Clarification Request

References: Test plans 14 and 15, BTL Specified Tests in 7.3.1.X9.1 7.3.1.X9.2

Date of BTL-WG Response: 16-Nov-2017

☒All Actions Necessitated have been Completed

Background:
The current test plan shows an inconsistent approach regarding the requirement for manipulation of the Event_Detection_Enable property.

Problem:

For AE-N-I-B testing (Base Requirements), 7.3.1.X9.1 (shown below) and 7.3.1.X9.2 ; there are no provisions in the test plan or the specified tests that allow these tests to be skipped if Event_Detection_Enable can only be TRUE.

7.3.1.X9.1 Event_Detection_Enable Inhibits Event Generation
Reason for Change: New functionality added with Addendum 135-2010af. This test does not exist in 135.1-2013.

Purpose: To verify that Event_Detection_Enable enables and disables event detection in objects which are configured for event reporting.

Test Concept: Select an event generating object, O1 that is configured for event reporting. Make the object generate an event, to an offnormal state if possible, so that if the object can have a non-normal state, it enters that state early in the test. This will help detect incorrect implementations that initiate a TO_NORMAL transition when the algorithm is disabled. Set the Event_Detection_Enable property to FALSE. Verify the Event_State is NORMAL and the Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts are equal to their respective initial conditions, as mandated in the standard. Repeat the process that made the object generate an event and observe that no notification messages are transmitted.

Configuration Requirements: O1 is configured to detect and report unconfirmed events and requires acknowledgments for all transitions. Event_Detection_Enable is equal to TRUE. DELAY shall represent the time delay appropriate to the transition being tested (i.e. Time_Delay for TO_OFFNORMAL, 0 for TO_FAULT and FAULT to NORMAL transitions, and either Time_Delay or Time_Delay_Normal for TO_NORMAL). For this test, NO_TS equals a BACnetDateTime with all unspecified values, a BACnet Time with all unspecified values, or a sequence number of 0.

Test Steps:
1. VERIFY Event_Detection_Enable = TRUE
2. MAKE (a condition exist which will cause O1 to transition, to an offnormal state if possible)
3. WAIT DELAY
4. 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' = (any valid event type),
'Notify Type' = (value from the Notify_Type property configured for O1),
'Message Text' = (any valid message text),
'AckRequired' = TRUE,
'From State' = (any valid event state),
'To State' = (any event state appropriate to the event type),
'Event Values' = (any values appropriate to the event type)

5. IF Event_Detection_Enable is writable THEN
   WRITE Event_Detection_Enable = FALSE
   ELSE
   MAKE (Event_Detection_Enable to FALSE. This property is expected to be set during system configuration and is not expected to change dynamically.)
6. WAIT DELAY + **Notification Fail Time + Internal Processing Fail Time**
7. CHECK (that the IUT did not send any further event notifications for O1)
8. VERIFY Event_State = NORMAL
9. VERIFY Acked_Transitions = (T,T,T)
10. VERIFY Event_Time_Stamps = [NO_TS , NO_TS , NO_TS]
11. IF the Event_Message_Texts property exists THEN
    VERIFY Event_Message_Texts = ['', '', '']
12. MAKE (a condition exist which would cause O1 to transition, if Event_Detection_Enable were TRUE)
13. WAIT DELAY + **Notification Fail Time**
14. CHECK (that the IUT did not send any event notifications for O1)
15. VERIFY Event_State = NORMAL
16. VERIFY Acked_Transitions = (T,T,T)
17. VERIFY Event_Time_Stamps = [NO_TS, NO_TS, NO_TS]
18. IF the Event_Message_Texts property exists THEN
    VERIFY Event_Message_Texts = ['', '', '']

Notes to Tester: This behavior can alternately be tested using the ConfirmedEventNotification service, but it is not necessary to test both.

Question:

1) Is BTL imposing a requirement that Event_Detection_Enable can be made to have a value of TRUE or FALSE?
2) Is BTL imposing a requirement that Event_Detection_Enable can be made to have a value of FALSE?
3) If No to Question #1, can test 7.3.1.X9.1 be skipped if Event_Detection_Enable is non-configurable in all objects?

4) Can test 7.3.1.X9.2 be skipped if there is no object with the intrinsic event properties but which can be configured with Event_Detection_Enable equal to FALSE?

Response:

The BTL-WG reviewed the test and the standard. Based on the standard's position that Event_Detection_Enable is not expected to change during normal operations, the approach to testing Event_Detection_Enable by this test needs to be reconsidered.

1) **No. This is an error in the test package.**
2) **No. This is an error in the test package.**
3) **Yes. The test shall be skipped by all devices until it can be corrected.**
4) **Yes. In this case, the test shall be skipped. In addition, this test shall be skipped if no object in the IUT, which can be made to have Event_Detection_Enable be false, is able to detect faults while event detection is disabled.**