BACnet® TESTING LABORATORIES
ADDENDA

Addendum bu to
BTL Test Package 20.0.1

Revision v3
Revised 10/27/2022

Approved by the BTL Working Group on 2022-10-03.
Approved by the BTL Working Group Voting Members on 2022-10-27.
Published on 2022-10-28.

[This foreword and the "Overview" on the following pages are not part of this Test Package. They are merely informative and do not contain requirements necessary for conformance to the Test Package.]

FOREWORD

The purpose of this addendum is to present current changes being made to the BTL Test Package. These modifications are the result of change proposals made pursuant to the continuous maintenance procedures and of deliberations within the BTL-WG Committee. The changes are summarized below.

In the following document, language to be added to existing clauses within the BTL Test Package 20.0.1 is indicated through the use of *italics*, while deletions are indicated by ~~strikethrough~~. Where entirely new subclauses are proposed to be added, plain type is used throughout

In contrast, changes to BTL Specified Tests also contain a yellow highlight to indicate the changes made by this addendum. When this addendum is applied, all highlighting will be removed. Change markings on tests will remain to indicate the difference between the new test and an existing 135.1 test. If a test being modified has never existed in 135.1, the applied result should not contain any change markings. When this is the case, square brackets will be used to describe the changes required for this test.

Each addendum can stand independently unless specifically noted via dependency within the addendum. If multiple addenda change the same test or section, each future released addendum that changes the same test or section will note in square brackets whether or not those changes are reflected.

BTL-20.0.1 bu-1: Clarification on UnconfirmedCOVNotification Service Initiation [BTLWG-782]

Overview:

135-2016bu-3 provided clarification of unconfirmed COV notifications:
... since the subscriber device is known, the UnconfirmedCOVNotification or UnconfirmedCOVNotificationMultiple shall be transmitted as a unicast to the subscriber device.

Changes:

Checklist Changes

None

Test Plan Changes

None

Specified Test Changes

[In BTL Specified Tests, modify the existing tests 8.3.1 through 8.3.3, as shown]

8.3 UnconfirmedCOVNotification Service Initiation Tests

8.3.1 Change of Value Notification *for changes to* Present_Value Property *in Objects with a COV_Increment*
Reason for Change: Addendum 135-2008w-1, and 135-2-i-1 Add more primitive value objects and the Lighting Output Object. Add clarification to test the last COVNotification shall reflect the correct values. Addendum 135-2016bu-3 clarified COV notification shall always be unicast

Purpose: To verify that the IUT can initiate UnconfirmedCOVNotification service requests conveying a change of the Present_Value property ~~of Analog Input, Analog Output, and Analog Value, objects~~.

Test Steps: The steps for this test case are identical to the test steps in 8.2.1 except that the SubscribeCOV service request in step 1 shall have a value of FALSE for the 'Issue Confirmed Notifications' parameter, all of the ConfirmedCOVNotification requests shall be UnconfirmedCOVNotification *unicasts* ~~requests~~, and there is no acknowledgment of the unconfirmed services. ~~The MAC address used for the notification message shall be such that the TD is one of the recipients.~~

Notes to Tester: The IUT may initiate additional COVNotifications. The final COVNotification shall accurately reflect Present_Value and Status_Flags.

8.3.2 Change of Value Notification for Changes to Status_Flags Property

Reason for Change: Addendum 135-2008w-1 Add more primitive value objects. Addendum 135-2016bu-3 clarified COV notification shall always be unicast

Purpose: To verify that the IUT can initiate UnconfirmedCOVNotification service requests conveying a change of the Status_Flags property ~~of Analog Input, Analog Output, and Analog Valueobjects~~.

Test Steps: The steps for this test case are identical to the test steps in 8.2.2 except that the SubscribeCOV service request in step 1 shall have a value of FALSE for the 'Issue Confirmed Notifications' parameter, all of the ConfirmedCOVNotification requests shall be UnconfirmedCOVNotification *unicasts* ~~requests~~, and there is no acknowledgment of the unconfirmed services. ~~The MAC address used for the notification message shall be such that the TD is one of the recipients.~~

8.3.3 Change of Value Notification for Changes to Present_Value Property in Objects without a COV_Increment

Reason for Change: Renamed test. Add clarification to test that the last COVNotification shall reflect the correct values. Addendum 135-2016bu-3 clarified that COV notification shall always be unicast

Purpose: To verify that the IUT can initiate UnconfirmedCOVNotification service requests conveying a change of the Present_Value property ~~of Binary Input, Binary Output, and Binary Value objects~~ *of objects that do not support COV_Increment*.

Test Steps: The steps for this test case are identical to the test steps in 8.2.3 except that the SubscribeCOV service request in step 1 shall have a value of FALSE for the 'Issue Confirmed Notifications' parameter, all of the ConfirmedCOVNotification requests shall be UnconfirmedCOVNotification *unicasts* ~~requests~~, and there is no acknowledgment of the unconfirmed services. ~~The MAC address used for the notification message shall be such that the TD is one of the recipients.~~

Notes to Tester: The IUT may initiate additional COVNotifications. The final COVNotification shall accurately reflect Present_Value and Status_Flags.

[In BTL Specified Tests, modify the existing test 9.X41.1.4, as shown]
9.X41.1.4 Unconfirmed Change of Value Notification From Property Value

Reason for Change: Added new test to support DS-COVM-B testing. Addendum 135-2016bu-3 clarified that COV notification shall always be unicast

Purpose: To verify that the IUT initiates an UnconfirmedCOVMultipleNotification service request when a subscribed to property changes.

Test Concept: A COVM subscription is made which contains a subscription *with 'Issue Confirmed Notifications' = FALSE* to property P1 in object O1. The value of P1 is changed and it is verified that the IUT *sends* ~~send~~ a *unicast* COVM notification.

Test Steps:

[No Changes]

BTL-20.0.1 bu-2: Clarify the Global Group Object Reliability Evaluation [BTLWG-786]

Overview:

135-2016bu7 Clarify the Global Group object reliability evaluation

Changes:

Checklist Changes

None

Test Plan Changes

[Update conditionality and directives for BTL - 7.3.2.13.X3 and add a new test to the Global Group object Base Requirements section]

3.36.1 Base Requirements

Base requirements must be met by any IUT that can contain Global Group objects.

BTL - 7.2.3 - Read-only Property Test		
	Test Conditionality	Must be executed.
	Test Directives	Test the Present_Value property of each Global Group object.
	Testing Hints	
BTL - 7.3.2.13.X2 - Reliability MEMBER_FAULT Test		
	Test Conditionality	If no object pointed to by the Group_Members property can be made to contain Status_Flags FAULT flag equal to TRUE, then this test shall be skipped.
	Test Directives	
	Testing Hints	
BTL - 7.3.2.13.X3 - Reliability COMMUNICATION_FAILURE Test		
	Test Conditionality	If the Groups_Members property is not writable and can not be made to contain references to external objects, this test shall be skipped.
	Test Directives	~~Repeat this test with a member of the Group_Members property pointing to a device not communicating and with a member of the Group_Members property pointing to a device responding with a BACnet-Error-PDU.~~
	Testing Hints	
BTL - 7.3.2.13.X4 - Present_Value Tracking and Reliability Test		
	Test Conditionality	If the Reliability property is not present or can not be made to not equal NO_FAULT_DETECTED, this test shall be skipped.
	Test Directives	The test shall be executed using a Global Group object.
	Testing Hints	
BTL - 7.3.2.13.X7 - First Stage Faults Take Precedence Over Second Stage Faults When Presenting Reliability		
	Test Conditionality	*If no object pointed to by the Group_Members property can be made to contain Status_Flags FAULT flag equal to TRUE, or if the Group_Members property is not writable and can not be made to contain references to external objects, this test shall be skipped.*
	Test Directives	
	Testing Hints	

Specified Test Changes

[Modify BTL Specified Test 7.3.2.13.X3]

7.3.2.13.X3 Reliability COMMUNICATION_FAILURE Test
Reason for Change: New Tests for Global Group object type. *COMMUNICATION_FAILURE does not include errors returned by an object.*

Purpose: This test case verifies that the Member_Status_Flags FAULT flag will remain FALSE while the Reliability property is COMMUNICATION_FAILURE.

Test Concept: Force a member of the Group_Members property to stop communicating and verify the Reliability property equals COMMUNICATION_FAILURE and the Member_Status_Flags FAULT flag remains FALSE.

Configuration Requirements: The IUT shall be configured with a Global Group object with the Group_Members containing a member M1 at index N1that can be made to discontinue communications ~~and also respond with an error such as OBJECT/UNKNOWN_OBJECT~~. ~~(See Notes To Tester).~~ The Out_Of_Service property of the Global Group object must remain FALSE throughout the test. W1 is the maximum time it takes for the Global Group to receive an update from M1.

Notes to Tester: Reliability will change to COMMUNICATION_FAILURE when a member is no longer able to communicate its Status_Flags property. This can occur when the device goes offline ~~or the object is deleted within the device~~.

Test Steps:

1. MAKE (M1 ~~fail (communications or error)~~ *discontinue communications*)
2. WAIT (W1)
3. VERIFY Reliability = COMMUNICATION_FAILURE
4. IF (Reliability is present) THEN
   VERIFY Reliability = COMMUNICATION_FAILURE
5. VERIFY Member_Status_Flags = {?, FALSE, ?, ?}

[Add new test into BTL Specified Tests]

7.3.2.13.X7 - First Stage Faults Take Precedence Over Second Stage Faults When Presenting Reliability
Reason for Change: No longer an internal matter which Reliability is presented, COMMUNICATION_FAILURE takes precedence for all objects, 135-2016bu-7.

Purpose: This test verifies that the Reliability and FAULT flags will reflect a COMMUNICATION_FAILURE, even when the conditions for a MEMBER_FAULT are also present in the Global Group Object.

Test Concept: Force a member of the Group_Members property to stop communicating and verify the Reliability property equals COMMUNICATION_FAILURE and the Member_Status_Flags FAULT flag remains FALSE. Then force a different member of the Group_Members property to enter a Fault condition and verify the Member_Status_Flags FAULT flag equals TRUE and Reliability equals COMMUNICATION_FAILURE. The Fault conditions are then removed and reapplied in the inverse order, and the Member_Status_Flags FAULT flag and Reliability properties are verified again.

Configuration Requirements: The IUT shall be configured with a Global Group object, O1, with the Group_Members property containing a member M1 at index N1 that can be made to discontinue communications. O1's Group_Members property shall also contain a member M2 at index N2 that can be made to indicate a fault condition. The IUT begins the test with Reliability equal to NO_FAULT_DETECTED. The Out_Of_Service property of the Global Group object must remain FALSE throughout the test. W1 is the maximum time it takes for the Global Group object to receive an update from M1. W2 is the maximum time it takes for the Global Group object to receive an update from M2. The test steps begin with M1 communicating with O1 and M2 with its FAULT flag cleared.

Test Steps:

1. MAKE (M1 discontinue communications)
2. WAIT (W1)
3. IF (Reliability is present) THEN
   VERIFY Reliability = COMMUNICATION_FAILURE
4. VERIFY Member_Status_Flags = {?, FALSE, ?, ?}
5. MAKE (M2 go into a fault condition)
6. WAIT (W2)
7. VERIFY M2.Status_Flags = {?, TRUE, ?, ?}
8. IF (Reliability is present) THEN
   VERIFY Reliability = COMMUNICATION_FAILURE

9. VERIFY Member_Status_Flags = {?, TRUE, ?, ?}
10. MAKE (M1 resume communications)
11. MAKE (M2 come out of the fault condition)
12. WAIT (the greater of W1 and W2)
13. VERIFY M2.Status_Flags = {?, FALSE, ?, ?}
14. If (Reliability is present) THEN
    VERIFY Reliability = NO_FAULT_DETECTED
15. MAKE (M2 go into a fault condition)
16. WAIT (W2)
17. VERIFY M2.Status_Flags = {?, TRUE, ?, ?}
18. If (Reliability is present) THEN
    VERIFY Reliability = MEMBER_FAULT
19. VERIFY Member_Status_Flags = {?, TRUE, ?, ?}
20. MAKE (M1 discontinue communications)
21. WAIT (W1)
22. If (Reliability is present) THEN
    VERIFY Reliability = COMMUNICATION_FAILURE
23. VERIFY Member_Status_Flags = {?, TRUE, ?, ?}

BTL-20.0.1 bu-3: Add Testing for Reading of List Property with ReadRange [BTLWG-1178]

Overview:

135-2016-bu1 “Introduce BACnetARRAY of BACnetLIST collection property data type” introduces a new datatype “BACnetARRAY of BACnetLIST of <datatype>”.

Additionally, were two new Error Codes added to the ReadRange Service (15.8.1.3.1) / (18.3.).

Only testing of the new error codes for the ReadRange service is added in this proposal as there are no properties yet which are BACnetARRAY of BACnetLIST

Changes:

Checklist Changes

None

Test Plan Changes

[Add the following two tests to “Execute ReadRange” Base Requirements]

4.16.1 Base Requirements

Base requirements must be met by any IUT that executes ReadRange.

…		
BTL - 9.21.2.X7 - Attempting to Read a List Property by Sequence That Does not Have Sequence Numbers		
	Test Conditionality	*Must be executed if the IUT claims Protocol_Revision 21 or greater and supports a List Property that supports Timestamps.*
	Test Directives	
	Testing Hints	
BTL - 9.21.2.X8 - Attempting to Read a List Property by ByTime That Does not Have Timestamps		
	Test Conditionality	*Must be executed if the IUT claims Protocol_Revision 21 or greater and supports a List Property that supports Sequence Numbers.*
	Test Directives	
	Testing Hints	

Test Changes

[Add a new tests into BTL Specified Tests Section “9.21.2 Negative ReadRange Service Execution Tests”]

9.21.2.X7 - Attempting to Read a List Property by Sequence That Does not Have Sequence Numbers
Reason for Change: No test exists for this functionality. Tests added as per 135-2016-bu1

References: 15.8.1.3.1, 18.3

Purpose: To verify the correct execution of the ReadRange service request when the requested property does not support sequence numbers.

Test Concept: A ReadRange request is transmitted by the TD requesting a specified sequence number and count of items. The IUT shall respond by returning the appropriate error code. This test is only applied to devices with a Protocol_Revision of 21 or higher.

Test Configuration: A list property that does not support sequence numbers must be selected for this test. If no suitable property exists in the device, this test shall be skipped.

Test Steps:

1. TRANSMIT Read-Range-Request,
   'Object Identifier' = (the object configured for this test),
   'Property Identifier' = (the list property configured for this test),
   'Reference Sequence Number' = 42,
   'Count' = (any valid value)
2. RECEIVE BACnet-Error-PDU,
   'Error Class' = PROPERTY,
   'Error Code’ = LIST_ITEM_NOT_NUMBERED

9.21.2.X8 - Attempting to Read a List Property by ByTime That Does not Have Timestamps
Reason for Change: No test exists for this functionality. Tests added as per 135-2016-bu1

References: 15.8.1.3.1, 18.3

Purpose: To verify the correct execution of the ReadRange service request when the requested property does not support timestamps.

Test Concept: A ReadRange request is transmitted by the TD requesting a specified reference time and count of items. The IUT shall respond by returning the appropriate error code. This test is only applied to devices with a Protocol_Revision of 21 or higher.

Test Configuration: A list property that does not support timestamps must be selected for this test. If no suitable property exists in the device, this test shall be skipped.

Test Steps:

1. TRANSMIT Read-Range-Request,
   'Object Identifier' = (the object configured for this test),
   'Property Identifier' = (the list property configured for this test),
   'Reference Time' = (any valid specific BACnetDateTime)
   'Count' = (any valid value)
2. RECEIVE BACnet-Error-PDU,
   'Error Class' = PROPERTY,
   'Error Code’ = LIST_ITEM_NOT_TIMESTAMPED