BACnet® TESTING LABORATORIES
ADDENDA

Addendum cr1 to BTL Test Package 20.0.1

Revision v4
Revised 11/8/2022

Approved by the BTL Working Group on2022-10-03.
Approved by the BTL Working Group Voting Members on 2022-11-07.
Published on 2022-11-10.

[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 cr1-1: Clarify AddListElement Failure Test [BTLWG-604, CR-0398]

Overview:

With CR-0398 it was decided that INVALID_DATATYPE errors need additional directions for the choice of data used.

Changes:

Checklist Changes

None

Test Plan Changes

None

Test Changes

[Modify test 9.14.2.3 in BTL Specified Tests]

9.14.2.3 An AddListElement Failure Part Way Through a List
Reason For Change: Updated test to include additional error codes. Added 'Notes to Tester' which was missing in 135.1-2013. Modified Notes to Tester to reflect the directions given in CR-0398 to the choice of List Elements used in step 2.

Purpose: To verify the ability of the IUT to respond to an AddListElement service request to add multiple elements to a list where one of the elements cannot be added. Upon failure, the AddListElement service should leave the list unchanged.

Notes to Tester: Value selected for step 2~~3~~ is 'inappropriate', not a value which is 'allowed' but not supported by this instance of the property. i.e. it is not one of the datatypes that would ever be supported by an instance of this property in this object type. DATATYPE_NOT_SUPPPORTED is only correct when the datatype requested is supported, for example in a CHOICE, by this property in this object type, but not supported by this instance of the property. Care must be taken, that the inserted error is not likely to cause confusion in the IUT as to which element contains the error.

Test Steps:

```
1.  READ InitialList = (L), ListProp2
2.  TRANSMIT AddListElement-Request,
        'Object Identifier' =       L,
        'Property Identifier' =     ListProp
        'List of Elements' =        (two or more elements to be added to the list with the second element
                                    having an inappropriate datatype)
3.  IF (Protocol_Revision is present and Protocol_Revision >= 7) THEN
        RECEIVE AddListElement-Error,
            'Error Class'  =            PROPERTY,
            'Error Code'  =             INVALID_DATATYPE,
            'First Failed Element' =    2
        | (BACnet-Reject-PDU,
            Reject Reason = INVALID_TAG | INVALID_PARAMETER_DATA_TYPE)
    ELSE
        RECEIVE AddListElement-Error,
            'Error Class'  =            SERVICES,
            'Error Code'  =             INVALID_PARAMETER_DATATYPE
            'First Failed Element' =    2
        | (AddListElement-Error,
            'Error Clas's =             PROPERTY,
            *'Error Code' =              INVALID_DATATYPE)*
            *'First Failed Element' =    2*
        | *(BACnet-Reject-PDU,*
            *Reject Reason = INVALID_TAG | INVALID_PARAMETER_DATA_TYPE)*
4.  VERIFY (L), ListProp =          InitialList
```

BTL-20.0.1 cr1-2: Fix Error in Tests for Logging ACK_NOTIFICATION [BTLWG-611]

Overview:

Fix 7.3.2.27 Internal Logging of ACK_NOTIFICATIONs wrong record count calculation.

Changes:

Checklist Changes

None

Test Plan Changes

None
[Changes noted in this work item have already been applied in TP 20.0.1

Test Changes

[Modify existing test 7.3.2.25.3]

7.3.2.25.3 Internal Logging of ACK_NOTIFICATIONs

Reason for Change: Removed unused step #2 and fixed the Result Flags in step 11.

Purpose: To verify the IUT correctly collects and represents an ACK_NOTIFICATION which it initiates.

Test Concept: Make the IUT generate an ACK_NOTIFICATION message. Use ReadRange to retrieve that same event from an Event Log and compare the two representations. If the IUT does not support logging of the ACK_NOTIFICATIONs which it initiates, this test shall be skipped.

Configuration Requirements: O1 is an event initiating object in the IUT, which is configured to send event notifications to the TD. LO1 is an Event Log object in the IUT which logs ACK_NOTIFICATIONs.

Test Steps:

1. WRITE Enable = TRUE
~~2. READ RC = LO1, Record_Count~~
2~~3~~. MAKE (a condition exist that will cause the object to generate an event transition)
~~4. WAIT D1~~
3~~5~~. RECEIVE ConfirmedEventNotification-Request,

'Process Identifier' = (PI1, any valid process identifier),
'Initiating Device Identifier' = IUT,
'Event Object Identifier' = O1,
'Time Stamp' = (T1, any valid timestamp),
'Notification Class' = (N1, any valid notification class),
'Priority' = (P1, any valid priority),
'Event Type' = (ET1, any standard event type),
'Message Text' = (optional, any valid message text),
'Notify Type' = ALARM | EVENT,
'AckRequired' = TRUE | FALSE,
'From State' = (S1, any valid state for this event type),
'To State' = (S2, any valid state for this event type),
'Event Values' = (any values appropriate to the event type)

4~~6~~. TRANSMIT BACnet-SimpleACK-PDU
5~~7~~. TRANSMIT AcknowledgeAlarm-Request,

'Acknowledging Process Identifier' = (any valid value),

```
        'Event Object Identifier' =                 O1,
        'Event State Acknowledged' =                S2,
        'Time Stamp' =                              T1,
        'Acknowledgement Source' =                  (any valid value),
        'Time of Acknowledgment' =                  (the current time)
*6*~~8~~. RECEIVE BACnet-SimpleACK-PDU
*7*~~9~~. BEFORE **Notification Fail Time**
        RECEIVE ConfirmedEventNotification-Request,
            'Process Identifier' =                  PI1,
            'Initiating Device Identifier' =        IUT,
            'Event Object Identifier' =             O1,
            'Time Stamp' =                          (T2, any valid timestamp > T1),
            'Notification Class' =                  N1,
            'Priority' =                            P1,
            'Event Type' =                          ET1,
            'Message Text' =                        (optional, any valid message text),
            'Notify Type' =                         ACK_NOTIFICATION,
            'From State' =                          S1
*10*~~10~~.   TRANSMIT BACnet-SimpleACK-PDU
*11.  READ RC = LO1, Record_Count*
*12*~~11~~.   TRANSMIT ReadRange-Request,
        'Object Identifier' =                       LO1,
        'Property Identifier' =                     Log_Buffer,
        ‘Reference Index’ =                         RC,
        ‘Count’ =                                   -1
*13*~~12~~.   RECEIVE ReadRange-ACK,
        'Object Identifier' =                       LO1,
        'Property Identifier' =                     Log_Buffer,
        'Result Flags' =                            {~~FALSE~~*?*, ?, *FALSE*~~TRUE~~},
        'Item Count' =                              1,
        'Item Data' =                               (logged data that matches the information received in step *7*4,
                                                    except that Process_Identifier can be any value and is not
                                                    required to match)
```

Notes to Tester: When the UnconfirmedEventNotification service is used instead of the ConfirmedEventNotification service, the TD shall skip the steps in which BACnet-SimpleACK-PDUs are sent in response to ConfirmedEventNotifications.

BTL-20.0.1 cr1-3: Update Test 8.5.17.10 to use pTimeDelay [BTLWG-714, CR-0446]

Overview:

Based on BTL-CR-0446, this test needs to be updated to wait pTimeDelay after transitioning from FAULT to NORMAL before making a transition from NORMAL to OFFNORMAL.

Changes:

Checklist Changes

None

Test Plan Changes

None

Test Changes

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

Reason for Change: Fix the Status_Flags in step 8. 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(O1transition to an off normal state)

4. WAIT pTimeDelay

45. 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)

~~5~~*6*. VERIFY pCurrentState = OFFNORMAL
~~5~~*7*. MAKE(O1 enter a fault state)
~~6~~*8*. 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~~*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))

~~7~~*9*. MAKE(O1 clear the fault condition)
~~8~~*10*. BEFORE **Notification Fail Time**
RECEIVE UnconfirmedEventNotification-Request
'Process Identifier' = (any valid process identifier),
'Initiating Device Identifier' = IUT,
'Event Object Identifier' = O1,
'Time Stamp' = (*TS1:* 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))

~~9~~*11*.VERIFY pCurrentReliability = NO_FAULT_DETECTED
12. VERIFY pCurrentState = NORMAL
13. WAIT until TS1 + pTimeDelay
~~10~~*14*. 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

BTL-20.0.1 cr1-4: Clarify Test Forwarding Router-Busy-To-Network [BTLWG-717, CR-0453]

Overview:

If there are no DNETs connected to the router, the router can respond without the ‘Network Number’ field in Router-Busy-To-Network.

In CRR 0453, it was agreed to allow absence of ‘Network Number’ field in the test 10.2.2.4.2 if the router does not have any DNETs attached to it.

Changes:

Checklist Changes

None

Test Plan Changes

[Modify Test Plan 10.1.2]

10.1.2Routes Packets between Physical BACnet LANs

...

...

BTL**~~135.1-2019~~ - 10.2.2.4.2 - Forwarding Router-Busy-To-Network Information for all DNETs**		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	

Test Changes

[Modify 135.1-2019 - 10.2.2.4.2 : This is test is being placed in BTL Specified Test document and modified as below:]

10.2.2.4.2 Forwarding Router-Busy-To-Network Information for all DNETs

Reason for Change: Test step 2 is modified to allow absence of “Network Number” field.

Purpose: To verify that the IUT correctly forwards information indicating that all DNETs reachable through a particular router are temporarily unreachable because of traffic congestion.

Test Steps:

1. TRANSMIT PORT B,
   DESTINATION = LOCAL BROADCAST,
   SOURCE = R2-3,
   Router-Busy-To-Network
2. RECEIVE PORT A,
   DESTINATION = LOCAL BROADCAST,
   SOURCE = IUT,
   Router-Busy-To-Network,
   Network Numbers = 3,6 | 6,3 | *(absent)*

BTL-20.0.1 cr1-5: Add Test Directives for Message Text Entry in AE-ACK-B [BTLWG-838, CR-402]

Overview:

In response to CR-0402, the BTL-WG agreed to add test directives to tests 9.1.4.X1/9.1.5.X1 to minimize the number of times the tests are applied.

9.1.1.15 - Unsupported Message Text Character Set Acknowledge Alarm Test => TP18.0(renumbered from 9.l1.X1)

Changes:

Checklist Changes

None

Test Plan Changes

5.5 Alarm and Event Management - Acknowledge - B

[Modify the Test Directives section Base Requirements]

5.5.1 Base Requirements

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

135.1-2019 - 9.1.1.15 - Unsupported Message Text Character Set Acknowledge Alarm Test		
	Test Conditionality	If the IUT supports all character sets, this test shall be skipped.
	Test Directives	Repeat for every unsupported character set. Use at least one ConfirmedEventNotification and one UnconfirmedEventNotification.
	Testing Hints	

Test Changes

None

BTL-20.0.1 cr1-6: Clarify Loop Out_Of_Service Usage in Test Plan [BTLWG-842, CR-0384]

Overview:

The Out_of_Service property in Loop object has addendum history. Having for years been presumed to behave like in other objects, an earlier BTL-CR brought to our attention that the standard did not state that Out_of_Service in Loop must behave like in other objects. So BTL removed the Out_of_Service in Loop test.

In 135-2010ad-6, PR13, added language to mandate Out_of_Service in Loop behave like other objects. So BTL can, with a Protocol_Revision conditional, add back the Out_of_Service testing in Loop.

Changes:

Checklist Changes

None

Test Plan Changes

3.13 Loop

3.13.2 Supports Writable Out_Of_Service Property

The Out_Of_Service property in Loop objects contained in the IUT are writable.

BTL - 7.3.1.1.X1 - Out_Of_Service, Status_Flags, and Reliability Tests		
	Test Conditionality	~~Must be executed.~~ *If the IUT implements Protocol_Revision 13 or higher, this test must be executed. If the IUT implements a Protocol_Revision less than 13 and does not support writable Present_Value and Reliability properties when Out_Of_Service is TRUE, this test shall be skipped.*
	Test Directives	
	Testing Hints	

BTL-20.0.1 cr1-7: Update Test Conditionality for DS-COV-A Testing [BTLWG-906, CR-0486]

Overview:

CR-0486 indicated that the Test Conditionality change in Test Plan for DS-COV-A tests for Change of Value Notifications for **135.1-2013 - 9.2.1.1 and 135.1-2013 - 9.3.2**

Changes:

Checklist Changes

None

Test Plan Changes

[In BTL Test Plan, make the following changes to the Test Conditionality]

4.9.10 Can Subscribe for COV from CharacterString Objects

The IUT can subscribe for, receive, and process Change of Value notifications from CharacterString Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type CharacterString.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type CharacterString.
	Testing Hints	

4.9.11 Can Subscribe for COV from Date Objects

The IUT can subscribe for, receive, and process Change of Value notifications from Date Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type Date.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type Date.
	Testing Hints	

4.9.12 Can Subscribe for COV from DateTime Objects

The IUT can subscribe for, receive, and process Change of Value notifications from DateTime Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type BACnetDateTime.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*

	Test Directives	Test one instance of each object type where Present_Value is of data type BACnetDateTime.
	Testing Hints	

4.9.13 Can Subscribe for COV from Integer Objects

The IUT can subscribe for, receive, and process Change of Value notifications from Integer Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type Integer.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type Integer.
	Testing Hints	

4.9.14 Can Subscribe for COV from Large Analog Objects

The IUT can subscribe for, receive, and process Change of Value notifications fromLarge Analog Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type Large Analog.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type Large Analog.
	Testing Hints	

4.9.15 Can Subscribe for COV from OctetString Objects

The IUT can subscribe for, receive, and process Change of Value notifications from OctetString Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type OctetString.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type OctetString.
	Testing Hints	

4.9.16 Can Subscribe for COV from Positive Integer Objects

The IUT can subscribe for, receive, and process Change of Value notifications from Positive Integer Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is using data type Unsigned, including the possible value zero.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is using data type Unsigned, including the possible value zero.
	Testing Hints	

4.9.17 Can Subscribe for COV from Time Objects

The IUT can subscribe for, receive, and process Change of Value notifications from Time Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type Time.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type Time.
	Testing Hints	

4.9.18 Can Subscribe for COV from Pulse Converter Objects

The IUT can subscribe for, receive, and process Change of Value notifications from Pulse Converter Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type REAL.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type REAL.
	Testing Hints	

...

4.9.20 Can Subscribe for COV from Load Control Objects

The IUT can subscribe for, receive, and process Change of Value notifications from Load Control Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type BACnetShedState.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type BACnetShedState.
	Testing Hints	

...

4.9.23 Can Subscribe for COV from Lighting Output Objects

The IUT can subscribe for, receive, and process Change of Value notifications from Lighting Output Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type REAL.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*

	Test Directives	Test one instance of each object type where Present_Value is of data type REAL.
	Testing Hints	

4.9.24 Can Subscribe for COV from Binary Lighting Output Objects

The IUT can subscribe for, receive, and process Change of Value notifications from Binary Lighting Output Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type BACnetBinaryLightingPV.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*
	Test Directives	Test one instance of each object type where Present_Value is of data type BACnetBinaryLightingPV.
	Testing Hints	

4.9.25 Can Subscribe for COV from Staging Objects

The IUT can subscribe for, receive, and process Change of Value notifications from Staging Objects.

135.1-2019 - 9.2.1.1 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.3.2 is executed.*
	Test Directives	Test one instance of each object type.
	Testing Hints	
135.1-2019 - 9.3.2 - Change of Value Notifications		
	Test Conditionality	~~Must be executed.~~ *This test can be skipped if 9.2.1.1 is executed.*
	Test Directives	Test one instance of each object type.
	Testing Hints	

BTL-20.0.1 cr1-8: Add Test Plan Base Requirements for T-VMT-A and T-VMMV-A [BTLWG-910]

Overview:

The checklist entries for these BIBBs place a requirement on product that select them, but this requirement is not tested by the test plan.

Changes:

Checklist Changes

None

Test Plan Changes

[Replace section 7.12.1 (Trending - Viewing and Modifying Trends - A) Base Requirements]

7.12.1 Base Requirements

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

VERIFY Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims support for T-V-A in the Checklist
	Testing Hints	

[Replace section 7.13.1 (Trending - Viewing and Modifying Multiple Values - A) Base Requirements]

7.13.1 Base Requirements

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

VERIFY Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims support for T-V-A in the checklist.
	Testing Hints	

Test Changes

None

BTL-20.0.1 cr1-9: Clarify Configuration Requirements for Test 7.3.2.9.2 [BTLWG-921, CR-0339]

Overview:

BTL-CR-0339 talks about the problem of failing step 3 of Test 7.3.2.9.2 if the device is fast in executing the command from step 1. The probability of failure is increased by the fact, that the VERIFY contains an implicit "wait InternalProcessingFailTime". The response for this clarification request was to add improvements to the configuration requirements of the test.

Changes:

Checklist Changes

None

Test Plan Changes

[Modify reference for 7.3.2.9.2 from 135.1-2019 to BTL in section 3.9.1 Command Object Base Requirements]

Test Changes

[Modify the Configuration Requirements of Test 7.3.2.9.2 in 135.1-2019]

7.3.2.9.2 Quit on Failure Test
Reason for Change: to ensure that the Command object is configured with delays to enable testing of In_Process.

Dependencies: ReadProperty Service Execution Tests, 9.18; WriteProperty Service Execution Tests, 9.22.
BACnet Reference Clause: 12.10.8.

Purpose: To verify that a Command object can successfully execute Quit_On_Failure procedures.

Test Concept: The IUT is configured with two action lists that include a sequence of externally visible outputs with a write somewhere in the sequence that will fail. The action lists are identical except that one has Quit_On_Failure set to TRUE and the other set to FALSE. The TD triggers both action lists. The external outputs are observed to verify that the failure procedures are properly implemented. If the IUT does not support action list configuration for this Test Concept, then this test shall be omitted.

Configuration Requirements: The IUT shall be configured with a Command object having at least two action lists, X and Y, that includes writing to a sequence of externally visible outputs. Somewhere in the sequence there shall be a write command that will fail that is followed by write commands that will succeed. Both action lists shall be identical except that list X shall have Quit_On_Failure set to TRUE, and Y shall have Quit_On_Failure set to FALSE. *Delays shall be configured into the Action, if necessary, sufficient to let the tester observe that In_Process equals TRUE while the action is in process. If no Command object can be so configured, then verification of In_Process being TRUE shall be skipped.*

[the test steps are renumbered without change marking to maintain clarity]

Test Steps:
1. WRITE Present_Value = X
~~2. RECEIVE Simple-ACK-PDU~~
2. VERIFY In_Process = TRUE
3. CHECK (for the externally visible actions and verify that they stop when the failure occurs)
4. VERIFY In_Process = FALSE,
5. VERIFY All_Writes_Successful = FALSE
6. WRITE Present_Value = Y
~~8. RECEIVE Simple-ACK-PDU~~

7. VERIFY In_Process = TRUE
8. CHECK (for the externally visible actions and verify that they continue to the end after the failure occurs)
9. VERIFY In_Process = FALSE,
10. VERIFY All_Writes_Successful = FALSE

BTL-20.0.1 cr1-10: Improve Directions for 8.20.5.1 and 8.20.5.2 [BTLWG-922, CR-0338]

Overview:

BTL Specified Tests 8.20.5.1 and 8.20.5.2 require that the IUT issues a RP request for each property in the original RPM. Relax both tests to permit a behavior that allows the IUT to omit RP requests for properties which are not essential to the operation being performed.

Changes:
[Fully describes the changes by example by specifying the document that requires changes and the changes needed]

Checklist Changes

None

Test Plan Changes

None

Test Changes

[Modify test 8.20.5.1 in BTL Specified Tests 20.0_final]

8.20.5.1 The IUT Determines the TD does not Support the ReadPropertyMultiple Service

Reason for Change: Modified test to allow multiple objects in addition to single objects. Notes to Tester added to permit a behavior that allows the IUT to omit RP requests for properties which are not essential to the operation being performed.

Purpose: Verifies the IUT's ability to automatically change its service choice from ReadPropertyMultiple to ReadProperty when the IUT determines the TD does not support the ReadPropertyMultiple service.

Test Concept: The IUT is configured in a manner that would normally cause it to access one or more properties in the TD via the ReadPropertyMultiple service. Prior to sending a ReadPropertyMultiple request, however, the IUT determines that the TD does not support the ReadPropertyMultiple service. The IUT instead attempts to access the TD's property values via the ReadProperty service (it is assumed that the IUT will make this determination by reading the TD's Protocol_Services_Supported property, but this test specifically does not attempt to verify this behavior).

Configuration Requirements: The TD is configured so that it does not support the ReadPropertyMultiple service. The IUT is configured such that it is ~~capable of~~ accessing one or more properties of a single *or multiple* object*s* in the TD via the ReadProperty and ReadPropertyMultiple services. ~~If the IUT cannot be configured in this way, then this test shall be omitted.~~

Notes to Tester: In step 3, observing a ReadProperty of the properties from Step 1 is what is expected, but it is acceptable if fewer properties are read with ReadProperty, if those which are omitted are not essential to the operation being performed.

Test Steps:

```
1.  MAKE (a condition in the IUT that would normally cause it to send a ReadPropertyMultiple request to the TD to
            access one or more propert*y*~~ies~~ values ~~of a single object~~)
2.  WAIT (a time interval specified by the vendor as sufficient for the IUT to determine that the TD does not support the
            ReadPropertyMultiple service)
3.  REPEAT X = (the properties that the IUT is to read) DO {
        RECEIVE ReadProperty-Request,
            'Object Identifier' =       (object identifier referenced by X),
            'Property Identifier' =     (property identifier referenced by X)
        TRANSMIT ReadProperty-Ack,
            'Object Identifier' =          (object identifier referenced by X),
            'Property Identifier' =        (property identifier referenced by X),
            'Property Value' =             (any valid value)
```

[Modify test 8.20.5.2 in BTL Specified Tests 20.0_final]

8.20.5.2 Fallback to ReadProperty on Reject - UNRECOGNIZED_SERVICE Response

Reason for Change: Modified to allow use of RPM and use of multiple objects. Notes to Tester added to permit a behavior that allows the IUT to omit RP requests for properties which are not essential to the operation being performed.

BACnet Reference Clauses: 15.5 and 15.7

Purpose: Verifies the IUT's ability to automatically change its service choice from ReadPropertyMultiple to ReadProperty when the TD returns a Reject-PDU and a Reject Reason of UNRECOGNIZED_SERVICE.

Test Concept: The IUT is configured to send the TD a ReadPropertyMultiple request to access one or more properties of one or more objects. The TD responds with a Reject-PDU and a Reject Reason of UNRECOGNIZED_SERVICE. With no additional configuration, the IUT sends one or more ReadProperty requests to the TD, where each ReadProperty request specifies an individual property from the original ReadPropertyMultiple request covering all the properties from the original ReadPropertyMultiple request.

Configuration Requirements: The TD is configured so that it does not support the ReadPropertyMultiple service. The IUT is configured such that it attempts to acquire values from the TD using the ReadPropertyMultiple service without first interrogating the TD’s Protocol_Services_Supported property. If the IUT cannot be configured in this way then this test shall be omitted.

Notes to Tester: In step 4, observing a ReadProperty of the properties from Step 1 is what is expected, but it is acceptable if fewer properties are read with ReadProperty, if those which are omitted are not essential to the operation being performed.

Test Steps:

```
*1.*   *MAKE (the IUT send a ReadPropertyMultiple-Request for a list of properties L)*
~~1~~*2*.  RECEIVE ReadPropertyMultiple-Request,
           *-- the following is repeated for each object O referenced in L*
           'Object Identifier' =          (*O*~~object indentifier of the specified object~~),
           'List of Property References' = (one or more properties of *O as specified in L* ~~the specified object~~)
~~2~~*3*.  TRANSMIT BACnet-Reject-PDU,
           'Reject Reason' =              UNRECOGNIZED_SERVICE
~~3~~*4*.  REPEAT X = (the properties from *L in the order the IUT chooses* ~~Step 1~~) DO {
           RECEIVE ReadProperty-Request,
               'Object Identifier' =      (object identifier referenced by X),
               'Property Identifier' =    (property identifier referenced by X)
           TRANSMIT ReadProperty-Ack,
               'Object Identifier' =      (object identifier referenced by X),
               'Property Identifier' =    (property identifier referenced by X),
               'Property Value' =         (any valid value)
       }
```

BTL-20.0.1 cr1-11: Add Test Directives in Test Plan for Scheduling Tests [BTLWG-923, CR-0337]

Overview:

Improve directives for Scheduling tests per CR-0337 response.

Changes:

Checklist Changes

None

Test Plan Changes

6.4.5 Is Able to Schedule Values of Different Data Types (Not Simultaneously)

The IUT supports Schedule objects that are capable of being configured to schedule values of different datatypes. An example would be a Schedule object that was originally configured to write values of datatype REAL to an Analog Value object's Present_Value property but is later reconfigured to write Enumerated values to a Binary Value object's Present_Value.

Such a reconfiguration involves changing the List_Of_Object_Property_References property to refer to properties of a specific datatype, and changing the 'value' members of the BACnetTimeValue pairs in the Weekly_Schedule and Exception_Schedule properties to have values in the same datatype as the properties referenced by the List_Of_Object_Property_References property

BTL - 7.3.2.23.11.1 - Internally Written Datatypes Test, non-NULL values		
	Test Conditionality	Must be executed
	Test Directives	Select a single Schedule object that supports multiple datatypes, and run the test multiple times on that one Schedule object. Change the datatype by vendor described means, before running the test, and repeat for at least two different datatypes.
	Testing Hints	
135.1-2019 – 7.3.2.23.11.2 – Internally Written Datatypes Test, NULL values and Priority Arrays		
	Test Conditionality	Must be executed
	Test Directives	*Select a single Schedule object that supports multiple datatypes, and run the test multiple times on that one Schedule object. Change the datatype by vendor described means, before running the test, and repeat for at least two different datatypes.*
	Testing Hints	

[Modified Clause 6.5.10 in BTL Test Plan]

6.5.10 Is Able to Schedule Values of Different Data Types (Not Simultaneously)

The IUT supports Schedule objects that are capable of being configured to schedule values of different datatypes. An example would be a Schedule object that was originally configured to write values of datatype REAL to an Analog Value object's Present_Value property but is later reconfigured to write Enumerated values to a Binary Value object's Present_Value.

Such a reconfiguration involves changing the List_Of_Object_Property_References property to refer to properties of a specific datatype, and changing the 'value' members of the BACnetTimeValue pairs in the Weekly_Schedule and Exception_Schedule properties to have values in the same datatype as the properties referenced by the List_Of_Object_Property_References property

135.1-2019 - 7.3.2.23.11.3 - Externally Written Datatypes Test, non-NULL values		
	Test Conditionality	Must be executed
	Test Directives	*Select a single Schedule object that supports multiple datatypes, and run the test multiple times on that one Schedule object. Change the datatype by vendor described means, before running the test, and repeat for at least two different datatypes.*

	Testing Hints	
135.1-2019 - 7.3.2.23.11.4 - Externally Written Datatypes Test, NULL values and Priority Arrays		
	Test Conditionality	Must be executed
	Test Directives	*Select a single Schedule object that supports multiple datatypes, and run the test multiple times on that one Schedule object. Change the datatype by vendor described means, before running the test, and repeat for at least two different datatypes.*
	Testing Hints	

Test Changes

None

BTL-20.0.1 cr1-12: Improve Trend Test 9.21.1.3 [BTLWG-962, CR-0302]

Overview:

The test is changing to ensure that the result does not always start from the first record, and that if multiple records are in the result then they are in the required proper order.

Changes:

Checklist Changes

None

Test Plan Changes

None
[The changes per the work item are already in the BTL Test Plan 20.0.1]

Test Changes

9.21.1.3 Reading Items by Position with Negative Count

Reason for Change: Make the test applicable to object types other than trends. Changed test to ensure that the result does not always start from the first record.

Purpose: To verify that the IUT correctly responds to a ReadRange service request to return items specified by indicating a position and the number of items before that position to return.

Test Concept: A ReadRange request is transmitted by the TD requesting a range of items known to be in *the list property P* ~~the Log_Buffer~~. This range is specified using the 'By Position' option and a negative value for 'Count'. The 'Reference Index' and 'Count' are selected so that the results can be conveyed in a single acknowledgement.

Configuration Requirements: A list property, P, is configured with N items and N > 1

Test Steps:

1. TRANSMIT ReadRange-Request,
    'Object Identifier' = (the ~~log~~ object configured for this test),
    'Property Identifier' = ~~Log_Buffer~~ *P*,
    'Reference Index' = (any value x: 1 ≤ x ≤ *N* ~~Record_Count~~),
    'Count' = (any value y: y < 0 AND |y| ≤ x)
2. RECEIVE ReadRange-ACK,
    'Object Identifier' = (the ~~log~~ object configured for this test),
    'Property Identifier' = ~~Log_Buffer~~ *P*,
    'Result Flags' = {?, ?, FALSE},
    'Item Count' = |y|,
    'Item Data' = (all of the *items* specified ~~trend records~~ in order of increasing position. The items specified include the item at the index specified by x, plus |y|-1 items preceding.)

--read the first Data-Item

3. TRANSMIT ReadRange-Request,
    'Object Identifier' = (the object configured for this test),
    'Property Identifier' = P,
    'Reference Index' = 2
    'Count' = -2

4. RECEIVE ReadRange-ACK,
    'Object Identifier' = (the object configured for this test),

'Property Identifier' = P,
'Result Flags' = {TRUE, ?, FALSE},
'Item Count' = 2,
'Item Data' = (The items specified include the item at the index 1 and 2 in order of increasing position)

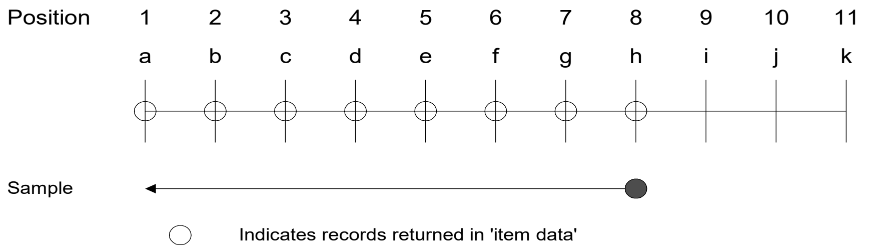
--read the last Data-Item

5. TRANSMIT ReadRange-Request,
'Object Identifier' = (the object configured for this test),
'Property Identifier' = P,
'Reference Index' = N,
'Count' = (any value y: y < 0 AND |y| ≤ N)

6. RECEIVE ReadRange-ACK,
'Object Identifier' = (the log object configured for this test),
'Property Identifier' = P,
'Result Flags' = {?, TRUE, FALSE},
'Item Count' = |y|,
'Item Data' = (all of the items specified in order of increasing position. The items specified include the item at the index specified by N, plus |y|-1 items preceding.)

Test Example (using the sample buffer at beginning of section):

1. TRANSMIT ReadRange-Request,
'Object Identifier' = (Trend Log, Instance 1),
'Property Identifier' = Log_Buffer,
'Reference Index' = 8,
'Count' = -8
2. RECEIVE ReadRange-ACK,
'Object Identifier' = (Trend Log, Instance 1),
'Property Identifier' = Log_Buffer,
'Result Flags' = {TRUE, FALSE, FALSE},
'Item Count' = 8
'Item Data' = Records < a, b, c, d, e, f, g, h > in that order.



BTL-20.0.1 cr1-13: Fix Test 9.20.1.X2 [BTLWG-1019, CR-0479]

Overview:

As pointed out in CR-0479 the test does not allow for the invalid-array-index error to be returned in a RPM-Ack.

Changes:

Checklist Changes

None

Test Plan Changes

None

Test Changes

[Note: This is a BTL Specified Test Only and therefore no Italics are used. The strike-out is here for clarity and will be removed when applied]

[In BTL Specified Tests, update 9.20.1.X2 based on CR-0479]

9.20.1.X2 ReadPropertyMultiple Array Properties
Reason for Change: No test exists for this functionality. This test is not in any SSPC proposal.

Purpose: To verify that the IUT can execute ReadPropertyMultiple service requests when the requested property is an array.

Test Concept: The TD reads the size of the array property, and then reads the first and last entries in the array. Finally, the TD reads past the end of the array and ensures that the IUT returns the correct error.~~, when its size as well as when a single element of the array is requested. Another request is made to read an element of an array where the array index is out of range.~~

Configuration Requirement: O1 is any object in the IUT database having array property P1~~ having size X~~.

Test Steps:

```
~~1.  VERIFY P1 = X, ARRAY INDEX = 0~~
1.      TRANSMIT ReadPropertyMultiple-Request,
            'Object Identifier' = O1,
            'Property Identifier' = P1,
            'Property Array Index' = 0
2.      RECEIVE ReadPropertyMultiple-ACK,
            'Object Identifier' = O1,
            'Property Identifier' = P1,
            'Property Array Index' = 0,
            'Property Value' = (X: the size of the array)

3.  IF (X>0) THEN
        TRANSMIT ReadPropertyMultiple-Request,
            'Object Identifier' = O1,
            'Property Identifier' = P1,
            'Property Array Index' = 1
        RECEIVE ReadPropertyMultiple-ACK,
            'Object Identifier' = O1,
            'Property Identifier' = P1,
            'Property Array Index' = 1,
            'Property Value' = (V, any valid value of the correct data type for property P1)
        TRANSMIT ReadPropertyMultiple-Request,
            'Object Identifier' = O1,
            'Property Identifier' = P1,
            'Property Array Index' = X,
        RECEIVE ReadPropertyMultiple-ACK,
            'Object Identifier' = O1,
            'Property Identifier' = P1,
            'Property Array Index' = X,
            'Property Value' = (V, any valid value of the correct data type for property P1)
~~7.      CHECK (V is any valid value of the correct data type for property P1)~~
4.  TRANSMIT ReadPropertyMultiple-Request,
        'Object Identifier' = O1,
        'Property Identifier' = P1,
        'Property Array Index' = (X+1)
5.  RECEIVE ReadPropertyMultiple-Error,
        'Error Class' = PROPERTY,
        'Error Code' = INVALID_ARRAY_INDEX
    | ReadPropertyMultiple-ACK,
            'Object Identifier' = O1,
            'Property Identifier' = P1,
            'Property Array Index' = X+1,
            'Property Access Error' = (
                'Error Class' = PROPERTY,
                'Error Code' = INVALID_ARRAY_INDEX
            )
```

BTL-20.0.1 cr1-14: Change Test Conditionality for Network Port Objects and Routers [BTLWG-1029, CR-0485]

Overview:

CR-0485 - clarified that changes needed to be made to either the test or the Test Conditionality. This work item changes the Test Conditionality.

Changes:

Checklist Changes

None

Test Plan Changes

[Note: Addenda bl changed the 'writable' to 'configurable' in the section title and changed the description for the section. These changes will be preserved. The only thing changing here per this clarification is the Test Conditionality statement.]

3.56.2 Supports Writable Network_Number Property

The Network_Number property in Network Port objects contained in the IUT is writable.

BTL - 7.3.2.X62.2 - Network-Number-Is Updates Network_Number_Quality Test		
	Test Conditionality	~~Must be executed.~~ *For routers which do not accept a value of zero in their Network_Number properties, this test shall be skipped.*
	Test Directives	
	Testing Hints	

Test Changes

None