

Clarification Request

References: BTL TP 23.1, 135.1-2019 Clauses 8.4.17.10 and 8.5.17.10.

Date of BTL-WG Response: 2023-08-31

Background:

Tests 8.4.17.10 and 8.5.17.10 require the IUT to be able to generate offnormal and FAULT events.

5.2 Alarm and Event Management - Notification - Internal - B

5.2.1 Base Requirements

Base requirements must be met by any IUT claiming conformance to this BIBB.

...		
BTL - 8.5.17.10 - After FAULT-to-NORMAL, Re-Notification of OFFNORMAL (UnconfirmedEventNotifications)		
	Test Conditionality	If the IUT has no object in which CHANGE_OF_RELIABILITY is implemented in an object that can be configured into an offnormal state, this test shall be skipped.
	Test Directives	The objects selected by the tester should include all variants that differ in the set of supported alarming properties, or the writability of any of those properties. At least one instance of each variant shall be selected.
	Testing Hints	

5.3 Alarm and Event Management - Notification - External - B

5.3.1 Base Requirements

Base requirements must be met by any IUT claiming conformance to this BIBB.

135.1-2019 - 8.4.17.10 - After FAULT-to-NORMAL, Re-Notification of OFFNORMAL (ConfirmedEventNotifications)		
	Test Conditionality	If the IUT does not support an Event Enrollment object which is capable of generating OFFNORMAL transitions, this test shall be skipped.
	Test Directives	Execute test using an Event Enrollment object monitoring an object (O1) in a device other than the IUT.
	Testing Hints	
BTL - 8.5.17.10 - After FAULT-to-NORMAL, Re-Notification of OFFNORMAL (UnconfirmedEventNotifications)		
	Test Conditionality	If the IUT does not support an Event Enrollment object which is capable of generating OFFNORMAL transitions, this test shall be skipped.
	Test Directives	Execute test using an Event Enrollment object monitoring an object (O1) in a device other than the IUT.
	Testing Hints	

8.4.17.10 After FAULT-to-NORMAL, Re-Notification of OFFNORMAL (ConfirmedEventNotifications)

Purpose: To verify that objects go to the NORMAL state after leaving the FAULT state, then transition to OFFNORMAL if the condition still exists.

Test Concept: The test concept corresponds to 8.5.17.10

Configuration Requirements: The configuration requirements are identical to those in 8.5.17.10, except that the 'Issue Confirmed Notifications' parameter shall have a value of TRUE.

Test Steps: The test steps for this test case are identical to the test steps in 8.5.17.10, except that the UnconfirmedEventNotification requests are ConfirmedEventNotification requests and the TD acknowledges receiving the notifications.

Notes to Tester: The passing results for this test case are identical to the ones in 8.5.17.10, except that the event notifications shall be conveyed using a ConfirmedEventNotification service request.

8.5.17.10 After FAULT-to-NORMAL, Re-Notification of OFFNORMAL (UnconfirmedEventNotifications)

Reason for Change: Fix the Status_Flags in step 7. Added verify of pCurrentState after each transition. Add steps to wait pTimeDelay time delay before verifying pCurrentState when the state transitions from NORMAL to OFFNORMAL.

Purpose: To verify that objects go to the NORMAL state after leaving the FAULT state, then transition to OFFNORMAL if the condition still exists.

Test Concept: Select a fault detecting object O1 which is able to detect OFFNORMAL conditions. Make O1 transition to an OFFNORMAL state and then transition to FAULT. Remove the condition causing the FAULT and verify O1 transitions from FAULT to NORMAL, then verify that the object transitions from NORMAL to the original OFFNORMAL state.

Configuration Requirements: O1 is configured to detect and report unconfirmed events and faults. O1 is configured to have no fault conditions present, and Event_State is NORMAL. The 'Issue Confirmed Notifications' parameter shall have a value of FALSE.

Test Steps:

1. VERIFY pCurrentReliability = NO_FAULT_DETECTED
2. VERIFY pCurrentState = NORMAL
3. MAKE(O1 transition to an off normal state)
4. *WAIT pTimeDelay*
54. BEFORE Notification Fail Time

RECEIVE UnconfirmedEventNotification-Request

'Process Identifier' =	(any valid process identifier),
'Initiating Device Identifier' =	IUT,
'Event Object Identifier' =	O1,
'Time Stamp' =	(any valid time stamp),
'Notification Class' =	(the notification class configured for O1),
'Priority' =	(the value configured for the transition),
'Event Type' =	(ET1, any valid off normal event type),
'Message Text' =	(optional, any valid message text),

'Notify Type' =	ALARM EVENT,
'AckRequired' =	TRUE FALSE,
'From State' =	NORMAL,
'To State' =	OFFNORMAL,
'Event Values' =	(property-values appropriate for O1)

65. VERIFY pCurrentState = OFFNORMAL

76. MAKE(O1 enter a fault state)

87. BEFORE **Notification Fail Time**

RECEIVE UnconfirmedEventNotification-Request

'Process Identifier' =	(any valid process identifier),
'Initiating Device Identifier' =	IUT,
'Event Object Identifier' =	O1,
'Time Stamp' =	(the current local time or sequence number),
'Notification Class' =	(the notification class configured for O1),
'Priority' =	(the value configured for the transition),
'Event Type' =	CHANGE_OF_RELIABILITY,
'Message Text' =	(optional, any valid message text),
'Notify Type' =	ALARM EVENT,
'AckRequired' =	TRUE FALSE,
'From State' =	OFFNORMAL,
'To State' =	FAULT,
'Event Values' =	((R1 any valid BACnetReliability), (?T, T, ?, ?), (A list of valid values for properties required to be reported for O1, and 0 or more other properties of O1))

98. MAKE(O1 clear the fault condition)

109. BEFORE **Notification Fail Time**

RECEIVE UnconfirmedEventNotification-Request

'Process Identifier' =	(any valid process identifier),
'Initiating Device Identifier' =	IUT,
'Event Object Identifier' =	O1,
'Time Stamp' =	(TSI: any valid time stamp),
'Notification Class' =	(the notification class configured for O1),
'Priority' =	(the value configured for the transition),
'Event Type' =	CHANGE_OF_RELIABILITY,
'Message Text' =	(optional, any valid message text),
'Notify Type' =	ALARM EVENT,
'AckRequired' =	TRUE FALSE,
'From State' =	FAULT,
'To State' =	NORMAL,
'Event Values' =	(NO_FAULT_DETECTED, (F, F, ?, ?), (A list of valid values for properties required to be reported for O1, and 0 or more other properties of O1))

1140. VERIFY pCurrentReliability = NO_FAULT_DETECTED

12. VERIFY pCurrentState = NORMAL

13. WAIT until TSI + pTimeDelay

1442. BEFORE **Notification Fail Time**

RECEIVE UnconfirmedEventNotification-Request

'Process Identifier' =	(any valid process identifier),
'Initiating Device Identifier' =	IUT,
'Event Object Identifier' =	O1,

'Time Stamp' =	(the current local time or sequence number),
'Notification Class' =	(the notification class configured for O1),
'Priority' =	(the value configured for the transition),
'Event Type' =	ET1,
'Message Text' =	(optional, any valid message text),
'Notify Type' =	ALARM EVENT,
'AckRequired' =	TRUE FALSE,
'From State' =	NORMAL,
'To State' =	OFFNORMAL,
'Event Values' =	(property-values appropriate for O1)

15. *VERIFY pCurrentReliability = NO_FAULT_DETECTED*

16. *VERIFY pCurrentState = OFFNORMAL*

Problems #1:

8.4.17.10 is missing in AE-N-I-B.

Problems #2:

For AE-N-I-B, 8.5.17.10 is allowed to be skipped if the IUT does not support either offnormal or FAULT events. For AE-N-E-B, these tests are not allowed to be skipped if the IUT does not support FAULT events.

Questions:

1. Should 8.4.17.10 be added to Clause 5.2.1 of AE-N-I-B of the Test Plan?
2. Tests 8.4.17.10 and 8.5.17.10 are required only if the IUT's Event Enrollment object supports both offnormal and FAULT eventing?

Response:

1. Yes
2. Yes