding DID for started secure coding sequence
[SWS_Dcm_01686]	Write Secure Coding DID without a started secure coding sequence
[SWS_Dcm_01687]	Write Secure Coding DID buffer overflow
[SWS_Dcm_01688]	Secure Coding validation trigger on inactive secure coding sequence
[SWS_Dcm_01689]	Secure Coding validation trigger without written data
[SWS_Dcm_01690]	Dcm internal processing of secure coding validation routine
[SWS_Dcm_01691]	Authentication of secure coding data





Number	Heading
[SWS_Dcm_01692]	Successful authenticated secure coding data
[SWS_Dcm_01693]	Successfully completed secure coding procedure
[SWS_Dcm_01694]	Failed secure coding write
[SWS_Dcm_01695]	Wrong signature for secure coding data
[SWS_Dcm_01696]	Behavior on too long signature validation
[SWS_Dcm_91075]	Definition of Port AuthenticationStateModeSwitchInterface_{Connection} provided by module Dcm

Table A.10: Added Specification Items in R23-11

A.3.2 Changed Specification Items in R23-11

Number	Heading
[SWS_Dcm_00027]	
[SWS_Dcm_00030]	
[SWS_Dcm_00092]	Definition of API function Dcm_CopyTxData
[SWS_Dcm_00094]	Definition of callback function Dcm_StartOfReception
[SWS_Dcm_00143]	
[SWS_Dcm_00293]	
[SWS_Dcm_00295]	
[SWS_Dcm_00333]	Definition of imported datatypes of module Dcm
[SWS_Dcm_00338]	Definition of API function Dcm_GetSecurityLevel
[SWS_Dcm_00339]	Definition of API function Dcm_GetSesCtrlType
[SWS_Dcm_00376]	
[SWS_Dcm_00377]	
[SWS_Dcm_00378]	
[SWS_Dcm_00379]	
[SWS_Dcm_00380]	
[SWS_Dcm_00381]	
[SWS_Dcm_00382]	
[SWS_Dcm_00392]	
[SWS_Dcm_00442]	
[SWS_Dcm_00464]	
[SWS_Dcm_00465]	
[SWS_Dcm_00466]	
[SWS_Dcm_00520]	Definition of API function Dcm_ResetToDefaultSession
[SWS_Dcm_00539]	Definition of configurable interface Dcm_ReadMemory
[SWS_Dcm_00540]	Definition of configurable interface Dcm_WriteMemory





Number	Heading
[SWS_Dcm_00543]	Definition of configurable interface Dcm_SetProgConditions
[SWS_Dcm_00544]	Definition of configurable interface Dcm_GetProgConditions
[SWS_Dcm_00556]	Definition of API function Dcm_CopyRxData
[SWS_Dcm_00574]	
[SWS_Dcm_00579]	
[SWS_Dcm_00614]	Definition of API function Dcm_DemTriggerOnDTCStatus
[SWS_Dcm_00641]	
[SWS_Dcm_00685]	Definition of ClientServerInterface SecurityAccess_{SecurityLevel}
[SWS_Dcm_00686]	Definition of ClientServerInterface DataServices_{Data}
[SWS_Dcm_00687]	Definition of SenderReceiverInterface DataServices_{Data}
[SWS_Dcm_00688]	Definition of ClientServerInterface InfotypeServices_{VehInfoData}
[SWS_Dcm_00690]	Definition of ClientServerInterface RoutineServices_{RoutineName}
[SWS_Dcm_00691]	Definition of ClientServerInterface RequestControlServices_{Tid}
[SWS_Dcm_00694]	Definition of ClientServerInterface ServiceRequestNotification
[SWS_Dcm_00698]	Definition of ClientServerInterface DCMServices
[SWS_Dcm_00754]	Definition of configurable interface Dcm_ProcessRequestDownload
[SWS_Dcm_00755]	Definition of configurable interface Dcm_ProcessRequestTransferExit
[SWS_Dcm_00756]	Definition of configurable interface Dcm_ProcessRequestUpload
[SWS_Dcm_00763]	Definition of configurable interface <Module>_<DiagnosticService>
[SWS_Dcm_00764]	Definition of configurable interface <Module>_<DiagnosticService>_<Sub Service>
[SWS_Dcm_00769]	Definition of ClientServerInterface DataServices_DIDRange_{Range}
[SWS_Dcm_00793]	Definition of configurable interface Xxx_ReadData
[SWS_Dcm_00794]	Definition of configurable interface Xxx_WriteData
[SWS_Dcm_00796]	Definition of configurable interface Xxx_ReadDataLength
[SWS_Dcm_00797]	Definition of configurable interface Xxx_ConditionCheckRead
[SWS_Dcm_00798]	Definition of configurable interface Xxx_GetScalingInformation
[SWS_Dcm_00802]	Definition of configurable interface Xxx_ShortTermAdjustment
[SWS_Dcm_00803]	Definition of configurable interface Xxx_IsDidAvailable
[SWS_Dcm_00804]	Definition of configurable interface Xxx_ReadDidData
[SWS_Dcm_00805]	Definition of configurable interface Xxx_WriteDidData
[SWS_Dcm_00950]	Definition of API function Dcm_GetVin
[SWS_Dcm_00976]	Definition of datatype Dcm_StatusType
[SWS_Dcm_00977]	Definition of ImplementationDataType Dcm_SecLevelType
[SWS_Dcm_00978]	Definition of ImplementationDataType Dcm_SesCtrlType
[SWS_Dcm_00979]	Definition of ImplementationDataType Dcm_ProtocolType
[SWS_Dcm_00980]	Definition of ImplementationDataType Dcm_NegativeResponseCodeType
[SWS_Dcm_00981]	Definition of datatype Dcm_CommunicationModeType





Number	Heading
[SWS_Dcm_00982]	Definition of datatype Dcm_ConfigType
[SWS_Dcm_00983]	Definition of ImplementationDataType Dcm_ConfirmationStatusType
[SWS_Dcm_00984]	Definition of ImplementationDataType Dcm_OpStatusType
[SWS_Dcm_00985]	Definition of datatype Dcm_ReturnReadMemoryType
[SWS_Dcm_00986]	Definition of datatype Dcm_ReturnWriteMemoryType
[SWS_Dcm_00987]	Definition of datatype Dcm_EcuStartModeType
[SWS_Dcm_00988]	Definition of datatype Dcm_ProgConditionsType
[SWS_Dcm_00989]	Definition of datatype Dcm_MsgItemtype
[SWS_Dcm_00990]	Definition of datatype Dcm_MsgType
[SWS_Dcm_00991]	Definition of datatype Dcm_MsgLenType
[SWS_Dcm_00993]	Definition of datatype Dcm_IdContextType
[SWS_Dcm_00994]	Definition of datatype Dcm_MsgContextType
[SWS_Dcm_01012]	Definition of ImplementationDataType Dcm_RangeArray_{Range}Type
[SWS_Dcm_01013]	Definition of ImplementationDataType Dcm_InfoTypeServicesArray_{VehInfo Data}Type
[SWS_Dcm_01014]	Definition of ImplementationDataType Dcm_RequestControlServicesIn Array_{Tid}Type
[SWS_Dcm_01015]	Definition of ImplementationDataType Dcm_RequestControlServicesOut Array_{Tid}Type
[SWS_Dcm_01017]	Definition of ImplementationDataType Dcm_ScalingInfoArray_{Data}Type
[SWS_Dcm_01018]	Definition of ImplementationDataType Dcm_RequestData Out_{Routine}_{Signal}PrimitivType
[SWS_Dcm_01019]	Definition of ImplementationDataType Dcm_RequestFlexibleOutArray Data_{Routine}_{Signal}Type
[SWS_Dcm_01020]	Definition of ImplementationDataType Dcm_StartData In_{Routine}_{Signal}PrimitivType
[SWS_Dcm_01021]	Definition of ImplementationDataType Dcm_StartData Out_{Routine}_{Signal}PrimitivType
[SWS_Dcm_01022]	Definition of ImplementationDataType Dcm_StartFlexibleInArray Data_{Routine}_{Signal}Type
[SWS_Dcm_01023]	Definition of ImplementationDataType Dcm_StartFlexibleOutArray Data_{Routine}_{Signal}Type
[SWS_Dcm_01024]	Definition of ImplementationDataType Dcm_StopData In_{Routine}_{Signal}PrimitivType
[SWS_Dcm_01025]	Definition of ImplementationDataType Dcm_StopData Out_{Routine}_{Signal}PrimitivType
[SWS_Dcm_01026]	Definition of ImplementationDataType Dcm_StopFlexibleInArray Data_{Routine}_{Signal}Type
[SWS_Dcm_01027]	Definition of ImplementationDataType Dcm_StopFlexibleOutArray Data_{Routine}_{Signal}Type
[SWS_Dcm_01028]	Definition of ImplementationDataType Dcm_KeyArray_{SecurityLevel}Type





Number	Heading
[SWS_Dcm_01029]	Definition of ImplementationDataType Dcm_SeedArray_{SecurityLevel}Type
[SWS_Dcm_01030]	Definition of Port DCMServices provided by module Dcm
[SWS_Dcm_01031]	Definition of Port DataServices_{Data} provided by module Dcm
[SWS_Dcm_01034]	Definition of Port DataServices_DIDRange_{Range} required by module Dcm
[SWS_Dcm_01035]	Definition of Port DataServices_{Data} required by module Dcm
[SWS_Dcm_01037]	Definition of Port InfotypeServices_{VehInfoData} required by module Dcm
[SWS_Dcm_01038]	Definition of Port RequestControlServices_{Tid} required by module Dcm
[SWS_Dcm_01039]	Definition of Port ServiceRequestManufacturerNotification_{Name} required by module Dcm
[SWS_Dcm_01040]	Definition of Port RoutineServices_{RoutineName} required by module Dcm
[SWS_Dcm_01041]	Definition of Port SecurityAccess_{SecurityLevel} required by module Dcm
[SWS_Dcm_01042]	Definition of Port ServiceRequestSupplierNotification_{Name} required by module Dcm
[SWS_Dcm_01068]	Definition of API function Dcm_SetActiveDiagnostic
[SWS_Dcm_01121]	Definition of ImplementationDataType Dcm_DataArrayTypeUint8_{Data}Type
[SWS_Dcm_01129]	
[SWS_Dcm_01130]	
[SWS_Dcm_01131]	
[SWS_Dcm_01132]	
[SWS_Dcm_01133]	
[SWS_Dcm_01134]	
[SWS_Dcm_01138]	Definition of ImplementationDataType Dcm_DidSupportedType
[SWS_Dcm_01151]	Definition of configurable interface Xxx_GetSeed
[SWS_Dcm_01152]	Definition of configurable interface Xxx_GetSecurityAttemptCounter
[SWS_Dcm_01153]	Definition of configurable interface Xxx_SetSecurityAttemptCounter
[SWS_Dcm_01159]	Definition of ImplementationDataType Dcm_SecurityAccessDataRecord Array_{SecurityLevel}Type
[SWS_Dcm_01165]	Definition of ImplementationDataType Dcm_RequestDataArrayType
[SWS_Dcm_01173]	
[SWS_Dcm_01203]	Definition of configurable interface Xxx_Start
[SWS_Dcm_01204]	Definition of configurable interface Xxx_Stop
[SWS_Dcm_01270]	Definition of configurable interface Xxx_ClearDTCCheckFnc
[SWS_Dcm_01271]	Definition of configurable interface Xxx_ReadDidRangeDataLength
[SWS_Dcm_01273]	
[SWS_Dcm_01285]	Definition of configurable interface Xxx_ReturnControlToECU
[SWS_Dcm_01286]	Definition of configurable interface Xxx_ResetToDefault
[SWS_Dcm_01290]	Definition of configurable interface Xxx_FreezeCurrentState
[SWS_Dcm_01305]	Definition of ImplementationDataType Dcm_InputOutputControlParameter Type





Number	Heading
[SWS_Dcm_01306]	Definition of ImplementationDataType Dcm_IOOperationRequest_{DID}Type
[SWS_Dcm_01307]	Definition of ImplementationDataType Dcm_IOOperationResponseType
[SWS_Dcm_01308]	Definition of SenderReceiverInterface IOControlRequest_{DID}
[SWS_Dcm_01309]	Definition of SenderReceiverInterface IOControlResponse_{DID}
[SWS_Dcm_01310]	Definition of Port DataServices_{Data} required by module Dcm
[SWS_Dcm_01311]	Definition of Port DataServices_{Data} provided/required by module Dcm
[SWS_Dcm_01312]	Definition of Port IOControlRequest_{DID} provided/required by module Dcm
[SWS_Dcm_01313]	Definition of Port IOControlResponse_{DID} required by module Dcm
[SWS_Dcm_01314]	Definition of configurable interface Xxx_ResetToDefault
[SWS_Dcm_01315]	Definition of configurable interface Xxx_FreezeCurrentState
[SWS_Dcm_01316]	Definition of configurable interface Xxx_ShortTermAdjustment
[SWS_Dcm_01320]	Definition of ImplementationDataType Dcm_ControlMask_{DID}Type
[SWS_Dcm_01338]	Definition of configurable interface Xxx_RequestControl
[SWS_Dcm_01341]	Definition of configurable interface Xxx_Indication
[SWS_Dcm_01342]	Definition of configurable interface Xxx_Confirmation
[SWS_Dcm_01345]	
[SWS_Dcm_01415]	Behavior on application returning unexpected return code
[SWS_Dcm_01590]	Security events for Dcm
[SWS_Dcm_91001]	Definition of mandatory interfaces in module Dcm
[SWS_Dcm_91002]	Definition of optional interfaces in module Dcm
[SWS_Dcm_91003]	Definition of configurable interface Xxx_GetSeed
[SWS_Dcm_91004]	Definition of configurable interface Xxx_CompareKey
[SWS_Dcm_91005]	Definition of configurable interface Xxx_ReadData
[SWS_Dcm_91006]	Definition of configurable interface Xxx_ReadData
[SWS_Dcm_91007]	Definition of configurable interface Xxx_WriteData
[SWS_Dcm_91008]	Definition of configurable interface Xxx_WriteData
[SWS_Dcm_91009]	Definition of configurable interface Xxx_WriteData
[SWS_Dcm_91010]	Definition of configurable interface Xxx_ReadDataLength
[SWS_Dcm_91011]	Definition of configurable interface Xxx_ConditionCheckRead
[SWS_Dcm_91012]	Definition of configurable interface Xxx_GetScalingInformation
[SWS_Dcm_91013]	Definition of configurable interface Xxx_RequestResults
[SWS_Dcm_91014]	Definition of configurable interface Xxx_GetInfotypeValueData
[SWS_Dcm_91015]	Definition of datatype Dcm_ExtendedOpStatusType
[SWS_Dcm_91016]	Definition of configurable interface Xxx_StartConfirmation
[SWS_Dcm_91017]	Definition of configurable interface Xxx_StopConfirmation
[SWS_Dcm_91018]	Definition of configurable interface Xxx_RequestResultsConfirmation
[SWS_Dcm_91019]	Definition of ModeDeclarationGroup DcmDiagnosticSessionControl





Number	Heading
[SWS_Dcm_91020]	Definition of ModeSwitchInterface Dcm_DiagnosticSessionControlModeSwitchInterface
[SWS_Dcm_91021]	Definition of ModeDeclarationGroup DcmEcuReset
[SWS_Dcm_91022]	Definition of ModeSwitchInterface Dcm_EcuResetModeSwitchInterface
[SWS_Dcm_91023]	Definition of ModeDeclarationGroup DcmModeRapidPowerShutDown
[SWS_Dcm_91024]	Definition of ModeSwitchInterface Dcm_ModeRapidPowerShutDownModeSwitchInterface
[SWS_Dcm_91025]	Definition of ModeDeclarationGroup DcmCommunicationControl
[SWS_Dcm_91026]	Definition of ModeSwitchInterface CommunicationControlModeSwitchInterface_{ComMChannelName}
[SWS_Dcm_91027]	Definition of ModeDeclarationGroup DcmControlDTCSetting
[SWS_Dcm_91028]	Definition of ModeSwitchInterface Dcm_ControlDTCSettingModeSwitchInterface
[SWS_Dcm_91031]	Definition of ModeDeclarationGroup DcmSecurityAccess
[SWS_Dcm_91032]	Definition of ModeSwitchInterface Dcm_SecurityAccessModeSwitchInterface
[SWS_Dcm_91033]	Definition of Port DiagnosticSessionControlModeSwitchInterface provided by module Dcm
[SWS_Dcm_91034]	Definition of Port EcuResetModeSwitchInterface provided by module Dcm
[SWS_Dcm_91035]	Definition of Port ModeRapidPowerShutDownModeSwitchInterface provided by module Dcm
[SWS_Dcm_91036]	Definition of Port CommunicationControlModeSwitchInterface_{ComMChannelName} provided by module Dcm
[SWS_Dcm_91037]	Definition of Port ControlDTCSettingModeSwitchInterface provided by module Dcm
[SWS_Dcm_91039]	Definition of Port SecurityAccessModeSwitchInterface provided by module Dcm
[SWS_Dcm_91040]	Definition of ImplementationDataType Dcm_RequestDataOut_{Routine}_{Signal}Type
[SWS_Dcm_91041]	Definition of ImplementationDataType Dcm_RequestDataOut_{Routine}_{Signal}ArrayType
[SWS_Dcm_91042]	Definition of ImplementationDataType Dcm_StartDataIn_{Routine}_{Signal}Type
[SWS_Dcm_91043]	Definition of ImplementationDataType Dcm_StartDataIn_{Routine}_{Signal}ArrayType
[SWS_Dcm_91044]	Definition of ImplementationDataType Dcm_StartDataOut_{Routine}_{Signal}Type
[SWS_Dcm_91045]	Definition of ImplementationDataType Dcm_StartDataOut_{Routine}_{Signal}ArrayType
[SWS_Dcm_91046]	Definition of ImplementationDataType Dcm_StopDataIn_{Routine}_{Signal}Type
[SWS_Dcm_91047]	Definition of ImplementationDataType Dcm_StopDataIn_{Routine}_{Signal}ArrayType





Number	Heading
[SWS_Dcm_91048]	Definition of ImplementationDataType Dcm_StopData Out_{Routine}_{Signal}Type
[SWS_Dcm_91049]	Definition of ImplementationDataType Dcm_StopData Out_{Routine}_{Signal}ArrayType
[SWS_Dcm_91050]	Definition of ImplementationDataType Dcm_ControlMask_{Data}ArrayType
[SWS_Dcm_91051]	Definition of ImplementationDataType Dcm_DataElement_{Data}Type
[SWS_Dcm_91052]	Definition of ImplementationDataType Dcm_RequestData In_{Routine}_{Signal}Type
[SWS_Dcm_91053]	Definition of ImplementationDataType Dcm_RequestFlexibleInArray Data_{Routine}_{Signal}Type
[SWS_Dcm_91054]	Definition of ImplementationDataType Dcm_RequestData In_{Routine}_{Signal}PrimitiveType
[SWS_Dcm_91055]	Definition of ImplementationDataType Dcm_RequestData In_{Routine}_{Signal}ArrayType
[SWS_Dcm_91057]	Definition of SenderReceiverInterface DataServices_{DID}
[SWS_Dcm_91058]	Definition of Port DataServices_{DID} provided by module Dcm
[SWS_Dcm_91059]	Definition of Port DataServices_{DID} required by module Dcm
[SWS_Dcm_91060]	Definition of Port DataServices_{DID} provided/required by module Dcm
[SWS_Dcm_91061]	Definition of SenderReceiverInterface DataServices_{DID}
[SWS_Dcm_91062]	Definition of ImplementationDataType Dcm_DataElement_{Data}_Primitive Type
[SWS_Dcm_91063]	Definition of ImplementationDataType Dcm_DataElement_{Data}_ArrayType
[SWS_Dcm_91064]	Definition of ImplementationDataType Dcm_ResponseDataArrayType
[SWS_Dcm_91065]	Definition of ClientServerInterface UploadDownloadServices
[SWS_Dcm_91066]	Definition of ImplementationDataType Dcm_FileAndDirNameType
[SWS_Dcm_91067]	Definition of ModeDeclarationGroup DcmAuthenticationState_{Connection}
[SWS_Dcm_91068]	Definition of ImplementationDataType Dcm_AuthenticationRoleType
[SWS_Dcm_91069]	Definition of API function Dcm_SetDeauthenticatedRole
[SWS_Dcm_91070]	Definition of configurable interface Dcm_ProcessTransferDataRead
[SWS_Dcm_91071]	Definition of configurable interface Dcm_ProcessTransferDataWrite
[SWS_Dcm_91072]	Definition of ClientServerInterface Authentication
[SWS_Dcm_91073]	Definition of Port Authentication_{Connection} required by module Dcm
[SWS_Dcm_91074]	Definition of ModeSwitchInterface Dcm_AuthenticationStateModeSwitch Interface_{Connection}
[SWS_Dcm_91076]	Definition of configurable interface Dcm_CsmAsyncJobFinished
[SWS_Dcm_91077]	Definition of configurable interface Dcm_KeyMAsyncCertificateVerify Finished
[SWS_Dcm_91078]	Definition of configurable interface Dcm_ProcessRequestAddFile
[SWS_Dcm_91079]	Definition of configurable interface Dcm_ProcessRequestDeleteFile
[SWS_Dcm_91080]	Definition of configurable interface Dcm_ProcessRequestReplaceFile





Number	Heading
[SWS_Dcm_91081]	Definition of configurable interface Dcm_ProcessRequestReadFile
[SWS_Dcm_91082]	Definition of configurable interface Dcm_ProcessRequestReadDir
[SWS_Dcm_91083]	Definition of configurable interface Dcm_WriteFile
[SWS_Dcm_91084]	Definition of Port UploadDownloadServices required by module Dcm
[SWS_Dcm_91085]	Definition of configurable interface Dcm_ReadFileOrDir
[SWS_Dcm_91087]	Definition of ImplementationDataType Dcm_Cemr_{DID}Type
[SWS_Dcm_91089]	Definition of ImplementationDataType Dcm_ControlMask_{Data}_Primitive Type
[SWS_Dcm_91090]	Definition of configurable interface Xxx_ReadData
[SWS_Dcm_91091]	Definition of configurable interface Xxx_ReadData
[SWS_Dcm_91092]	Definition of configurable interface Xxx_WriteData
[SWS_Dcm_91093]	Definition of configurable interface Xxx_GetScalingInformation
[SWS_Dcm_91094]	Definition of configurable interface Xxx_GetScalingInformation
[SWS_Dcm_91095]	Definition of configurable interface Xxx_WriteData
[SWS_Dcm_91096]	Definition of configurable interface Xxx_ReadDataLength
[SWS_Dcm_91097]	Definition of configurable interface Xxx_ReadDataLength
[SWS_Dcm_91098]	Definition of configurable interface Xxx_ReturnControlToECU
[SWS_Dcm_91099]	Definition of configurable interface Xxx_ResetToDefault
[SWS_Dcm_91100]	Definition of configurable interface Xxx_ResetToDefault
[SWS_Dcm_91101]	Definition of configurable interface Xxx_FreezeCurrentState
[SWS_Dcm_91102]	Definition of configurable interface Xxx_FreezeCurrentState
[SWS_Dcm_91103]	Definition of configurable interface Xxx_ShortTermAdjustment
[SWS_Dcm_91104]	Definition of configurable interface Xxx_ShortTermAdjustment
[SWS_Dcm_91105]	Definition of configurable interface Xxx_Start
[SWS_Dcm_91106]	Definition of configurable interface Xxx_Stop
[SWS_Dcm_91107]	Definition of configurable interface Xxx_RequestResults
[SWS_Dcm_91118]	Definition of ImplementationDataType Dcm_DataElement_{Data}_Structured Type
[SWS_Dcm_91119]	Definition of ImplementationDataType Dcm_RequestData Out_{Routine}_{Signal}StructuredType
[SWS_Dcm_91120]	Definition of ImplementationDataType Dcm_RequestData In_{Routine}_{Signal}StructuredType
[SWS_Dcm_91121]	Definition of ImplementationDataType Dcm_StartData In_{Routine}_{Signal}StructuredType
[SWS_Dcm_91122]	Definition of ImplementationDataType Dcm_StartData Out_{Routine}_{Signal}StructuredType
[SWS_Dcm_91123]	Definition of ImplementationDataType Dcm_StopData In_{Routine}_{Signal}StructuredType





Number	Heading
[SWS_Dcm_91124]	Definition of ImplementationDataType Dcm_StopData Out_{Routine}_{Signal}StructuredType

Table A.11: Changed Specification Items in R23-11

A.3.3 Deleted Specification Items in R23-11

Number	Heading
[SWS_Dcm_00112]	
[SWS_Dcm_00113]	
[SWS_Dcm_00140]	
[SWS_Dcm_00141]	
[SWS_Dcm_00541]	
[SWS_Dcm_00557]	
[SWS_Dcm_00560]	
[SWS_Dcm_00655]	
[SWS_Dcm_00656]	
[SWS_Dcm_00788]	
[SWS_Dcm_00789]	
[SWS_Dcm_00790]	
[SWS_Dcm_01128]	
[SWS_Dcm_01145]	
[SWS_Dcm_01146]	
[SWS_Dcm_01168]	

Table A.12: Deleted Specification Items in R23-11

A.3.4 Added Constraints in R23-11

Number	Heading
[SWS_Dcm_CONSTR_06108]	Not supported port interfaces for secure coding
[SWS_Dcm_CONSTR_06109]	Static length of secure coding DIDs





Number	Heading
[SWS_Dcm_CONSTR_06110]	Secure Coding validation routine input parameter
[SWS_Dcm_CONSTR_06111]	Secure Coding validation routine output parameter
[SWS_Dcm_CONSTR_06112]	Secure Coding validation routine is synchronous
[SWS_Dcm_CONSTR_06113]	Require asynchronous signature verification

Table A.13: Added Constraints in R23-11

A.3.5 Changed Constraints in R23-11

Number	Heading
[SWS_Dcm_CONSTR_06026]	Usage of variable data length in case of S/R communication, NvRam access or ECU signal access

Table A.14: Changed Constraints in R23-11

A.3.6 Deleted Constraints in R23-11

Number	Heading
[SWS_Dcm_CONSTR_06038]	Restrictions on datatype usage
[SWS_Dcm_CONSTR_06067]	Dependency for DcmDspDataBlockIdRef

Table A.15: Deleted Constraints in R23-11

A.4 Not applicable requirements

[SWS_Dcm_NA_00999]

Upstream requirements: SRS_BSW_00159, SRS_BSW_00167, SRS_BSW_00168, SRS_BSW_00170, SRS_BSW_00171, SRS_BSW_00323, SRS_BSW_00336, SRS_BSW_00339, SRS_BSW_00344, SRS_BSW_00345, SRS_BSW_00375, SRS_BSW_00380, SRS_BSW_00383, SRS_BSW_00384, SRS_BSW_00385, SRS_BSW_00386, SRS_BSW_00388, SRS_BSW_00389, SRS_BSW_00390, SRS_BSW_00392, SRS_BSW_00393, SRS_BSW_00395, SRS_BSW_00396, SRS_BSW_00397, SRS_BSW_00398, SRS_BSW_00399, SRS_BSW_00400, SRS_BSW_00402, SRS_BSW_00403, SRS_BSW_00404, SRS_BSW_00405, SRS_BSW_00406, SRS_BSW_00409, SRS_BSW_00416, SRS_BSW_00417, SRS_BSW_00419, SRS_BSW_00422, SRS_BSW_00423, SRS_BSW_00425, SRS_BSW_00426, SRS_BSW_00427, SRS_BSW_00428, SRS_BSW_00429, SRS_BSW_00432, SRS_BSW_00433, SRS_BSW_00437, SRS_BSW_00450, SRS_BSW_00451, SRS_BSW_00458, SRS_BSW_00461, SRS_BSW_00466, SRS_BSW_00467, SRS_BSW_00469, SRS_BSW_00470, SRS_BSW_00471, SRS_BSW_00472, SRS_BSW_00478, RS_Diag_04002, RS_Diag_04019, RS_Diag_04024, RS_Diag_04031, RS_Diag_04032, RS_Diag_04057, RS_Diag_04063, RS_Diag_04064, RS_Diag_04068, RS_Diag_04071, RS_Diag_04077, RS_Diag_04091, RS_Diag_04093, RS_Diag_04097, RS_Diag_04105, RS_Diag_04107, RS_Diag_04109, RS_Diag_04110, RS_Diag_04111, RS_Diag_04112, RS_Diag_04113, RS_Diag_04117, RS_Diag_04118, RS_Diag_04120, RS_Diag_04121, RS_Diag_04123, RS_Diag_04124, RS_Diag_04125, RS_Diag_04126, RS_Diag_04127, RS_Diag_04129, RS_Diag_04131, RS_Diag_04133, [RS_Diag_04135](#), RS_Diag_04136, RS_Diag_04137, RS_Diag_04139, RS_Diag_04140, RS_Diag_04148, RS_Diag_04150, RS_Diag_04151

[These requirements are not applicable to this specification.]