Document Title	Acceptance Test Specification of Diagnostic Services
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	627
Document Classification	Auxiliary

Document Status	Final
Part of AUTOSAR Standard	Acceptance Tests for Classic Platform
Part of Standard Release	1.2.0

	Do	cument (Change History
Date	Release	Changed by	Change Description
2016-12-15	1.2.0	AUTOSAR Release Management	Checked and adapted to Classic Platform Release 4.2.2
2015-10-31	1.1.0	AUTOSAR Release Management	 Formalized PCO in test cases Checked and adapted to Classic Platform Release 4.2.1 Fixed bug in test case ATS_DIAG_00034: Wrong value for TestNotCompletedSinceLastClear
2014-07-30	1.0.0	AUTOSAR Release Management	Initial release



This specification 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 specification.

The material contained in this specification is protected by copyright and other types of Intellectual Property Rights. The commercial exploitation of the material contained in this specification requires a license to such Intellectual Property Rights.

This specification 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 specification may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The AUTOSAR specifications have been developed for automotive applications only. They have neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Advice for users

AUTOSAR specifications may contain exemplary items (exemplary reference models, "use cases", and/or references to exemplary technical solutions, devices, processes or software).

Any such exemplary items are contained in the specifications for illustration purposes only, and they themselves are not part of the AUTOSAR Standard. Neither their presence in such specifications, nor any later documentation of AUTOSAR conformance of products actually implementing such exemplary items, imply that intellectual property rights covering such exemplary items are licensed under the same rules as applicable to the AUTOSAR Standard.

Table of Contents

1	Acronyms and abbreviations	5
2	Scope	6
3	General Test Objective and Approach	7
	3.1 Test System	
	3.1.1 Overview on Architecture	7 7
	3.2 Test Configuration	8
	3.2.1 Required ECU Extract of System Description Files 3.2.2 Required ECU Configuration Description Files 3.2.3 Required Software Component Description Files 3.2.4 Mandatory vs. Customizable Parts	8 8
	3.3 Test Case Design	
	3.3.1 DEM related test cases	
4	Re-usable Test Steps	10
5	Test Cases	11
	5.1 Test Cases on DiagnosticMonitor (Dem) 5.1.1 [ATS_DIAG_00032] Test Step Group: Reset event	11 12 13 15 16 18 21 25 26 28
	5.2 Test Cases on DataServices (Dcm)	33
	5.2.1 [ATS_DIAG_00022] Writing and reading of data with fixed length	35 37

Acceptance Test Specification of Diagnostic Services AUTOSAR TC Release 1.2.0

5.2.5	[ATS_DIAG_00026] Writing and reading of data with endianness	
	conversion	40
5.2.6	[ATS_DIAG_00027] Retrieving of scaling information	41
5.2.7	[ATS_DIAG_00028] Periodical reading of data	43
5.2.8	[ATS_DIAG_00029] Invocation of callbacks for	
	"InputOutputControlByIdentifier"	45
5.3 Te	est Cases on RoutineServices (Dcm)	49
	[ATS_DIAG_00030] Handling of RoutineControl with fixed-length data	49
5.3.2	[ATS_DIAG_00031] Handling of RoutineControl with dynamic-length	
	data	51

1 Acronyms and abbreviations

Abbreviation /	Description:
Acronym:	
AT	Acceptance Test
CAN	Controller Area Network
ECU	Electronic Control Unit
ICC1	Implementation Conformance Class 1 (whole BSW & RTE)
ICC2	Implementation Conformance Class 2 (functional cluster of BSW)
ICC3	Implementation Conformance Class 3 (individual BSW modules)
IUT	Implementation under test
LT	Lower Tester
NM	Network Management
PCO	Point of Control and Observation
PDU	Protocol Data Unit
RfC	Request for Change
Rx	Reception
SUT	System Under Test
SWC	Software Component
TCP	Test Coordination Procedures
Тх	Transmission
UT	Upper Tester

2 Scope

The following test cases are used to verify the correct behavior of the diagnostic services.

Each test case documents for which releases of the AUTOSAR software specification it can be used:

- When test cases are known to be applicable for a release, this is mentioned in the "AUTOSAR Releases" field of the test case specifications.
 You can find a summary of the applicability of all test cases to the software specification releases in the "AUTOSAR_TR_ATSReleaseApplicability" document.
- When test cases are known to require adaptations (in their configuration requirements or test sequences), this is mentioned in the "Needed Adaptation to other Releases" field of the test case specifications.

3 General Test Objective and Approach

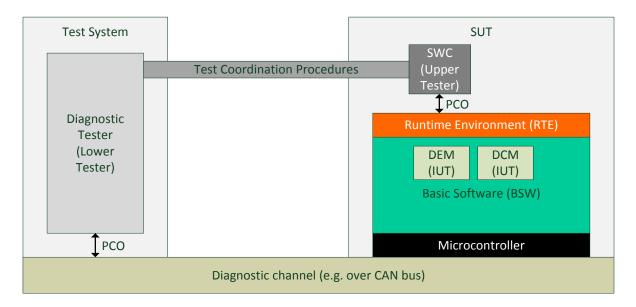
The diagnostics functionality of the AUTOSAR Basic Software implementation under test is tested by stimulating the SUT using diagnostic requests sent by the test environment (i.e. diagnostic tester) and by calling API functions that the DEM and DCM modules of the Basic Software provide to the SWCs (through the RTE).

The behavior of the diagnostic modules DEM and DCM is also observed on the diagnostic channel connecting the diagnostic tester with the SUT and at the interfaces of the SWCs which belong to the diagnostic API provided by the DEM and DCM modules.

3.1 Test System

3.1.1 Overview on Architecture

The basic test setup is depicted in the following figure:



The test cases require a SWC as Upper Tester and a diagnostic tester as Lower Tester.

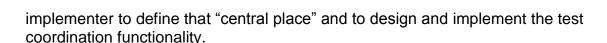
3.1.2 Specific Requirements

The SWC can call diagnostic service functions and provides callback functions to be called by the DEM and DCM modules upon reception of specific diagnostic request. In this way, this SWC observes the diagnostic functionality under test.

The diagnostic tester sends UDS requests on the diagnostic channel and receives the UDS responses of the SUT. These responses are then evaluated by the test system with respect to the expectation from the test case.

3.1.3 Test Coordination Requirements

As observation of the SUT is done by the test cases at both the Lower Tester and the Upper Tester, a test coordination procedure for collecting the local test verdicts (at LT and UT) at one central place is required. It is up to the test system designer /



3.2 Test Configuration

A proper configuration of the diagnostics functionality in the DEM and DCM modules is a mandatory prerequisite for execution of the acceptance test cases specified in this document.

Although the objective of AUTOSAR Acceptance Tests is to use ICC1 level configuration files (i.e. System Description and SWC Description) for BSW configuration, the diagnostic test cases contain the configuration requirements on EcuC (ICC3) level.

This is due to the current practice of using EcuC parameters for configuring the diagnostic modules. Furthermore, a future diagnostic format on ICC1 level (planned for AUTOSAR R4.2.1) is expected to better fit the needs of diagnostic development than the current SWC ServiceNeeds on diagnostics. Therefore, the configuration requirements of the acceptance tests on diagnostic services shall later be transformed to this new format.

3.2.1 Required ECU Extract of System Description Files

An ECU Extract with the definitions for the diagnostic connection (e.g. using CAN TP) between SUT and Lower Tester (external diagnostic tester) is usually needed to configure the SUT. However, it is the user's responsibility to create this ECU Extract.

3.2.2 Required ECU Configuration Description Files

The test cases require an ECU Configuration Value Description with definition of configuration parameters for the DEM and DCM modules. This description file (or alternatively the matching configuration with the configuration tooling) needs to be created by the user from the configuration requirements of the test cases.

3.2.3 Required Software Component Description Files

For the test cases on diagnostic functionality, the required definitions in the SWC Description (e.g. PortInterfaces) can be directly derived from the configuration requirements and interactions between the SWC and the RTE specified in the test cases. The SWC Description needs to be created by the user based on these requirements.

3.2.4 Mandatory vs. Customizable Parts

The configuration requirements on the "DCM Record Data Identifiers" as stated in the test cases are mandatory to be applied since they define specific DCM functionality that is in the focus of the tests.

The configuration parts in the ECU Configuration Value Description related to communication between ECU and diagnostic tester need to be customized by the test implementer according to user or system specific requirements.

3.3 Test Case Design

3.3.1 DEM related test cases

The test cases on the service interface "Dem_DiagnosticMonitor" are based on use cases containing different situations of reporting an event by an application SWC to the DEM.

For each test case, a SWC is defined and needs to be implemented. This SWC executes the test steps defined in the test case which stimulate and observe the DEM through the SWC's ports to the DEM services.

The functionality of retrieving events, getting the "fault detection counter", getting the associated DTC and resetting the event is tested in each of the above test cases using re-usable test steps (see Ch. 4).

Enable and storage conditions are not tested here because the required functionality is located in a different service interface.

3.3.2 DCM related test cases

The test cases on the service interface "DataServices" (in R3.x: "Dcm_DidServices") are based on the use cases made up by the related UDS services (ReadDataByldentifier, WriteDataByldentifier, ReadScalingDataByldentifier, InputOutputControlByldentifier).

However, the test cases focus on the interaction between the SWC and the DCM module (through the RTE). Therefore, DCM functionality that is internal and has no interference with SWCs (e.g. checking for validity of a request with respect to current diagnostic session or current security level) is not in scope of the tests.

The design approach for the test cases is to stimulate the SUT using UDS requests from the test environment and to observe the SUT's behavior at the RTE interface of the DCM module and its UDS responses.

4 Re-usable Test Steps

Test steps that are re-used by multiple test cases are defined as "Test Step Groups". They are defined together with the test cases in Chapter 5:

- Test Step Group: Reset event
- Test Step Group: Check event reset
- Test Step Group: Check event passed after failed
- Test Step Group: Check failed event

5 Test Cases

5.1 Test Cases on DiagnosticMonitor (Dem)

5.1.1 [ATS_DIAG_00032] Test Step Group: Reset event

Test Objective	Test Step Group: Reset event		
ID		AUTOSAR Releases	
Affected Modules		State	reviewed
Trace to Requirement on Acceptance Test Document			
Trace to SWS Item	DiagnosticEventManager: SWS_EDiagnosticEventManager: SWS_EDIAGNOSTICEVENTM	Dem_00187 Dem_00051 Dem_00333 Dem_00052 Dem_00053 Dem_00379 Dem_00204 Dem_00185 Dem_00195 Dem_00196 Dem_00197	
Requirements / Reference to Test Environment			
Configuration Parameters			
Summary	This test step group is used by the	e test cases.	
Needed Adaptation to other Releases			
Pre-conditions	1. Input parameter "eventID" has a	a value	
Main Test Execu	ıtion		
Test Steps			Pass Criteria
Step 1	[LT <diagnostic>] Reset all DemEvents with associa using UDS request "ClearDiagnosticInformation" (0x1</diagnostic>	ted DTC	SUT sends positive response (0x54)
Step 2	[SWC] Get status of eventID using Dem_GetEventStatus() (SWS_De (SWS_Dem_00051)	em_00195)	E_OK is returned
Step 3	[SWC] Check that the status bits are as e (SWS_Dem_00036) (SWS_Dem_	expected _00379)	Bit 0 (TestFailed): 0 Bit 1 (TestFailedThisOperationCycle): 0 Bit 2 (PendingDTC): 0

		Bit 3 (ConfirmedDTC): 0
Step 4	[SWC] Get FAILED status of eventID using Dem_GetEventFailed() (SWS_Dem_00333) (SWS_Dem_00196) (SWS_Dem_00052)	E_OK is returned
Step 5	[SWC] Check the FAILED status	FAILED status is FALSE (SWS_Dem_00187)
Step 6	[SWC] Get TESTED status of eventID using Dem_GetEventTested() (SWS_Dem_00333) (SWS_Dem_00197) (SWS_Dem_00053)	E_OK is returned
Step 7	[SWC] Check the TESTED status	TESTED status is FALSE
Step 8	[SWC] Get fault detection counter of eventID using Dem_GetFaultDetectionCounter() (SWS_Dem_00203) (SWS_Dem_00204)	E_OK is returned
Step 9	[SWC] Check the fault detection counter	Fault detection counter is 0 (SWS_Dem_00343)
Post- conditions		

5.1.2 [ATS_DIAG_00033] Test Step Group: Check event reset

Test Objective	Test Step Group: Check event re	eset	
ID		AUTOSAR Releases	
Affected Modules		State	reviewed
Trace to Requirement on Acceptance Test Document			
Item	DiagnosticEventManager: SWS_DiagnosticEventManager: SWS_Di	Dem_0005 Dem_0003 Dem_0037 Dem_0020 Dem_0019 Dem_0019	1 6 9 4 5 8
Requirements / Reference to Test Environment	J J J		
Configuration Parameters Summary	This test step group is used by the	ne test case	c
Needed Adaptation to	Triis test step group is used by the	ie iesi case	ა.

other Releases		
	Input parameter "eventID" has a value	
Main Test Exec		
Test Steps		Pass Criteria
Step 1	[SWC] Get DTC associated with eventID using Dem_GetDTCOfEvent() (SWS_Dem_00198) (SWS_Dem_00269)	E_OK is returned
Step 2	[SWC] Check the DTC value is equal to the configured one	DTC value is equal to the configured one
Step 3	[SWC] Get status of eventID using Dem_GetEventStatus() (SWS_Dem_00195) (SWS_Dem_00051)	E_OK is returned
Step 4	[SWC] Check that the status bits are as expected (SWS_Dem_00036) (SWS_Dem_00379)	Bit 0 (TestFailed): 0 Bit 1 (TestFailedThisOperationCycle): 0 Bit 2 (PendingDTC): 0 Bit 3 (ConfirmedDTC): 0 Bit 4 (TestNotCompletedSinceLastClear): don't care Bit 5 (TestFailedSinceLastClear): 0 Bit 6 (TestNotCompletedThisOperationCycle): don't care Bit 7 (WarningIndicatorRequested): 0
Step 5	[SWC] Get fault detection counter of eventID using Dem_GetFaultDetectionCounter() (SWS_Dem_00203) (SWS_Dem_00204)	E_OK is returned
Step 6	[SWC] Check the fault detection counter	Fault detection counter is 0 (SWS_Dem_00343)
Post- conditions		

5.1.3 [ATS_DIAG_00034] Test Step Group: Check event passed after failed

Test Objective	Test Step Group: Check event passed after failed		
ID		AUTOSAR Releases	
Affected Modules		State	reviewed
Trace to Requirement on Acceptance			

	T	
Test Document		
Trace to SWS Item	DiagnosticEventManager: SWS_Dem_0026 DiagnosticEventManager: SWS_Dem_0008 DiagnosticEventManager: SWS_Dem_0033	51 33
	DiagnosticEventManager: SWS_Dem_0005 DiagnosticEventManager: SWS_Dem_0005 DiagnosticEventManager: SWS_Dem_0005	53 36
	DiagnosticEventManager: SWS_Dem_0037 DiagnosticEventManager: SWS_Dem_0020 DiagnosticEventManager: SWS_Dem_0019	04 95
	DiagnosticEventManager: SWS_Dem_0019 DiagnosticEventManager: SWS_Dem_0019 DiagnosticEventManager: SWS_Dem_0019	97 98
Requirements / Reference	DiagnosticEventManager: SWS_Dem_0020	J3
to Test Environment		
Configuration Parameters		
Summary	This test step group is used by the test case	es.
Needed Adaptation to other Releases		
	Input parameter "eventID" has a value	
Main Test Exec		
Test Steps		Pass Criteria
	[SWC] Get DTC associated with eventID using Dem_GetDTCOfEvent()	Pass Criteria E_OK is returned
Test Steps	[SWC] Get DTC associated with eventID using	
Test Steps Step 1	[SWC] Get DTC associated with eventID using Dem_GetDTCOfEvent() (SWS_Dem_00198) (SWS_Dem_00269) [SWC] Check the DTC value is equal to the	E_OK is returned DTC value is equal to the configured
Test Steps Step 1 Step 2	[SWC] Get DTC associated with eventID using Dem_GetDTCOfEvent() (SWS_Dem_00198) (SWS_Dem_00269) [SWC] Check the DTC value is equal to the configured one [SWC] Get status of eventID using Dem_GetEventStatus()	E_OK is returned DTC value is equal to the configured one E_OK is returned Bit 0 (TestFailed): 0 Bit 1 (TestFailedThisOperationCycle): 1
Test Steps Step 1 Step 2 Step 3	[SWC] Get DTC associated with eventID using Dem_GetDTCOfEvent() (SWS_Dem_00198) (SWS_Dem_00269) [SWC] Check the DTC value is equal to the configured one [SWC] Get status of eventID using Dem_GetEventStatus() (SWS_Dem_00195) (SWS_Dem_00051) [SWC] Check that the status bits are as expected	E_OK is returned DTC value is equal to the configured one E_OK is returned Bit 0 (TestFailed): 0 Bit 1 (TestFailedThisOperationCycle):
Test Steps Step 1 Step 2 Step 3	[SWC] Get DTC associated with eventID using Dem_GetDTCOfEvent() (SWS_Dem_00198) (SWS_Dem_00269) [SWC] Check the DTC value is equal to the configured one [SWC] Get status of eventID using Dem_GetEventStatus() (SWS_Dem_00195) (SWS_Dem_00051) [SWC] Check that the status bits are as expected	E_OK is returned DTC value is equal to the configured one E_OK is returned Bit 0 (TestFailed): 0 Bit 1 (TestFailedThisOperationCycle): 1 Bit 2 (PendingDTC): 1 Bit 3 (ConfirmedDTC): 1 Bit 4 (TestNotCompletedSinceLastClear): 0
Test Steps Step 1 Step 2 Step 3	[SWC] Get DTC associated with eventID using Dem_GetDTCOfEvent() (SWS_Dem_00198) (SWS_Dem_00269) [SWC] Check the DTC value is equal to the configured one [SWC] Get status of eventID using Dem_GetEventStatus() (SWS_Dem_00195) (SWS_Dem_00051) [SWC] Check that the status bits are as expected	E_OK is returned DTC value is equal to the configured one E_OK is returned Bit 0 (TestFailed): 0 Bit 1 (TestFailedThisOperationCycle): 1 Bit 2 (PendingDTC): 1 Bit 3 (ConfirmedDTC): 1 Bit 4 (TestNotCompletedSinceLastClear):

	T	
		0
Step 5	[SWC] Get FAILED status of eventID using Dem_GetEventFailed() (SWS_Dem_00333) (SWS_Dem_00196) (SWS_Dem_00052)	E_OK is returned
Step 6	[SWC] Check the FAILED status	FAILED status is FALSE (SWS_Dem_00187)
Step 7	[SWC] Get TESTED status of eventID using Dem_GetEventTested() (SWS_Dem_00333) (SWS_Dem_00197) (SWS_Dem_00053)	E_OK is returned
Step 8	[SWC] Check the TESTED status	TESTED status is TRUE
Step 9	[SWC] Get fault detection counter of eventID using Dem_GetFaultDetectionCounter() (SWS_Dem_00203) (SWS_Dem_00204)	E_OK is returned
Post- conditions		

5.1.4 [ATS_DIAG_00035] Test Step Group: Check failed event

Test Objective	Test Step Group: Check failed event		
ID	ATS_DIAG_00035	AUTOSAR Releases	
Affected Modules		State	reviewed
Trace to Requirement on Acceptance Test Document			
Trace to SWS Item	DiagnosticEventManager: SWS_DiagnosticEventManager: SWS_Di	Dem_0033 Dem_0005 Dem_0005 Dem_00037 Dem_0033 Dem_0019 Dem_0019 Dem_0019	3 2 3 6 9 8 4 5 6
Requirements / Reference to Test Environment			
Configuration Parameters			
Summary	This test step group is used by the	ne test case	S.
Needed			
1E of E2			Document ID 627: ALITOSAB ATS Diagnostic Carries

Adaptation to other		
Releases		
	Input parameter "eventID" has a value	
Main Test Exec	ution	
Test Steps		Pass Criteria
Step 1	[SWC] Get status of eventID using Dem_GetEventStatus() (SWS_Dem_00195) (SWS_Dem_00051)	E_OK is returned
Step 2	[SWC] Check that the status bits are as expected (SWS_Dem_00036) (SWS_Dem_00379)	Bit 0 (TestFailed): 1 Bit 1 (TestFailedThisOperationCycle): 1 Bit 2 (PendingDTC): 1 Bit 3 (ConfirmedDTC): 1 Bit 4 (TestNotCompletedSinceLastClear): 0 Bit 5 (TestFailedSinceLastClear): 1 Bit 6 (TestNotCompletedThisOperationCycle): 0 Bit 7 (WarningIndicatorRequested): 1: if indicator is configured for eventID 0: else
Step 3	[SWC] Get FAILED status of eventID using Dem_GetEventFailed() (SWS_Dem_00333) (SWS_Dem_00196) (SWS_Dem_00052)	E_OK is returned
Step 4	[SWC] Check the FAILED status	FAILED status is TRUE
Step 5	[SWC] Get TESTED status of eventID using Dem_GetEventTested() (SWS_Dem_00333) (SWS_Dem_00197) (SWS_Dem_00053)	E_OK is returned
Step 6	[SWC] Check the TESTED status	TESTED status is TRUE
Post- conditions		

5.1.5 [ATS_DIAG_00077] Reporting of an event without FreezeFrame and without debouncing

F				
Test Objective	Reporting of an event without FreezeFrame and without debouncing			
ID		AUTOSAR Releases	4.0.3 4.1.1 4.2.1 4.2.2	
Affected Modules	Dem	State	reviewed	

	I		
Trace to	ATR: ATR_ATR_00018		
Requirement on Acceptance			
Test Document			
Trace to SWS	DiagnosticEventManage	" SWS Dem 00330	1
Item	DiagnosticEventManagei		
Requirements /	[SWC] A tester SWC that	t stimulates and obse	erves the Dem API through the RTE.
Reference			C
to Test			
Environment	A Dana avant FO4 with D	DTC 0-444444	
Configuration Parameters	properties:	emD1C = 0x111111	must be configured with the following
i arameters	- No associated FreezeF	rame	
	- No debouncing by Dem	(i.e. definition of <d< th=""><th>emDebounceMonitorInternal>)</th></d<>	emDebounceMonitorInternal>)
	- DemAgingAllowed = FA	LSE	
_	 No DemEventFailureCy 		
Summary	Ensure that a reported D (i.e. immediately qualified		eezeFrame and without debouncing
			is not stored. Then, report event E01
	and ensure that the DTC		is not stored. Then, report event 201
Needed	Needed Adaptation for	Release 3.2.2	
Adaptation to			
other Releases	Configuration: [low]	Some configuration	parameters have different names in
			g.g DemHealingAllowed instead of
	Test Steps: [none]	DemAgingAllowed).	•
	Needed Adaptation for	Release 4.2.1 or 4.2	2.2
	Configuration: [none]		
	Configuration: [none]	not affected by R4.2	2.1 and R4.2.2
	Test Steps: [none]	not ancolou by 14.2	2.1 dild 1(4.2.2
	r det eteper [meme]	1	
Pre-conditions	1. Event E01 is not FAILE	ED .	
	2. Operation cycle for even	ent E01 has started	
Main Test Exec	ution		
Test Steps			Pass Criteria
Step 1	[SWC]		All test steps passed
	Execute "check event res	set" group for event	
01	E01		F 0//:
Step 2	[SWC] Invoke Dem SetEventSt:	atus(F01 FAII FD)	E_OK is returned
Step 3	Invoke Dem_SetEventStatus(E01, FAILED) [SWC]		
J. C. P. C.	Delay for 1 sec		
Step 4	[SWC]		All test steps passed
	Execute "check failed eve	ent" group for event] ' '
	E01		
Step 5	[SWC]	(= 2.1	All test steps passed
_	Execute "reset event" gro	oup for event E01	
Post-	1. Event E01 is reset		
conditions			

5.1.6 [ATS_DIAG_00078] Reporting of an event with counter-based debouncing and jump after event status change

Test Objective	Reporting of an event with counter-based debouncing and jump after event status change		
ID	ATS_DIAG_00078	AUTOSAR Releases	4.0.3 4.1.1 4.2.1 4.2.2
Affected Modules	Dem	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00018	·	
Trace to SWS Item	DiagnosticEventManager: SWS_Dem_00330 DiagnosticEventManager: SWS_Dem_00413 DiagnosticEventManager: SWS_Dem_00526 DiagnosticEventManager: SWS_Dem_00414 DiagnosticEventManager: SWS_Dem_00415 DiagnosticEventManager: SWS_Dem_00416 DiagnosticEventManager: SWS_Dem_00417 DiagnosticEventManager: SWS_Dem_00418 DiagnosticEventManager: SWS_Dem_00419 DiagnosticEventManager: SWS_Dem_00422 DiagnosticEventManager: SWS_Dem_00423 DiagnosticEventManager: SWS_Dem_00424 DiagnosticEventManager: SWS_Dem_00425		
Requirements / Reference to Test Environment	[SWC] A tester SWC that	t stimulates and obse	erves the Dem API through the RTE.
Configuration Parameters	A Dem event E02 must be configured with DemDTC = 0x222222 and the following properties: - DemDebounceCounterBased - DemDebounceCounterIncrementStepSize = 7 - DemDebounceCounterFailedThreshold = 127 - DemDebounceCounterDecrementStepSize = 17 - DemDebounceCounterPassedThreshold = -128 - DemDebounceCounterJumpUp = TRUE - DemDebounceCounterJumpUpValue = 0 - DemDebounceCounterJumpDown = TRUE - DemDebounceCounterJumpDown > TRUE - DemDebounceCounterJumpDownValue = 0 - DemAgingAllowed = FALSE - No DemEventFailureCycleCounterThreshold defined		
Summary	Ensure that a reported DEM event is debounced (counter-based and with jump after event status change) and stored correctly: PREFAILED and PREPASSED events are reported repeatedly while the event status is checked. After the right number of PREPASS or PREFAILED reports, the event status is expected to change to FAILED or PASSED, respectively.		
Needed Adaptation to	Needed Adaptation for Release 3.2.2		
other Releases	Configuration: [low] Test Steps: [none]		parameters have different names in .g DemHealingAllowed instead of
	Needed Adaptation for Release 4.1.1 and 4.2.1		

	Configuration: [none]		
		not affected by R4.2	2.1 and R4.2.2
	Test Steps: [none]		
Dro conditions	1	-D	
	 Event E02 is not FAILE Operation cycle for ever 		
Main Test Execu		THE LOZ HUS STUTTED	
Test Steps	AUOII		Pass Criteria
Step 1	[SWC]		All test steps passed
Olep I	Execute "check event res	et" group for event	All test steps passed
	E02		
Step 2	Debouncing of PREF	AILED event	
	[SWC]	001	
	Execute the following LO DemDebounceCounterIng		
	1 = 18 times.	orementotepoize) -	
Step 3	Start of LOOP		E OK is returned
	[SWC]		
	Invoke Dem_SetEventSta		
o	DEM_EVENT_STATUS_	PKEFAILED)	- O(C)
Step 4	[SWC]	2 ueina	E_OK is returned
	Get FAILED status of E02 Dem_GetEventFailed()	ı uəlliy	
Step 5	[SWC]		FAILED is FALSE
	Check FAILED status		5-
Step 6	[SWC]		E_OK is returned
	Invoke Dem_GetFaultDet		
01 =	E02 to retrieve FDCounte	er	EDO: .tt
Step 7	[SWC]	(i±1) *	FDCounter has the expected value
	Check that FDCounter is DemDebounceCounterInd		
	(with i being the number of		
	counting from 0)		
	End of LOOP		
Step 8	[SWC]	atuo/E02	E_OK is returned
	Invoke Dem_SetEventSta DEM_EVENT_STATUS_		
Step 9	[SWC]		
	Delay for 1 sec		
Step 10	[SWC]		E_OK is returned
	Invoke Dem_GetFaultDet		
	E02 to retrieve FDCounte	er	
Step 11	[SWC]	107	FDCounter has the expected value
Stop 12	Check that FDCounter is		All toot atons possed
Step 12	[SWC] Execute "check failed eve		All test steps passed
	event E02	7.10 toot stops tot	
Step 13	Debouncing of PREP	ASSED event	
	[SWC]		

	Execute the following LOOP for ceil(128 DIV DemDebounceCounterDecrementStepSize) - 1 = 7 times	
Step 14	Start of LOOP	E_OK is returned
	[SWC] Invoke Dem_SetEventStatus(E02, DEM_EVENT_STATUS_PREPASSED)	
Step 15	[SWC] Get FAILED status of E02 using Dem_GetEventFailed()	E_OK is returned
Step 16	[SWC] Check FAILED status	FAILED is TRUE
Step 17	[SWC] Invoke Dem_GetFaultDetectionCounter() on E02 to retrieve FDCounter	E_OK is returned
Step 18	[SWC] Check that FDCounter is -(i+1) * DemDebounceCounterDecrementStepSize (with i being the number of repetitions counting from 0) End of LOOP	FDCounter has the expected value
Step 19	[SWC] Invoke Dem_SetEventStatus(E02, DEM_EVENT_STATUS_PREPASSED)	E_OK is returned
Step 20	[SWC] Delay for 1 sec	
Step 21	[SWC] Invoke Dem_GetFaultDetectionCounter() on E02 to retrieve FDCounter	E_OK is returned
Step 22	[SWC] Check that FDCounter is -128	FDCounter has the expected value
Step 23	[SWC] Execute "check event passed after failed" test steps for event E02	All test steps passed
Step 24	Debouncing of PREFAILED event [SWC] Execute the following LOOP for ceil(127 DIV DemDebounceCounterIncrementStepSize) 1 = 18 times.	
Step 25	Start of LOOP [SWC] Invoke Dem_SetEventStatus(E02, DEM_EVENT_STATUS_PREFAILED)	E_OK is returned
Step 26	[SWC] Get FAILED status of E02 using Dem_GetEventFailed()	E_OK is returned
Step 27	[SWC] Check FAILED status	FAILED is FALSE
Step 28	[SWC] Invoke Dem_GetFaultDetectionCounter() on E02 to retrieve FDCounter	E_OK is returned

Step 29	[SWC] Check that FDCounter is (i+1) * DemDebounceCounterIncrementStepSize (with i being the number of repetitions counting from 0) End of LOOP	FDCounter has the expected value
Step 30	[SWC] Invoke Dem_SetEventStatus(E02, DEM_EVENT_STATUS_PREFAILED)	E_OK is returned
Step 31	[SWC] Delay for 1 sec	
Step 32	[SWC] Invoke Dem_GetFaultDetectionCounter() on E02 to retrieve FDCounter	E_OK is returned
Step 33	[SWC] Check that FDCounter is 127	FDCounter has the expected value
Step 34	[SWC] Execute "check failed event" test steps for event E02	All test steps passed
Step 35	[SWC] Execute "reset event" test steps for event E02	All test steps passed
Post- conditions	1. Event E02 is reset	

5.1.7 [ATS_DIAG_00085] Reporting of an event with counter-based debouncing and without jump

			•
Test Objective	Reporting of an event with counter-based debouncing and without jump		
ID	ATS_DIAG_00085	AUTOSAR Releases	4.0.3 4.1.1 4.2.1 4.2.2
Affected Modules	Dem	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00018		
ltem	DiagnosticEventManager: SWS_DiagnosticEventManager: SWS_Di	Dem_00413 Dem_00526 Dem_00414 Dem_00415 Dem_00416 Dem_00417 Dem_00418 Dem_00419 Dem_00422	
Requirements / Reference to Test Environment	[SWC] A tester SWC that stimula	tes and obse	erves the Dem API through the RTE

Canfiguration	A Dama assent E00 with D	DTO 0000000	and the configuration of the the fellowing
Configuration Parameters		emDTC = 0x3333333	must be configured with the following
Parameters	properties: - DemDebounceCounterl	Rasad	
	- DemDebounceCounterBased - DemDebounceCounterIncrementStepSize = 7		
	- DemDebounceCounterFailedThreshold = 127		
	- DemDebounceCounterDecrementStepSize = 17		
	- DemDebounceCounterl		
	- DemDebounceCounter		
	DemAgingAllowed = FANo DemEventFailureCy		d defined
0			
Summary		EM event is debound	ced (counter-based without jump) and
	stored correctly:	SSED avanta ara ra	ported repeatedly while the event
			ported repeatedly while the event REPASS or PREFAILED reports, the
			or PASSED, respectively.
Needed	Needed Adaptation for		o or i respectively.
Adaptation to	Inceded Adaptation for	NCICASE 3.2.2	
other Releases			
The Releases	Configuration: [low]		parameters have different names in
	L		e.g DemHealingAllowed instead of
	Test Steps: [none]	DemAgingAllowed).	
	Needed Adaptation for	Release 4.1.1 and 4	l.2.1
	Configuration: [none]		
		not affected by R4.2	2.1 and R4.2.2
	Test Steps: [none]		
Pre-conditions	1. Event E03 is not FAILE	ED	
	2. Operation cycle for eve		
Main Test Exec	ution		
Test Steps			Pass Criteria
Step 1	[SWC]		All test steps passed
Step 1	Execute "check event res		All lest steps passed
	E03	set group for event	
Step 2	Debouncing of PREF	All FD event	1
Step 2	Debounding of FIXE	AILLD EVEIR	
	[SWC]		
	Execute the following LO	OP for cail/127 DIV	
	DemDebounceCounterIn		
	1 = 18 times.	.c. 5///0///0/0/p0/20) =	
Step 3	Start of LOOP		E_OK is returned
otep 3	July of Loop		
	[SWC]		
	Invoke Dem_SetEventSt	atus(F03	
	DEM_EVENT_STATUS_		
Step 4	[SWC]		E_OK is returned
Cicp 4	Get FAILED status of E0	3 using	L_OR IS TERUTION
	Dem_GetEventFailed()	o aonig	
Step 5	[SWC]		FAILED is FALSE
otep 5	Check FAILED status		I ALLED IS I ALOL
Stop 6			E OK is returned
Step 6	[SWC]		E_OK is returned

	Invoke Dem_GetFaultDetectionCounter() on E03 to retrieve FDCounter	
Step 7	[SWC] Check that FDCounter is (i+1) * DemDebounceCounterIncrementStepSize (with i being the number of repetitions counting from 0)	FDCounter has the expected value
	End of LOOP	
Step 8	[SWC] Invoke Dem_SetEventStatus(E03, DEM_EVENT_STATUS_PREFAILED)	E_OK is returned
Step 9	[SWC] Delay for 1 sec	
Step 10	[SWC] Invoke Dem_GetFaultDetectionCounter() on E03 to retrieve FDCounter	E_OK is returned
Step 11	[SWC] Check that FDCounter is 127	FDCounter has the expected value
Step 12	[SWC] Execute "check failed event" test steps for event E03	All test steps passed
Step 13	Debouncing of PREPASSED event [SWC] Execute the following LOOP for ceil(255 DIV DemDebounceCounterDecrementStepSize) 1 = 14 times	
Step 14	Start of LOOP [SWC] Invoke Dem_SetEventStatus(E03, DEM_EVENT_STATUS_PREPASSED)	E_OK is returned
Step 15	[SWC] Get FAILED status of E03 using Dem_GetEventFailed()	E_OK is returned
Step 16	[SWC] Check FAILED status	FAILED is TRUE
Step 17	[SWC] Invoke Dem_GetFaultDetectionCounter() on E03 to retrieve FDCounter	E_OK is returned
Step 18	[SWC] Check that FDCounter is 127 - (i+1) * DemDebounceCounterDecrementStepSize (with i being the number of repetitions counting from 0)	FDCounter has the expected value
Step 19	End of LOOP [SWC]	E_OK is returned
oteh 13	Invoke Dem_SetEventStatus(E03, DEM_EVENT_STATUS_PREPASSED)	
Step 20	[SWC] Delay for 1 sec	
Step 21	[SWC] Invoke Dem_GetFaultDetectionCounter() on	E_OK is returned

	E03 to retrieve FDCounter	
Step 22	[SWC]	FDCounter has the expected value
Step 22	Check that FDCounter is -128	i boodilier has the expected value
Step 23	[SWC] Execute "check event passed after failed" test steps for event E03	All test steps passed
Step 24	Debouncing of PREFAILED event	
	[SWC] Execute the following LOOP for ceil(255 DIV DemDebounceCounterIncrementStepSize) - 1 = 36 times	
Step 25	Start of LOOP	E_OK is returned
	[SWC] Invoke Dem_SetEventStatus(E03, DEM_EVENT_STATUS_PREFAILED)	
Step 26	[SWC] Get FAILED status of E03 using Dem_GetEventFailed()	E_OK is returned
Step 27	[SWC] Check FAILED status	FAILED is FALSE
Step 28	[SWC] Invoke Dem_GetFaultDetectionCounter() on E03 to retrieve FDCounter	E_OK is returned
Step 29	[SWC] Check that FDCounter is -128 + (i+1) * DemDebounceCounterIncrementStepSize (with i being the number of repetitions counting from 0) End of LOOP	FDCounter has the expected value
Step 30	[SWC] Invoke Dem_SetEventStatus(E03, DEM_EVENT_STATUS_PREFAILED)	E_OK is returned
Step 31	[SWC] Delay for 1 sec	
Step 32	[SWC] Invoke Dem_GetFaultDetectionCounter() on E03 to retrieve FDCounter	E_OK is returned
Step 33	[SWC] Check that FDCounter is 127	FDCounter has the expected value
Step 34	[SWC] Execute "check failed event" test steps for event E03	All test steps passed
Step 35	[SWC] Execute "reset event" test steps for event E03	All test steps passed
Post- conditions	1. Event E03 is reset	

5.1.8 [ATS_DIAG_00245] Reporting of an event with time-based debouncing

	Reporting of an event with			
ID	ATS_DIAG_00245	AUTOSAR Releases	4.0.3 4.1.1 4.2.1 4.2.2	
Affected Modules	Dem	State	reviewed	
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00018			
Trace to SWS Item	DiagnosticEventManager: SWS_Dem_00330 DiagnosticEventManager: SWS_Dem_00413 DiagnosticEventManager: SWS_Dem_00527 DiagnosticEventManager: SWS_Dem_00426 DiagnosticEventManager: SWS_Dem_00427 DiagnosticEventManager: SWS_Dem_00428 DiagnosticEventManager: SWS_Dem_00430 DiagnosticEventManager: SWS_Dem_00432 DiagnosticEventManager: SWS_Dem_00434 DiagnosticEventManager: SWS_Dem_00183			
Requirements / Reference to Test Environment				
Configuration Parameters	A Dem event E04 with DemDTC = 0x444444 must be configured with the following properties: - DemDebounceTimeBase - DemDebounceTimeFailedThreshold = 6000 milliseconds - DemDebounceTimePassedThreshold = 4000 milliseconds - DemAgingAllowed = FALSE - No DemEventFailureCycleCounterThreshold defined			
Summary	Report PREFAILED event and ensure that the event is not FAILED until 6 sec (DemTimeFailedThreshold) later and then set to FAILED. Afterwards, report PREPASSED event and ensure that the event is still FAILED until 4 sec (DemTimePassedThreshold) later and then set to PASSED.			
Adaptation to	Needed Adaptation for Release 3.2.2			
other Releases	Configuration: [low] Test Steps: [none]		n parameters have different names in e.g DemHealingAllowed instead of).	
	Needed Adaptation for Release 4.1.1 and 4.2.1			
	Configuration: [none] Test Steps: [none]	not affected by R4.	2.1 and R4.2.2	
Pre-conditions	Event E04 is not FAILE Operation cycle for events		·	
Main Test Execu	ıtion			
Test Steps			Pass Criteria	
Step 1	[SWC] Execute "check event res	et" group for event	All test steps passed	

	E04	
Step 2	Debouncing of PREFAILED event	E_OK is returned
	[SWC] Invoke Dem_SetEventStatus(E04, DEM_EVENT_STATUS_PREFAILED)	
Step 3	[SWC] Get FAILED status of E04 using Dem_GetEventFailed() for multiple times during the next DemTimeFailedThreshold = 6 seconds and check it	FAILED status is always FALSE
Step 4	[SWC] Get FAILED status of E04 using Dem_GetEventFailed()	E_OK is returned
Step 5	[SWC] Check FAILED status	FAILED status is TRUE
Step 6	[SWC] Execute "check failed event" test steps for event E04	All test steps passed
Step 7	Debouncing of PREPASSED event [SWC] Invoke Dem_SetEventStatus(E04, DEM_EVENT_STATUS_PREPASSED)	E_OK is returned
Step 8	[SWC] Invoke Dem_GetEventFailed() on E04	E_OK is returned
Step 9	[SWC] Get FAILED status of E04 using Dem_GetEventFailed() for multiple times during the next DemTimePassedThreshold = 4 seconds and check it	FAILED status is always TRUE
Step 10	[SWC] Get FAILED status of E04 using Dem_GetEventFailed()	E_OK is returned
Step 11	[SWC] Check FAILED status	FAILED status is FALSE
Step 12	[SWC] Execute "check event passed after failed" test steps for event E04	All test steps passed
Step 13	[SWC] Execute "reset event" test steps for event E04	All test steps passed
Post- conditions	1. Event E04 is reset.	

5.1.9 [ATS_DIAG_00246] Reporting of an event with FreezeFrame and without pre-storage

Test Objective	Reporting of an event with FreezeFrame and without pre-storage		
ID	ATS_DIAG_00246 AUTOSAR 4.0.3 4.1.1 4.2.1 4.2.2 Releases		
Affected Modules	Dem	State	reviewed

	ATR: ATR_ATR_00018		
Requirement			
on Acceptance Test Document			
Trace to SWS	DiagnosticEventManage		
Item	DiagnosticEventManager: SWS_Dem_00337		
D	DiagnosticEventManager: SWS_Dem_00183		
Requirements /	[SWC] A tester SWC that stimulates and observes the Dem API through the RTE.		
to Test			
Environment			
Configuration	A Dom ayant EOE with D	emDTC = 0x555555 must be configured with the following	
_	properties:	embro = 0x333333 mast be configured with the following	
i didilictors	- DemDebounceMonitorInternal		
	- Dembebouncewonttorinternal - FreezeFrame contains variable FreezeFrameCounter (uint32)		
	 DemEnvironmentDataC 		
	DEM_CAPTURE_SYNC	HRONOUS_TO_REPORTING	
	 DemAgingAllowed = FA 		
	 No DemEventFailureCy 	cleCounterThreshold defined	
	The CMC mandalas the F	rearrations data through a DID want to the DEM The	
		reezeFrame data through a DID port to the DEM. The	
		rameCounter is incremented by 1 (with wrap-around at each periodic invocation of the SWC's main runnable	
		e initial value of the FreezeFrameCounter is arbitrary.	
Summary		d memorize the current FreezeFrameCounter value. Then,	
Summary		illed and that the associated FreezeFrame contains the	
	FreezeFrameCounter with		
Needed			
Adaptation to			
other Releases		Some configuration parameters have different names in	
other Releases		Some configuration parameters have different names in	
other Releases		R3.2 than in R4.x (e.g DemHealingAllowed instead of	
other Releases	Configuration: [low]		
other Releases		R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed).	
other Releases	Configuration: [low] Test Steps: [low]	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function	
other Releases		R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed).	
other Releases		R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the	
other Releases		R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only	
		R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used).	
	Test Steps: [low]	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used).	
	Test Steps: [low]	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used).	
	Test Steps: [low] Needed Adaptation for	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1	
	Test Steps: [low]	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different	
	Test Steps: [low] Needed Adaptation for	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different	
	Test Steps: [low] Needed Adaptation for	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to DemEnvironmentDataCapture,	
	Test Steps: [low] Needed Adaptation for Configuration: [low]	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to DemEnvironmentDataCapture, DEM_TRIGGER_TESTFAILED renamed to	
	Test Steps: [low] Needed Adaptation for Configuration: [low]	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to DemEnvironmentDataCapture,	
	Test Steps: [low] Needed Adaptation for Configuration: [low] Test Steps: [none]	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to DemEnvironmentDataCapture, DEM_TRIGGER_TESTFAILED renamed to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING	
	Test Steps: [low] Needed Adaptation for Configuration: [low]	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to DemEnvironmentDataCapture, DEM_TRIGGER_TESTFAILED renamed to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING	
	Test Steps: [low] Needed Adaptation for Configuration: [low] Test Steps: [none] Needed Adaptation for	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to DemEnvironmentDataCapture, DEM_TRIGGER_TESTFAILED renamed to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING	
	Test Steps: [low] Needed Adaptation for Configuration: [low] Test Steps: [none]	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to DemEnvironmentDataCapture, DEM_TRIGGER_TESTFAILED renamed to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING Release 4.2.1	
	Test Steps: [low] Needed Adaptation for Configuration: [low] Test Steps: [none] Needed Adaptation for	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to DemEnvironmentDataCapture, DEM_TRIGGER_TESTFAILED renamed to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING	
	Test Steps: [low] Needed Adaptation for Configuration: [low] Test Steps: [none] Needed Adaptation for	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to DemEnvironmentDataCapture, DEM_TRIGGER_TESTFAILED renamed to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING Release 4.2.1	
	Test Steps: [low] Needed Adaptation for Configuration: [low] Test Steps: [none] Needed Adaptation for Configuration: [low]	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to DemEnvironmentDataCapture, DEM_TRIGGER_TESTFAILED renamed to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING Release 4.2.1	
	Test Steps: [low] Needed Adaptation for Configuration: [low] Test Steps: [none] Needed Adaptation for Configuration: [low]	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to DemEnvironmentDataCapture, DEM_TRIGGER_TESTFAILED renamed to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING Release 4.2.1	
	Test Steps: [low] Needed Adaptation for Configuration: [low] Test Steps: [none] Needed Adaptation for Configuration: [low]	R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to DemEnvironmentDataCapture, DEM_TRIGGER_TESTFAILED renamed to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING Release 4.2.1 not affected by R4.2.2	

	2. Operation cycle for event E05 has started.		
Main Test Execution			
Test Steps		Pass Criteria	
Step 1	[SWC] Execute "check event reset" group for event E05	All test steps passed	
Step 2	[SWC] Memorize the current value of FreezeFrameCounter in FFC		
Step 3	[SWC] Invoke Dem_SetEventStatus(E05, FAILED)	E_OK is returned	
Step 4	[SWC] Delay for 1 sec		
Step 5	[SWC] Execute "check failed event" test steps for event E05	All test steps passed	
Step 6	[LT <diagnostic>] Read out DemDTC = 0x555555 and store the associated FreezeFrame into FF1</diagnostic>	Retrieving FreezeFrame successful	
Step 7	[SWC] Check the FreezeFrameCounter of the FreezeFrame FF1	FreezeFrameCounter value is the same as FFC	
Step 8	[SWC] Invoke Dem_GetEventFreezeFrameData() to retrieve FreezeFrame of DemDTC=0x5555555 and store it into FF2	Retrieving FreezeFrame successful	
Step 9	[SWC] Check the FreezeFrameCounter of the FreezeFrame FF2	FreezeFrameCounter value is the same as FFC	
Step 10	[SWC] Execute "reset event" test steps for event E05	All test steps passed	
Post- conditions	Event E05 is reset.		

5.1.10 [ATS_DIAG_00247] Reporting of an event with pre-stored FreezeFrame

Test Objective	Reporting of an event with pre-stored FreezeFrame		
ID	ATS_DIAG_00247	AUTOSAR Releases	4.0.3 4.1.1 4.2.1 4.2.2
Affected Modules	Dem	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00018		
ltem	DiagnosticEventManager: SWS_Dem_00330 DiagnosticEventManager: SWS_Dem_00039 DiagnosticEventManager: SWS_Dem_00040 DiagnosticEventManager: SWS_Dem_00461 DiagnosticEventManager: SWS_Dem_00261 DiagnosticEventManager: SWS_Dem_00002 DiagnosticEventManager: SWS_Dem_00189		

-				
	DiagnosticEventManager: SWS_Dem_00464 DiagnosticEventManager: SWS_Dem_00191			
	DiagnosticEventManage			
	DiagnosticEventManage	er: SWS_Dem_00479		
	DiagnosticEventManager: SWS_Dem_00183			
Daminamanta /	DiagnosticEventManager: SWS_Dem_00558			
Requirements / Reference	[SWC] A tester SWC tha	at stimulates and observes the Dem API through the RTE.		
to Test				
Environment				
Configuration	A Dem event E06 with DemDTC = 0x666666 must be configured with the following			
Parameters	properties: - DemDebounceMonitorInternal			
		variable FreezeFrameCounter (uint32)		
	 DemEnvironmentData 	Capture =		
		CHRONOUS_TO_REPORTING		
	 DemAgingAllowed = F. No DemEventFailureC 	ALSE ycleCounterThreshold defined		
	140 Demieventi aliareo	yele oouther time should defined		
		FreezeFrame data through a DID port to the DEM. The		
		FrameCounter is incremented by 1 (with wrap-around at		
		each periodic invocation of the SWC's main runnable ne initial value of the FreezeFrameCounter is arbitrary.		
Summary	, ,	eezeFrameCounter value and pre-store a FreezeFrame.		
· · · · · · · · · · · · · · · · · · ·	After 2 seconds, report a			
		ent is failed and that the associated FreezeFrame contains		
		er with the memorized value.		
Needed Adaptation to	Needed Adaptation for	Release 3.2.2		
other Releases		Some configuration parameters have different names in		
		R3.2 than in R4.x (e.g DemHealingAllowed instead of		
	Configuration: [low]	DemAgingAllowed).		
	Comigaration: [low]			
	Test Steps: [low]	In R3.2, the API function		
		Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only		
		retrieving of FreezeFrames by diagnostic tester is used).		
	Needed Adaptation for	Release 4.1.1		
		Some configuration parameters and values have different		
		names in R4.1.1 than in R4.2.1		
	Configuration: [low]			
	Toot Stone: [name]	DemFreezeFrameCapture renamed to		
	Test Steps: [none]	DemEnvironmentDataCapture, DEM_TRIGGER_TESTFAILED renamed to		
		DEM_TRIGGER_TESTI ALLED Terramed to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING		
	Needed Adaptation for	Release 4.2.1		
	Configuration: [low]			
		not affected by R4.2.2		
	Test Steps: [none]	,		
Pre-conditions	1. Event E06 is not FAIL	.ED.		
		Operation cycle for event E07 has started.		
	ution			

Test Steps		Pass Criteria
Step 1	[SWC] Execute "check event reset" group for event E06	All test steps passed
Step 2	[SWC] Memorize the current value of FreezeFrameCounter in FFC	
Step 3	[SWC] Invoke Dem_PrestoreFreezeFrame(E06)	E_OK is returned
Step 4	[SWC] Delay for 2 seconds	
Step 5	[SWC] Invoke Dem_SetEventStatus(E06, FAILED)	E_OK is returned
Step 6	[SWC] Delay for 1 second	
Step 7	[SWC] Execute "check failed event" test steps for event E06	All test steps passed
Step 8	[LT <diagnostic>] Read out DemDTC = 0x666666 and the associated FreezeFrame into FF1</diagnostic>	Retrieving FreezeFrame FF1 successful
Step 9	[SWC] Check the FreezeFrameCounter of FreezeFrame FF1	FreezeFrameCounter has the same value as FFC
Step 10	[SWC] Invoke Dem_GetEventFreezeFrameData() to retrieve FreezeFrame of DemDTC=0x6666666 into FF2	Retrieving FreezeFrame FF2 successful
Step 11	[SWC] Check the FreezeFrameCounter of FreezeFrame FF2	FreezeFrameCounter has the same value as FFC
Step 12	[SWC] Execute "reset event" test steps for event E06	All test steps passed
Post- conditions	1. Event E06 is reset.	

5.1.11 [ATS_DIAG_00248] Reporting of an event after pre-storing and clearing FreezeFrame

Test Objective	Reporting of an event after pre-storing and clearing FreezeFrame			
ID	ATS_DIAG_00248			
Affected Modules	Dem	State	reviewed	
Trace to Requirement on Acceptance	ATR: ATR_ATR_00018			

Test Document			
		OIMO Dana 00000	
Trace to SWS	DiagnosticEventManag		
item	DiagnosticEventManag DiagnosticEventManag		
	DiagnosticEventManag		
	DiagnosticEventManager: SWS_Dem_00183 DiagnosticEventManager: SWS_Dem_00558		
Requirements /		at stimulates and observes the Dem API through the RTE. It	
Reference		zeFrameCounter through a DID port to the DEM. The SWC	
to Test		eezeFrameCounter by 1 (with wrap-around at maximum	
Environment		odic invocation of the SWC's cyclic runnable (periodicity << 1	
Liivii Oiliiliciit		f the FreezeFrameCounter is arbitrary.	
		Tester that is capable of requesting UDS services on the	
		UDS responses from the SUT	
Configuration		DemDTC = 0x777777 must be configured with the following	
Parameters	properties:	Define to = 0x111111 must be configured with the following	
i didiliotoro	- DemDebounceMonito	rInternal	
		s variable FreezeFrameCounter (uint32)	
	- DemEnvironmentData	` ,	
		NCHRONOUS_TO_REPORTING	
	- DemAgingAllowed = F		
	- No DemEventFailureCycleCounterThreshold defined		
	The SWC provides the FreezeFrame data through a DID port to the DEM. The		
	The evve provides the	FreezeFrame data through a DID port to the DEM. The	
	SWC's variable Freeze	FrameCounter is incremented by 1 during each invocation of	
		FrameCounter is incremented by 1 during each invocation of	
Summary	SWC's variable Freeze the SWC's main runnat	FrameCounter is incremented by 1 during each invocation of	
Summary	SWC's variable Freeze the SWC's main runnab Pre-store a FreezeFran	FrameCounter is incremented by 1 during each invocation of ble.	
Summary	SWC's variable Freeze the SWC's main runnab Pre-store a FreezeFran	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame,	
Summary	SWC's variable Freeze the SWC's main runnab Pre-store a FreezeFran report a failed event an	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame,	
Summary	SWC's variable Freeze the SWC's main runnable Pre-store a FreezeFran report a failed event and Then, check that the event and th	FrameCounter is incremented by 1 during each invocation of ble. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value.	
Summary Needed	SWC's variable Freeze the SWC's main runnable Pre-store a FreezeFran report a failed event and Then, check that the event and th	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. Tent is failed and that the associated FreezeFrame contains er with the memorized value.	
·	SWC's variable Freeze the SWC's main runnable Pre-store a FreezeFran report a failed event and Then, check that the evente FreezeFrameCount	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. Tent is failed and that the associated FreezeFrame contains er with the memorized value.	
Needed	SWC's variable Freeze the SWC's main runnable Pre-store a FreezeFran report a failed event and Then, check that the evente FreezeFrameCount	FrameCounter is incremented by 1 during each invocation of ble. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. Tent is failed and that the associated FreezeFrame contains er with the memorized value. TRelease 3.2.2	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnable Pre-store a FreezeFran report a failed event and Then, check that the evente FreezeFrameCount	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. Tent is failed and that the associated FreezeFrame contains er with the memorized value. Telease 3.2.2 Some configuration parameters have different names in	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the event extension of the FreezeFrameCount Needed Adaptation for the Swan of the FreezeFrameCount Needed Adaptation for the Swan of the	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. tent is failed and that the associated FreezeFrame contains er with the memorized value. r Release 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g DemHealingAllowed instead of	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnable Pre-store a FreezeFran report a failed event and Then, check that the evente FreezeFrameCount	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. Tent is failed and that the associated FreezeFrame contains er with the memorized value. Telease 3.2.2 Some configuration parameters have different names in	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the event are the FreezeFrameCount Needed Adaptation for Configuration: [low]	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. nent is failed and that the associated FreezeFrame contains er with the memorized value. r Release 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed).	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the event extension of the FreezeFrameCount Needed Adaptation for the Swan of the FreezeFrameCount Needed Adaptation for the Swan of the	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. nent is failed and that the associated FreezeFrame contains er with the memorized value. r Release 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the event are the FreezeFrameCount Needed Adaptation for Configuration: [low]	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, dememorize the current FreezeFrameCounter value. The tilde and that the associated FreezeFrame contains the erwith the memorized value. The Release 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g., DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the event are the FreezeFrameCount Needed Adaptation for Configuration: [low]	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. rent is failed and that the associated FreezeFrame contains er with the memorized value. r Release 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the event are the FreezeFrameCount Needed Adaptation for Configuration: [low]	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, demenorize the current FreezeFrameCounter value. The tilde and that the associated FreezeFrame contains the erwith the memorized value. The Release 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g., DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the eventhe FreezeFrameCount Needed Adaptation for Configuration: [low] Test Steps: [low]	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. ent is failed and that the associated FreezeFrame contains er with the memorized value. r Release 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used).	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the event are the FreezeFrameCount Needed Adaptation for Configuration: [low]	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. ent is failed and that the associated FreezeFrame contains er with the memorized value. r Release 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used).	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the eventhe FreezeFrameCount Needed Adaptation for Configuration: [low] Test Steps: [low]	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. rent is failed and that the associated FreezeFrame contains er with the memorized value. r Release 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). r Release 4.1.1	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the eventhe FreezeFrameCount Needed Adaptation for Configuration: [low] Test Steps: [low]	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. ent is failed and that the associated FreezeFrame contains er with the memorized value. r Release 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). r Release 4.1.1 Some configuration parameters and values have different	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the eventhe FreezeFrameCount Needed Adaptation for Configuration: [low] Test Steps: [low]	FrameCounter is incremented by 1 during each invocation of ole. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. rent is failed and that the associated FreezeFrame contains er with the memorized value. r Release 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). r Release 4.1.1	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the eventhe FreezeFrameCount Needed Adaptation for Configuration: [low] Test Steps: [low]	FrameCounter is incremented by 1 during each invocation of ble. ne. After 2 seconds, clear the pre-stored FreezeFrame, dependent of the current FreezeFrameCounter value. The ent is failed and that the associated FreezeFrame contains the ent with the memorized value. The ent is failed and that the associated FreezeFrame contains the ent with the memorized value. The elease 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). The elease 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1	
Needed Adaptation to	SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the eventhe FreezeFrameCount Needed Adaptation for Configuration: [low] Test Steps: [low]	FrameCounter is incremented by 1 during each invocation of ble. ne. After 2 seconds, clear the pre-stored FreezeFrame, dependent of the current FreezeFrameCounter value. The ent is failed and that the associated FreezeFrame contains the ent with the memorized value. The ent is failed and that the associated FreezeFrame contains the ent with the memorized value. The elease 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). The elease 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1 DemFreezeFrameCapture renamed to	
Needed Adaptation to	SWC's variable Freeze the SWC's variable Freeze the SWC's main runnal Pre-store a FreezeFran report a failed event and Then, check that the eventhe FreezeFrameCount Needed Adaptation for Configuration: [low] Test Steps: [low] Needed Adaptation for Configuration: [low]	FrameCounter is incremented by 1 during each invocation of ble. ne. After 2 seconds, clear the pre-stored FreezeFrame, d memorize the current FreezeFrameCounter value. rent is failed and that the associated FreezeFrame contains er with the memorized value. r Release 3.2.2 Some configuration parameters have different names in R3.2 than in R4.x (e.g DemHealingAllowed instead of DemAgingAllowed). In R3.2, the API function Dem_GetEventFreezeFrameData() does not exist. So the test steps related to this API need to be removed (only retrieving of FreezeFrames by diagnostic tester is used). r Release 4.1.1 Some configuration parameters and values have different names in R4.1.1 than in R4.2.1	

		DEM CAPTURE S	YNCHRONOUS_TO_REPORTING
	<u> </u>	DEM_O/II TOILE_O	THE INCINCINGUE TO NET CIVING
	Needed Adaptation for F	Release 4.2.1	
	Configuration: [low]	not affected by R4.2	2.2
	Test Steps: [none]		
Pre-conditions	1. Event E07 is not FAILE		
	Operation cycle for eve	nt E07 has started.	
Main Test Execu	ution		
Test Steps			Pass Criteria
Step 1	[SWC] Execute "check event reso E07	et" group for event	All test steps passed
Step 2	[SWC] Invoke Dem_PrestoreFree	ezeFrame(E07)	E_OK is returned
Step 3	[SWC] Delay for 2 seconds		
Step 4	[SWC] Invoke Dem_ClearPrestoredFree	zeFrame(E07)	E_OK is returned
Step 5	[SWC] Memorize the current value of FreezeFrameCounter in FFC		
Step 6	[SWC] Invoke Dem_SetEventStatus(E07, FAILED)		E_OK is returned
Step 7	[SWC] Delay for 1 second		
Step 8	Execute "check failed event" test steps for event E07		All test steps passed
	Alternatively: [LT <diagnostic>] Read out DemDTC = 0x777777 and the associated FreezeFrame. Or: [SWC] Invoke Dem_GetEventFreezeFrameData() to retrieve FreezeFrame of</diagnostic>		Retrieving FreezeFrame successful
Step 10	DemDTC=0x777777 [SWC] Check the FreezeFrameC above FreezeFrame	Counter of the	FreezeFrameCounter has the same value as FFC
Step 11	[SWC] Execute "reset event" test E07		All test steps passed
Post- conditions	1. Event E07 is reset.		

5.2 Test Cases on DataServices (Dcm)

5.2.1 [ATS_DIAG_00022] Writing and reading of data with fixed length

Test Objective	Writing and reading of da	ta with fixed length		
ID	ATS DIAG 00022		4.0.3 4.1.1 4.2.1 4.2.2	
	/ (10_DI/(0_00022	Releases	1.0.0 4.1.1 4.2.1 4.2.2	
Affected Modules	Dcm	State	reviewed	
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00019			
Trace to SWS Item	DiagnosticCommunicationManager: SWS_Dcm_00253 DiagnosticCommunicationManager: SWS_Dcm_00439 DiagnosticCommunicationManager: SWS_Dcm_00437 DiagnosticCommunicationManager: SWS_Dcm_00255 DiagnosticCommunicationManager: SWS_Dcm_00395			
Requirements / Reference to Test Environment	[SWC] A tester SWC that interacts with the RTE interface of the SUT [LT <diagnostic>] Lower Tester that is capable of requesting UDS services on the SUT and receiving the UDS responses from the SUT</diagnostic>			
Configuration Parameters	A data record identifier (DID) with the following properties must be configured: - DcmDspDidIdentifier = 0xFE11 - DcmDspDidInfo.DcmDspDidDynamicallyDefined = FALSE - DcmDspDataSize = 32 - DcmDspDataUsePort = USE_DATA_SYNCH_CLIENT_SERVER - The DcmDspDataType of used signals must be set to UINT8 - Callbacks for "ConditionCheckRead", "ReadData" and "WriteData" mapped to runnables [RUN <conditioncheckread>], [RUN<readdata>] and [RUN<writedata>] of the [SWC]</writedata></readdata></conditioncheckread>			
Summary	Verify that writing and reading of DID data with fixed length works correctly: First, the DID data is read. This value is then changed and written back to the DID. When again reading the DID data, the changed value is expected.			
Adaptation to	Needed Adaptation for Release 3.2.2			
other Releases	Configuration: [low] Test Steps: [low]	in R3.2.2. In R3.2.2, the callou (but not in R4.x) and	mes for DID configuration are different It "ConditionCheckWrite" is available d should therefore be checked has sent the "WriteDataByldentifier"	
	Needed Adaptation for Release 4.1.1			
	Configuration: [low] Test Steps: [none]	been changed or re adapted for releases and variable length	ructure and some parameters have moved. The configuration needs to be s earlier than R4.2.1.(e.g. the fixed in R4.2.1 is no longer determined by ength but by the DcmDspDataType)	
	Needed Adaptation for Release 4.2.1			
	Configuration: [low]	not affected by R4.	2.2	
Needed Adaptation for Release 4.2.1				

	Toot Stone: [none]				
Dro conditions	Test Steps: [none]				
	1. UDS connection between [LT <diagnostic>] and SUT established 2. [SWC] The variable "val0" is initialized with a random value of type UINT32</diagnostic>				
Main Test Execu		· · · · · · · · · · · · · · · · · · ·			
Test Steps	Pass Criteria				
Step 1	[LT <diagnostic>]</diagnostic>				
	Send UDS request "ReadDataByldentifier"				
Step 2	for DID=0xFE11 to SUT [RUN <conditioncheckread>]</conditioncheckread>	[RUN <conditioncheckread>] is</conditioncheckread>			
Step 2	Wait for invocation	called by the RTE			
		(SWS_Dcm_00439)			
Step 3	[RUN <conditioncheckread>]</conditioncheckread>				
	Return 0 as error code and E_OK as return value				
Step 4	RUN <readdata> </readdata>	[RUN <readdata>] is called by the</readdata>			
Otop 4	Wait for invocation	RTE (SWS_Dcm_00437)			
Step 5	[RUN <readdata>]</readdata>				
	Return the value "val0" as DID data and E_OK as return value				
Step 6	[LT <diagnostic>]</diagnostic>	Positive response code received			
Otep 0	Receive UDS response from SUT and store	i ositive response code received			
	received DID value in "val1"				
Step 7	[LT] Calculate "val1" = "val1" XOR 0x01234567				
Step 8	[LT <diagnostic>]</diagnostic>				
Step 0	Send UDS request "WriteDataByldentifier"				
	for DID=0xFE11 and with value "val1" to				
Step 9	SUT	[DLIN WriteDates] is called by the			
Step 9	[RUN <writedata>] Wait for invocation</writedata>	[RUN <writedata>] is called by the RTE (SWS_Dcm_00395)</writedata>			
Step 10	[RUN <writedata>]</writedata>				
	Store the received DID value to "val0",				
	return 0 as error code and E_OK as return value				
Step 11	[LT <diagnostic>]</diagnostic>	Positive response code is received			
•	Receive UDS response from SUT	·			
Step 12	[LT <diagnostic>]</diagnostic>				
	Send UDS request "ReadDataByIdentifier" for DID=0xFE11 to SUT				
Step 13	[RUN <conditioncheckread>]</conditioncheckread>	[RUN <conditioncheckread>] is</conditioncheckread>			
	Wait for invocation	called by the RTE			
Ston 44	(DUN, Condition Class I Day 1.1	(SWS_Dcm_00439)			
Step 14	[RUN <conditioncheckread>] Return E_OK as return value</conditioncheckread>				
Step 15 [RUN <readdata>]</readdata>		[RUN <readdata>] is called by the</readdata>			
Wait for invocation		RTE (SWS_Dcm_00437)			
Step 16	[RUN <readdata>]</readdata>				
	Return the value "val0" as DID data and E OK as return value				
Step 17	[LT <diagnostic>]</diagnostic>	Positive response code is received			
	Receive UDS response from SUT and store				
01	received DID value in "val2"	" - 14" 1 " 10" !			
Step 18	[LT]	"val1" and "val2" have the same value			

	Compare "val1" and "val2"	
Post- conditions	None	

5.2.2 [ATS_DIAG_00023] Writing and reading of data with dynamic length

Test Objective	Writing and reading of data with dynamic length			
ID	· · · · · · · · · · · · · · · · · · ·			
טו	ATS_DIAG_00023		Releases	4.0.3 4.1.1 4.2.1 4.2.2
Affected Modules	Dcm	S	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00019			
Trace to SWS Item	DiagnosticCommunicationManager: SWS_Dcm_00253 DiagnosticCommunicationManager: SWS_Dcm_00439 DiagnosticCommunicationManager: SWS_Dcm_00436 DiagnosticCommunicationManager: SWS_Dcm_00437 DiagnosticCommunicationManager: SWS_Dcm_00255 DiagnosticCommunicationManager: SWS_Dcm_00395			
	[SWC] A tester SWC that interacts with the RTE interface of the SUT [LT <diagnostic>] Lower Tester that is capable of requesting UDS services on the SUT and receiving the UDS responses from the SUT</diagnostic>			
Configuration Parameters	A data record identifier (DID) with the following properties must be configured: - DcmDspDidIdentifier = 0xFE22 - DcmDspDidInfo.DcmDspDidDynamicallyDefined = FALSE - DcmDspDataSize = 256 - DcmDspDataUsePort = USE_DATA_SYNCH_CLIENT_SERVER - The DcmDspDataType of used signals must be set to UINT8_DYN - Callbacks for "ConditionCheckRead", "ReadDataLength", "ReadData" and "WriteData" mapped to runnables [RUN <conditioncheckread>], [RUN<readdatalength>], [RUN<readdata>] and [RUN<writedata>] of the [SWC]</writedata></readdata></readdatalength></conditioncheckread>			
Summary	Verify that writing and reading of DID data with dynamic length works correctly: Execute the following for the data lengths 1, 5, 16, 23 and 32: First, the DID data is read. This value is then changed and written back to the DID. When again reading the DID data, the changed value is expected.			
Needed Adaptation to	Needed Adaptation for	Release	3.2.2	
other Releases	Configuration: [low] Test Steps: [low]	in R3.2.2 In R3.2.2 (but not in	e. e, the callou n R4.x) and	mes for DID configuration are different t "ConditionCheckWrite" is available d should therefore be checked has sent the "WriteDataByldentifier"
	Needed Adaptation for Release 4.1.1			
	Configuration: [low] Test Steps: [none]	been cha adapted t	anged or rer for releases	ructure and some parameters have moved. The configuration needs to be a earlier than R4.2.1.(e.g. the fixed in R4.2.1 is no longer determined by
35 of 53		anu vana	able lettyttt i	Document ID 627: AUTOSAR ATS DiagnosticService

		DcmDspDataFixedL	ength but by the DcmDspDataType)	
		D. J 4 0 4		
	Needed Adaptation for Release 4.2.1			
	Configuration: [low]	not affected by R4.2	2.2	
	Test Steps: [none]	not anected by 144.2	L.L	
Pre-conditions	1. UDS connection between	en [LT <diagnostic>]</diagnostic>	and SUT established	
Main Test Execu	ution			
Test Steps			Pass Criteria	
Step 1	Execute the following loop multiple times with variable "len" = 1, 5, 16, 23, 32 for each iteration.			
Step 2	Start of LOOP			
	[SWC] Initialize the variable"val0" with random data of "len" bytes length			
Step 3	[LT <diagnostic>] Send UDS request "ReadDataByldentifier" for DID=0xFE22 to SUT</diagnostic>			
Step 4	[RUN <conditioncheckread>] Wait for invocation</conditioncheckread>		[RUN <conditioncheckread>] is called by the RTE (SWS_Dcm_00439)</conditioncheckread>	
Step 5	[RUN <conditioncheckread>] Return 0 as error code and E_OK as return value</conditioncheckread>			
Step 6	[RUN <readdatalength>] Wait for invocation</readdatalength>		[RUN <readdatalength>] is called by the RTE (SWS_Dcm_00436)</readdatalength>	
Step 7	[RUN <readdatalength>] Return the current data length "len" and E_OK as return value</readdatalength>			
Step 8	 [RUN <readdata>] Wait for invocation</readdata>		[RUN <readdata>] is called by the RTE (SWS_Dcm_00437)</readdata>	
Step 9	[RUN <readdata>] Return the value "val0" as DID data and E_OK as return value</readdata>			
Step 10	[LT <diagnostic>] Receive UDS response from SUT and store received DID value in "val1"</diagnostic>		Positive response code received	
Step 11	[LT] "val1" = "val1" XOR 0x0123456789ABCDEF0102 ("len" bytes length)			
Step 12	[LT <diagnostic>] Send UDS request "WriteDataByIdentifier" for DID=0xFE22 and with value "val1" to SUT</diagnostic>			
Step 13	[RUN <writedata>] Wait for invocation</writedata>		[RUN <writedata>] is called by the RTE (SWS_Dcm_00395)</writedata>	
Step 14	[RUN <writedata>] Store the received DID value to "val0", return 0 as error code and E_OK as return value</writedata>			

Step 15	[LT <diagnostic>] Receive UDS response from SUT</diagnostic>	Positive response code is received
Step 16	[LT <diagnostic>] Send UDS request "ReadDataByIdentifier" for DID=0xFE22 to SUT</diagnostic>	
Step 17	[RUN <conditioncheckread>] Wait for invocation</conditioncheckread>	[RUN <conditioncheckread>] is called by the RTE (SWS_Dcm_00439)</conditioncheckread>
Step 18	[RUN <conditioncheckread>] Return 0 as error code and E_OK as return value</conditioncheckread>	
Step 19	[RUN <readdata>] Wait for invocation</readdata>	[RUN <readdata>] is called by the RTE (SWS_Dcm_00437)</readdata>
Step 20	[RUN <readdata>] Return the value "val0" as DID data and E_OK as return value</readdata>	
Step 21	[LT <diagnostic>] Receive UDS response from SUT and store received DID value in "val2"</diagnostic>	Positive response code is received
Step 22	[LT] Compare "val1" and "val2" End of LOOP	"val1" and "val2" have the same length and value
Post- conditions	None	

5.2.3 [ATS_DIAG_00024] Reading data rejected by SWC

Test Objective	Reading data rejected by SWC		
ID	ATS_DIAG_00024	AUTOSAR Releases	4.0.3 4.1.1 4.2.1 4.2.2
Affected Modules	Dcm	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00019		
	DiagnosticCommunicationManager: SWS_Dcm_00253 DiagnosticCommunicationManager: SWS_Dcm_00439		
Reference to Test	[SWC] A tester SWC that interacts with the RTE interface of the SUT. In the SWC's runnable configured as "ReadData" callback, the counter "cnt_read" counts the number of invocations of that runnable. [LT <diagnostic>] Lower Tester that is capable of requesting UDS services on the SUT and receiving the UDS responses from the SUT</diagnostic>		
Configuration Parameters	The data record identifier (DID) with the following properties must be configured: - DcmDspDidIdentifier = 0xFE33 - DcmDspDataUsePort = USE_DATA_SYNCH_CLIENT_SERVER - Callbacks for "ConditionCheckRead" and "ReadData" mapped to runnables [RUN <conditioncheckread>] and [RUN<readdata>] of the [SWC] - Further DID configuration parameters are arbitrary.</readdata></conditioncheckread>		
Summary	Verify that if SWC rejects reading the data is indeed not read: When the read request for a DID is passed to the SWC and the SWC rejects it, the SUT must not call the read callback but send a negative response code.		

Needed	Needed Adaptation for	Release 3.2.2	
Adaptation to			
other Releases	Configuration: [low] Test Steps: [none]	Some parameter na in R3.2.2.	mes for DID configuration are different
	Needed Adaptation for	Release 4.1.1	
	Configuration: [low]	been changed or re	ructure and some parameters have moved. The configuration needs to be s earlier than R4.2.1.(e.g. the fixed
	Test Steps: [none]	and variable length	in R4.2.1 is no longer determined by Length but by the DcmDspDataType)
	Needed Adaptation for	Release 4.2.1	
	Configuration: [low]	not affected by R4.	2.2
	Test Steps: [none]		
Pre-conditions	 UDS connection between [LT<diagnostic>] and SUT established</diagnostic> In the [RUN<readdata>], the counter "cnt_read" that counts the number of invocations of that runnable is initialized with 0.</readdata> 		
Main Test Execu	ution		
Test Steps			Pass Criteria
Step 1	[LT <diagnostic>] Send UDS request "Rea for DID=0xFE33 to SUT</diagnostic>		
Step 2	[RUN <conditioncheckread>] Wait for invocation</conditioncheckread>		[RUN <conditioncheckread>] is called by the RTE (SWS_Dcm_00439)</conditioncheckread>
Step 3	[RUN <conditioncheckr Return 0x22 (conditions) code and E_NOT_OK a</conditioncheckr 	NotCorrect) as error	
Step 4	[LT <diagnostic>] Receive UDS response from SUT with negative response code</diagnostic>		Negative response code = 0x22
Step 5	[SWC] Check that the runnable "ReadData" callback has		Counter "cnt_read" of runnable for "ReadData" callback is still 0
Post- conditions	None		

5.2.4 [ATS_DIAG_00025] Writing data rejected by SWC

Test Objective	Writing data rejected by SWC		
ID		AUTOSAR Releases	4.0.3 4.1.1 4.2.1 4.2.2
Affected Modules	Dcm	State	reviewed
Trace to Requirement on Acceptance	ATR: ATR_ATR_00019		

Test Document				
Trace to SWS	DiagnosticCommunicatio	nManager: SWS_Do	cm 00255	
Item	DiagnosticCommunication			
	[SWC] A tester SWC that interacts with the RTE interface of the SUT. [LT <diagnostic>] Lower Tester that is capable of requesting UDS services on the SUT and receiving the UDS responses from the SUT</diagnostic>			
Configuration Parameters	The data record identifier (DID) with the following properties must be configured: - DcmDspDidIdentifier = 0xFE33 (re-used from AT_DCM_03) - DcmDspDataUsePort = USE_DATA_SYNCH_CLIENT_SERVER - Callback for "WriteData" mapped to runnable [RUN <writedata>] of the [SWC] - Further DID configuration parameters are arbitrary.</writedata>			
Summary	SUT must respond with a	ponse code: or a DID is passed to a negative response	the SWC and the SWC rejects it, the	
Needed Adaptation to	Needed Adaptation for	Release 3.2.2		
other Releases		Some parameter na in R3.2.2.	mes for DID configuration are different	
	Configuration: [low] Test Steps: [middle]	which is called by th	callout "ConditionCheckWrite" exists the DCM before the callout of SWC has to reject the request	
		using "ConditionCheckWrite" and the test case has to check that "WriteData" must not be called by the SUT.		
	Needed Adaptation for	Release 4.1.1		
	Configuration: [low]	been changed or relaases	ructure and some parameters have moved. The configuration needs to be searlier than R4.2.1.(e.g. the fixed	
	Test Steps: [none]		in R4.2.1 is no longer determined by Length but by the DcmDspDataType)	
	Needed Adaptation for	Release 4.2.1		
	Configuration: [low]	not affected by R4.	2.2	
Pre-conditions	Test Steps: [none] 1. UDS connection between	on [Tadiognostic: 1	and SLIT actablished	
Main Test Execu		cen [L1 <ulay110stic>]</ulay110stic>	and SOT Established	
Test Steps	ALLO I		Pass Criteria	
Step 1	[LT <diagnostic>]</diagnostic>			
	Send UDS request "Write for DID=0xFE33 to SUT	eDataByldentifier"		
Step 2	[RUN <writedata>] Wait for invocation</writedata>		[RUN <writedata>] is called by the RTE (SWS_Dcm_00395)</writedata>	
Step 3	[RUN <writedata>] Return 0x22 (conditionsN code and E_NOT_OK a</writedata>			
Step 4	[LT <diagnostic>] Receive UDS response f negative response code</diagnostic>	rom SUT with	Negative response code = 0x22	

Post-	None
conditions	

5.2.5 [ATS_DIAG_00026] Writing and reading of data with endianness conversion

Test Objective	Writing and reading of data with endianness conversion			
•	ATS_DIAG_00026		4.0.3 4.1.1 4.2.1 4.2.2	
Affected Modules	Dcm	State	reviewed	
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00019			
Trace to SWS Item	DiagnosticCommunication DiagnosticCommunication DiagnosticCommunication	nManager: SWS_Do	cm_00638	
	[SWC] A tester SWC that [LT <diagnostic>] Lower T SUT and receiving the UI</diagnostic>	ester that is capable	e of requesting UDS services on the	
Configuration Parameters	A data record identifier (DID) with the following properties must be configured: - DcmDspDidldentifier = 0xFE55 - DcmDspDidlnfo.DcmDspDidDynamicallyDefined = FALSE - DcmDspDataSize = 32 - DcmDspDataUsePort = USE_DATA_SENDER_RECEIVER - The DcmDspDataType of used signals must be set to UINT32 - In the DcmDslProtocolRow used for test execution: - DcmDslProtocolEndiannessConvEnabled = TRUE - A sender/receiver interface "DataService_ <data>" for this DID has to be configured. A read wait point for the sender/receiver interface has to be configured for the SWC.</data>			
Summary	Verify that the SUT correctly performs endianness conversion when writing and reading DID data: The LT writes a defined UINT32 value through UDS to the DID. The SWC receives this value and XORs it with the value 0x01020304. Then, LT reads the DID through UDS and the SWC returns the result of the XOR operation. The LT expects to receive "the original value XOR 0x04030201" because the SUT should have applied the endianness conversions for both the write and the read requests from the LT.			
Adaptation to other Releases	Needed Adaptation for Release 3.2.2			
		been changed or rel adapted for releases and variable length	ructure and some parameters have moved. The configuration needs to be s earlier than R4.2.1.(e.g. the fixed in R4.2.1 is no longer determined by Length but by the DcmDspDataType)	

	Needed Adaptation for Release 4.2.1	
	Configuration: [low] not affected by R4.2 Test Steps: [none]	2.2
	 UDS connection between [LT<diagnostic>]</diagnostic> [SWC] and [LT]: Initialize "org" with a UINT different to each other (e.g. 0x1234ABCD) 	and SUT established 32 value of which each byte is
Main Test Execu	ution	
Test Steps		Pass Criteria
Step 1	[SWC] Wait at read wait point for incoming data	
Step 2	[LT <diagnostic>] Send UDS request "WriteDataByIdentifier" for DID=0xFE55 and with value "org" to SUT</diagnostic>	
Step 3	[SWC] Receive data at read wait point and store it in "rx" (SWS_Dcm_00638)	
Step 4	[LT <diagnostic>] Receive UDS response from SUT</diagnostic>	Positive response code is received
Step 5	[SWC] Calculate "tx = rx XOR 0x01020304"	
Step 6	[SWC] Send "tx" to the sender/receiver interface of DID=0xFE55	
Step 7	[LT <diagnostic>] Send UDS request "ReadDataByIdentifier" for DID=0xFE55 to SUT</diagnostic>	
Step 8	[LT <diagnostic>] Receive UDS response from SUT and store received value in "val"</diagnostic>	Positive response code is received
Step 9	[LT] Compare "val" with "org"	"val" must be equal to "org XOR 0x04030201"
Post- conditions	None	

5.2.6 [ATS_DIAG_00027] Retrieving of scaling information

Test Objective	Retrieving of scaling information		
ID		AUTOSAR Releases	4.0.3 4.1.1 4.2.1 4.2.2
Affected Modules	Dcm	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00019		
	DiagnosticCommunicationManager: SWS_Dcm_00258 DiagnosticCommunicationManager: SWS_Dcm_00394		
•	[SWC] A tester SWC that interacts with the RTE interface of the SUT [LT <diagnostic>] Lower Tester that is capable of requesting UDS services on the</diagnostic>		

to Toot	CLIT and manabide with a the cliff	DO (t	L. OUT
to Test Environment	SUT and receiving the U	o responses from t	ne su i
Configuration Parameters	The data record identifier (DID) with the following properties must be configured: - DcmDspDidIdentifier = 0xFE66 - DcmDspDidInfo.DcmDspDidDynamicallyDefined = FALSE - DcmDspDataSize = 8 - DcmDspDataUsePort = USE_DATA_SYNCH_CLIENT_SERVER - DcmDspDataInfo.DcmDspDataScalingInfoSize = 9 - The DcmDspDataType of used signals must be set to UINT8 - Callbacks for "GetScalingInformation" mapped to runnable [RUN <getscalinginformation>] of the [SWC]</getscalinginformation>		
Summary		the diagnostic reques	st for retrieving the scaling information red for "GetScalingInformation" is
Needed Adaptation to	Needed Adaptation for	Release 3.2.2	
other Releases	Configuration: [low]	Some parameter na in R3.2.2.	mes for DID configuration are different
	Test Steps: [none]		
	Needed Adaptation for	Release 4.1.1	
	Configuration: [low]	The configuration structure and some parameters have been changed or removed. The configuration needs to be adapted for releases earlier than R4.2.1.(e.g. the fixed and variable length in R4.2.1 is no longer determined by DcmDspDataFixedLength but by the DcmDspDataType)	
	Test Steps: [none]		
	Needed Adaptation for Release 4.2.1		
	Configuration: [low]	not affected by R4.2.2	
Due conditions	Test Steps: [none]	an II Taliannastia 1	and OUT astablished
Pre-conditions	1. UDS connection between	een [L1 <diagnostic>]</diagnostic>	and SUT established
Main Test Execu	ution		b 0" ·
Test Steps	ln =		Pass Criteria
Step 1	[LT <diagnostic>] Send UDS request "ReadScalingDataByIdentifier" for DID=0xFE66 to SUT</diagnostic>		
Step 2	[RUN <getscalinginformation>] Wait for invocation</getscalinginformation>		[RUN <getscalinginformation>] is called by the RTE (SWS_Dcm_00394)</getscalinginformation>
Step 3	[RUN <getscalinginformation>] Return the hex data "01 90 00 E0 4B 00 1E A0 30" (taken from example in ISO 14229) as ScalingInfo, 0 as ErrorCode and E_OK as return value</getscalinginformation>		
Step 4	[LT <diagnostic>] Receive UDS response from SUT and store the received DID scaling information in "info"</diagnostic>		Positive response code received
Step 5	[LT] Verify the length and con	tent of "info"	"info" has the length "9" and the hex data content "01 90 00 E0 4B 00 1E A0 30"

Post-	None
conditions	

5.2.7 [ATS_DIAG_00028] Periodical reading of data

Test Objective	Periodical reading of data				
ID	ATS DIAG 00028		4 0 214 1 114 2 114 2 2		
טו	A15_DIAG_00028	Releases	4.0.3 4.1.1 4.2.1 4.2.2		
Affected Modules	Dcm	State	reviewed		
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00019				
Trace to SWS Item	DiagnosticCommunicatio DiagnosticCommunicatio DiagnosticCommunicatio	nManager: SWS_D	cm_00254		
Requirements / Reference to Test Environment	SUT and receiving the UI	ester that is capabl DS responses from	e of requesting UDS services on the the SUT		
Configuration Parameters	The transmission of periodic responses are configured to use the normal diagnostic responses: - DcmDslProtocolTransType = TYPE1 - An additional protocol (DcmDslProtocolRow) with DcmDslProtocolID = DCM_PERIODIC_TRANS_ON_IP DCM_PERIODIC_ON_CAN DCM_PERIODIC_ON_FLEXRAY - DcmDslPeriodicTransmissionConRef = <reference above="" additional="" protocol="" to=""> A periodic data record identifier (DID) with the following properties must be configured: - DcmDspDidIdentifier = 0xF277 (inside periodical DID range) - DcmDspDidInfo.DcmDspDidDynamicallyDefined = FALSE - DcmDspDataSize = 32 - DcmDspDataUsePort = USE_DATA_SYNCH_CLIENT_SERVER - The DcmDspDataType of used signals must be set to UINT32 - Callbacks for "ReadData" and "WriteData" mapped to runnables</reference>				
	[RUN <readdata>] and [RUN<writedata>] of the [SWC] Verify that the periodical reading of data works correctly: First, the periodical DID is written to with a random value. Then, when the SUT receives the diagnosis request for periodical reading the data of a DID, verify that the SWC function configured for "ReadData" is called periodically and that the SUT sends the correct data (which is changing) in all periodic responses for a specified time. After sending the diagnosis request for stopping the periodical read, the SUT must stop sending the periodic responses.</writedata></readdata>				
Needed Adaptation to	Needed Adaptation for Release 3.2.2				
other Releases	Some parameter names for DID configuration in R3.2.2. Configuration: [low]				
	In R3.2.2, the callout "ConditionCheckWrite" (but not in R4.x) and should therefore be chadditionally after LT has sent the "WriteData request.				

	Needed Adaptation fo	r Release 4.1.1	
	Configuration: [low]	been changed or re	ructure and some parameters have moved. The configuration needs to be s earlier than R4.2.1.(e.g. the fixed
	Test Steps: [none]	and variable length	in R4.2.1 is no longer determined by Length but by the DcmDspDataType)
	Needed Adaptation for Release 4.2.1		
	Configuration: [low]	not affected by R4.	2.2
	Test Steps: [none]		
Pre-conditions	2. [LT] The variable "rno		and SUT established andom value of type UINT32
Main Test Exec	ution		
Test Steps			Pass Criteria
Step 1	[LT <diagnostic>] Send UDS request "WriteDataByIdentifier" for DID=0xF277 and with value "rnd" to SUT</diagnostic>		
Step 2	[RUN <writedata>]</writedata>		[RUN <writedata>] is called by the RTE (SWS_Dcm_00395)</writedata>
Step 3	[RUN <writedata>] Store the received DID value to "val", return 0 as error code and E_OK as return value</writedata>		
Step 4	[LT <diagnostic>] Receive UDS response from SUT</diagnostic>		Positive response code is received
Step 5	[LT <diagnostic>] Send UDS request "ReadDataByPeriodicIdentifier" for periodic DID=0x77 (i.e. 0xF277) with transmissionMode=0x02 (sendAtMediumRate) to SUT</diagnostic>		
Step 6	The following loop is repeated and terminated by [LT <diagnostic>] sending a UDS request with "transmissionMode=stopSending"</diagnostic>		
Step 7	Start of LOOP [RUN <readdata>] Wait for invocation</readdata>		[RUN <readdata>] is called by the RTE (SWS_Dcm_00437)</readdata>
Step 8	[RUN <readdata>] Return the value "val" as DID data and E_OK as return value</readdata>		
Step 9	[SWC] Increment "val" by 1 (with wrap-around)		
Step 10	[LT <diagnostic>] Receive UDS response from SUT and check received DID value</diagnostic>		Positive response code is received. Received DID value must be equal to "rnd"
Step 11	[LT] Increment "rnd" by 1 (with wrap-around)		
Step 12	[LT <diagnostic>] If the loop has not alrea</diagnostic>	ady been iterated for	

	10 times, go to the beginning of the loop End of LOOP	
Step 13	[LT <diagnostic>] Send UDS request "ReadDataByPeriodicIdentifier" for periodic DID=0x77 (i.e. 0xF277) with transmissionMode=0x04 (stopSending) to SUT</diagnostic>	
Step 14	[LT <diagnostic>] Receive UDS response from SUT</diagnostic>	Positive response code is received
Step 15	Check for further UDS responses for service	No UDS responses for service "ReadDataByPeriodicIdentifier" are received anymore
Post- conditions	None	

5.2.8 [ATS_DIAG_00029] Invocation of callbacks for "InputOutputControlByldentifier"

To at Objective	lavoration of callbooks for Street)ttO = t	ID. dala askiti a a"
	Invocation of callbacks for "InputC		
ID	ATS_DIAG_00029	AUTOSAR Releases	4.0.3 4.1.1 4.2.1 4.2.2
Affected Modules	Dcm	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00019		
Trace to SWS Item	DiagnosticCommunicationManager: SWS_Dcm_00256 DiagnosticCommunicationManager: SWS_Dcm_00579 DiagnosticCommunicationManager: SWS_Dcm_00396 DiagnosticCommunicationManager: SWS_Dcm_00397 DiagnosticCommunicationManager: SWS_Dcm_00398 DiagnosticCommunicationManager: SWS_Dcm_00399		
Requirements / Reference to Test Environment	[SWC] A tester SWC that observes the RTE interface of the SUT [LT <diagnostic>] Lower Tester that is capable of requesting UDS services on the SUT and receiving the UDS responses from the SUT</diagnostic>		
Configuration Parameters	The data record identifier (DID) with the following properties must be configured: - DcmDspDidIdentifier = 0xFE88 - Contains one signal - DcmDspDataSize = 32 - DcmDspDataUsePort = USE_DATA_SYNCH_CLIENT_SERVER - DcmDspDidFreezeCurrentState = TRUE - DcmDspDidResetToDefault = TRUE - DcmDspDidShortTermAdjustment = TRUE - The DcmDspDataType of used signals must be set to UINT32 - Callbacks for "ShortTermAdjustment", "FreezeCurrentState", "ReturnControlToECU" and "ResetToDefault" of UDS service "InputOutputControlByIdentifier" mapped to runnables [RUN <shorttermadjustment>], [RUN<freezecurrentstate>], [RUN<returncontroltoecu>] and [RUN<resettodefault>] of the [SWC] - RAM of appropriate size assigned</resettodefault></returncontroltoecu></freezecurrentstate></shorttermadjustment>		

	1				
	Furthermore, a runnable to serve the Xxx_ReadData() callout for DID=0xFE88 must be implemented and mapped. This runnable reads data from the RAM assigned to the DID and returns it.				
Summary	Verify that the IOControl requests are correctly passed to the SWC: When the SUT receives each of the IOControl requests "ShortTermAdjustment", "FreezeCurrentState", "ReturnControlToECU" and "ResetToDefault", verify that the SWC runnable configured for the specific request is correctly called.				
Needed	Needed Adaptation for Release 3.2.2				
Adaptation to other Releases					
other Keleases	Configuration: [low]	Some parameter na in R3.2.2.	mes for DID configuration are different		
	Test Steps: [none]				
	Needed Adaptation for	Release 4.1.1			
	Configuration: [low] Test Steps: [low]	Some parameter names for DID configuration are differe or have been removed since AR4.1.1. ControlOptionRecord was renamed to ControlStateInfo, DcmDspDidReturnControlToEcu is not needed any more due to a configuration that is using multiplicity. The configuration needs to be adapted for releases earlier than R4.2.1.(e.g. the fixed and variable length in R4.2.1 no longer determined by DcmDspDataFixedLength but be the DcmDspDataType)			
	Needed Adaptation for Configuration: [low]	Release 4.2.1 not affected by R4.	2.2		
Pre-conditions	Test Steps: [none]	een [l Tzdiagnostics]	and SLIT established		
Main Test Exec	re-conditions 1. UDS connection between [LT <diagnostic>] and SUT established</diagnostic>				
Test Steps			Pass Criteria		
Step 1	[SWC] Write 0x1A1B1C1D into DID=0xFE88	memory of			
Step 2	[LT <diagnostic>] Send UDS request "InputOutputControlByIdentifier" for DID=0xFE88 and with "ControlStateInfo" = 0x03AABBCCDD, i.e. first byte = "InputOutputControlParameter" = 0x03 (ShortTermAdjustment) to SUT</diagnostic>				
Step 3	[RUN <shorttermadustment>] Wait for invocation</shorttermadustment>		[RUN <shorttermadustment>] is called by the RTE (SWS_Dcm_00399)</shorttermadustment>		
Step 4	[RUN <shorttermadjustment>] Check input parameter "ControlStateInfo", then return 0x00 as ErrorCode and E_OK as return value</shorttermadjustment>		"ControlStateInfo" must contain 0x03AABBCCDD		
Step 5	return value [LT <diagnostic>] Receive UDS response for "InputOutputControlByIdentifier" from SUT and check response code and</diagnostic>		- Response code indicates no error - "ControlStatusRecord" contains 0x031A1B1C1D		

	"ControlStatusRecord" of the response	
Step 6	[SWC] Write 0x2A2B2C2D into memory of DID=0xFE88	
Step 7	[LT <diagnostic>] Send UDS request "InputOutputControlByIdentifier" for DID=0xFE88 and with "InputOutputControlParameter" set to 0x02 (FreezeCurrentState) to SUT</diagnostic>	
Step 8	[RUN <freezecurrentstate>] Wait for invocation</freezecurrentstate>	[RUN <freezecurrentstate>] is called by the RTE (SWS_Dcm_00398)</freezecurrentstate>
Step 9	[RUN <freezecurrentstate>] Return 0x00 as ErrorCode and E_OK as return value</freezecurrentstate>	
Step 10	[LT <diagnostic>] Receive UDS response for "InputOutputControlByIdentifier" from SUT and check response code and "ControlStatusRecord" of the response</diagnostic>	- Response code indicates no error - "ControlStatusRecord" contains 0x022A2B2C2D
Step 11	[SWC] Write 0x3A3B3C3D into memory of DID=0xFE88	
Step 12	[LT <diagnostic>] Send UDS request "InputOutputControlByIdentifier" for DID=0xFE88 and with "InputOutputControlParameter" set to 0x00 (ReturnControlToECU) to SUT</diagnostic>	
Step 13	[RUN <returncontroltoecu>] Wait for invocation</returncontroltoecu>	[RUN <returncontroltoecu>] is called by the RTE [SWS_Dcm_00396]</returncontroltoecu>
Step 14	[RUN <returncontroltoecu>] Return 0x00 as ErrorCode and E_OK as return value</returncontroltoecu>	
Step 15	[LT <diagnostic>] Receive UDS response for "InputOutputControlByIdentifier" from SUT and check response code and "ControlStatusRecord" of the response</diagnostic>	- Response code indicates no error - "ControlStatusRecord" contains 0x003A3B3C3D
Step 16	[SWC] Write 0x4A4B4C4D into memory of DID=0xFE88	
Step 17	[LT <diagnostic>] Send UDS request "InputOutputControlByIdentifier" for DID=0xFE88 and with "InputOutputControlParameter" set to 0x01 (ResetToDefault) to SUT</diagnostic>	
Step 18	[RUN <resettodefault>] Wait for invocation</resettodefault>	[RUN <resettodefault>] is called by the RTE (SWS_Dcm_00397)</resettodefault>
Step 19	[RUN <resettodefault>] Return 0x00 as ErrorCode and E_OK as return value</resettodefault>	
Step 20	[LT <diagnostic>] Receive UDS response for</diagnostic>	- Response code indicates no error - "ControlStatusRecord" contains

	"InputOutputControlByIdentifier" from SUT and check response code and "ControlStatusRecord" of the response	0x014A4B4C4D
Post- conditions	None	

5.3 Test Cases on RoutineServices (Dcm)

5.3.1 [ATS_DIAG_00030] Handling of RoutineControl with fixed-length data

Test Objective	Handling of RoutineContr	rol with fix	ed-length o	data
ID	ATS DIAG 00030			4.0.3 4.1.1 4.2.1 4.2.2
	M10_DIMO_00000		Releases	1.0.0 4.1.1 4.2.1 4.2.2
Affected Modules	Dcm	S	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00019			
Trace to SWS Item	DiagnosticCommunicationManager: SWS_Dcm_00257 DiagnosticCommunicationManager: SWS_Dcm_00400 DiagnosticCommunicationManager: SWS_Dcm_00401 DiagnosticCommunicationManager: SWS_Dcm_00402 DiagnosticCommunicationManager: SWS_Dcm_00403 DiagnosticCommunicationManager: SWS_Dcm_00404 DiagnosticCommunicationManager: SWS_Dcm_00405			
Requirements / Reference to Test Environment	[SWC] A tester SWC that interacts with the RTE interface of the SUT [LT <diagnostic>] Lower Tester that is capable of requesting UDS services on the SUT and receiving the UDS responses from the SUT</diagnostic>			
Configuration Parameters	A routine identifier (RID) with the following properties(in DcmDspRoutine) must be configured: - DcmDspRoutineIdentifier = 0x0200 - DcmDspRoutineUsed = TRUE - DcmDspStopRoutineSupported = TRUE - DcmDspRequestResultsRoutineSupported = TRUE - DcmDspRoutineInfoRef->DcmDspStartRoutineIn configured as 1 UINT8 signal - DcmDspRoutineInfoRef->DcmDspStartRoutineOut configured as 2 UINT8 signals - DcmDspRoutineInfoRef->DcmDspRoutineStopIn configured as 3 UINT8 signals - DcmDspRoutineInfoRef->DcmDspRoutineStopOut configured as 4 UINT8 signals - DcmDspRoutineInfoRef->DcmDspRoutineRequestResOut configured as 5 UINT8 signals - DcmDspRoutineUsePort = TRUE - Callouts for "StartRoutine", "StopRoutine" and "RequestResultsRoutine" mapped to runnables [RUN <startroutine>], [RUN<stoproutine>] and [RUN<requestresultsroutine>] of the [SWC]</requestresultsroutine></stoproutine></startroutine>			
Summary	Verify that starting, stopping and requesting results for a routine with fixed-length data works correctly: The test system starts and stops the configured routine and requests its result. During these operations the size and the content of the passed data is checked.			
Needed Adaptation to	Needed Adaptation for Release 3.2.2			
other Releases	Configuration: [low] Parameter names for configuration of routines (contained DcmDspRoutine) have changed between R3.2 and R4.			
	Needed Adaptation for Release R4.1.1			
	Configuration: [low]	was removed in R4.2.1 and needs to be set to TRUE		
	Test Steps: [none]			

	Needed Adaptation for I	Release 4.2.1	
	Configuration: [low]		
	Test Steps: [none]	not affected by R4.2	2.2
Pre-conditions	1. UDS connection betwe	en [l T <diagnostic>]</diagnostic>	and SLIT established
Main Test Execu		cri [L1 \diagriostic>]	and GOT established
Test Steps			Pass Criteria
Step 1	[LT <diagnostic>] Send UDS request "Routing sub-function "StartRouting RoutineControlOptionRec</diagnostic>	e", RID=0x0200 and	
Step 2	[RUN <startroutine>] Wait for invocation (SWS</startroutine>	_Dcm_00400)	[RUN <startroutine>] is called by the RTE</startroutine>
Step 3	[RUN <startroutine>] Check the received argun [SWS_Dcm_00400]</startroutine>	nent "dataIn1"	"dataIn1" is equal to 0xA0
Step 4	[RUN <startroutine>] Return "dataOut1"=0xA1, 0 as error code and E_Oh</startroutine>		
Step 5	[LT <diagnostic>] Receive UDS response for service</diagnostic>		Positive response code received. "RoutineStatusRecord" is of size 2 and has the values "0xA1 0xA2"
Step 6	[LT <diagnostic>] Send UDS request "RoutineControl" with sub-function "StopRoutine", RID=0x0200 and RoutineControlOptionRecord="0xB0 0xB1 0xB2"</diagnostic>		
Step 7	Wait for invocation (SWS_Dcm_00402)		[RUN <stoproutine>] is called by the RTE</stoproutine>
Step 8	[RUN <stoproutine>] Check the received arguments "dataIn1", "dataIn2" and "dataIn3" (SWS_Dcm_00402)</stoproutine>		"dataIn1" is equal to 0xB0, "dataIn2" is equal to 0xB1, "dataIn3" is equal to 0xB2
Step 9	[RUN <stoproutine>] Return "dataOut1"=0xB3, "dataOut2"=0xB4, "dataOut3"=0xB5, "dataOut4"=0xB6, 0 as error code and E_OK as return value</stoproutine>		
Step 10	[LT <diagnostic>] Receive UDS response for service "RoutineControl" from SUT and check the received "RoutineStatusRecord" (SWS_Dcm_00403)</diagnostic>		Positive response code received. "RoutineStatusRecord" is of size 4 and has the values "0xB3 0xB4 0xB5 0xB6"
Step 11	[LT <diagnostic>] Send UDS request "RoutineControl" with sub-function "RequestRoutineResults" and RID=0x0200</diagnostic>		
Step 12	[RUN <requestresults>] Wait for invocation (SWS_Dcm_00404)</requestresults>		[RUN <requestresults>] is called by the RTE</requestresults>
Step 13	[RUN <requestresults>] Return "dataOut1"=0xC0, "dataOut3"=0xC2, "dataO "dataOut5"=0xC4, 0 as er</requestresults>	out4"=0xC3,	

	as return value	
·	Receive UDS response for service "RoutineControl" from SUT and check the	Positive response code received. "RoutineStatusRecord" is of size 5 and has the values "0xC0 0xC1 0xC2 0xC3 0xC4"
Post- conditions	None	

5.3.2 [ATS_DIAG_00031] Handling of RoutineControl with dynamic-length data

Test Objective	Handling of RoutineControl with dynamic-length data		
	ATS_DIAG_00031		4.0.3 4.1.1 4.2.1 4.2.2
Affected Modules	Dcm	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00019	•	
	DiagnosticCommunicationManager: SWS_Dcm_00257 DiagnosticCommunicationManager: SWS_Dcm_00400 DiagnosticCommunicationManager: SWS_Dcm_00401 DiagnosticCommunicationManager: SWS_Dcm_00402 DiagnosticCommunicationManager: SWS_Dcm_00403 DiagnosticCommunicationManager: SWS_Dcm_00404 DiagnosticCommunicationManager: SWS_Dcm_00405		
	[SWC] A tester SWC that interacts with the RTE interface of the SUT [LT <diagnostic>] Lower Tester that is capable of requesting UDS services on the SUT and receiving the UDS responses from the SUT</diagnostic>		
	A routine identifier (RID) with the following properties(in DcmDspRoutine) must be configured: - DcmDspRoutineIdentifier = 0x0201 - DcmDspRoutineUsed = TRUE - DcmDspRoutineFixedLength = FALSE - DcmDspStopRoutineSupported = TRUE - DcmDspRequestResultsRoutineSupported = TRUE - DcmDspRoutineUsePort = TRUE - All in and out parameters of the Start, Stop and RequestResults routines are configured with variable length - Callouts for "StartRoutine", "StopRoutine" and "RequestResultsRoutine" mapped to runnables [RUN <startroutine>], [RUN<stoproutine>] and [RUN<requestresultsroutine>] of the [SWC]</requestresultsroutine></stoproutine></startroutine>		
Summary	Verify that starting, stopping and requesting results for a routine with dynamic- length data works correctly: The test system starts and stops the configured routine and requests its result. During these operations, the size and the content of the passed data is checked.		
Needed Adaptation to other Releases	Configuration: [N/A] Feature Test Steps: [N/A]		s not available in R3.2.2
	·		

Needed Adaptation for Release 4.1.1 The values N1, N2,, N5 are defining a size in byte R4.1.1 they define a size in bits. Test Steps: [none] Configuration parameter DcmDspRoutineFixedLengt was removed in R4.2.1 and needs to be set to FALSI Needed Adaptation for Release 4.2.1	า
Configuration: [low] R4.1.1 they define a size in bits. Test Steps: [none] Configuration parameter DcmDspRoutineFixedLengt was removed in R4.2.1 and needs to be set to FALS	า
was removed in R4.2.1 and needs to be set to FALS	
Needed Adaptation for Release 4.2.1	
Configuration: [low] not affected by R4.2.2	
Test Steps: [none]	
Pre-conditions 1. The sizes N1, N2,, N5 (each of value 165535 byte) for the dynamic leng have been defined. 2. Arbitrary data contents for "DATA1 of length N1", "DATA2 of length N2" etc. to "DATA5") have been defined. These data contents shall be byte-wise difference from each other. 3. UDS connection between [LT <diagnostic>] and SUT established</diagnostic>	(up
Main Test Execution	
Test Steps Pass Criteria	
Step 1 [LT <diagnostic>] Send UDS request "RoutineControl" with sub-function "StartRoutine", RID=0x0201 and RoutineControlOptionRecord="DATA1"</diagnostic>	
Step 2 [RUN <startroutine>] [RUN<startroutine>] is called by Wait for invocation (SWS_Dcm_00400) RTE</startroutine></startroutine>	the
Step 3 [RUN <startroutine>] "currentDataLength" is equal to Note the received arguments "dataIn1" is equal to "DATA1"</startroutine>	I 1,
Step 4 [RUN <startroutine>] Return "dataOut1" = "DATA2", "currentDataLength" = N2, 0 as error code and E_OK as return value</startroutine>	
Step 5 [LT <diagnostic>] Receive UDS response for service "RoutineControl" from SUT and check the received "RoutineStatusRecord" (SWS_Dcm_00401) Positive response code received "RoutineStatusRecord" take the content equal to "DA" (SWS_Dcm_00401)</diagnostic>	N2
Step 6 [LT <diagnostic>] Send UDS request "RoutineControl" with sub-function "StopRoutine", RID=0x0201 and RoutineControlOptionRecord="DATA3"</diagnostic>	
Step 7 [RUN <stoproutine>] [RUN<stoproutine>] is called by Wait for invocation (SWS_Dcm_00402) RTE</stoproutine></stoproutine>	the
Step 8 [RUN <stoproutine>] "currentDataLength" is equal to Note the received arguments "dataIn1" is equal to "DATA3"</stoproutine>	13,
(SWS_Dcm_00402)	

	Receive UDS response for service "RoutineControl" from SUT and check the received "RoutineStatusRecord" (SWS_Dcm_00403)	"RoutineStatusRecord" is of size N4 and has the content equal to "DATA4"
·	[LT <diagnostic>] Send UDS request "RoutineControl" with sub-function "RequestRoutineResults" and RID=0x0201</diagnostic>	
Step 12	[RUN <requestresults>] Wait for invocation (SWS_Dcm_00404)</requestresults>	[RUN <requestresults>] is called by the RTE</requestresults>
·	[RUN <requestresults>] Return "dataOut1="DATA5", "currentDataLength" = N5, 0 as error code and E_OK as return value</requestresults>	
·	[LT <diagnostic>] Receive UDS response for service "RoutineControl" from SUT and check the received "RoutineStatusRecord" [SWS_Dcm_00405]</diagnostic>	Positive response code received. "RoutineStatusRecord" is of size N5 and has the content equal to "DATA5"
Post- conditions	None	