

Clarification Request

References: 135.1-2013 7.3.1.1 7.3.1.1 Out_Of_Service, Status_Flags, and Reliability Tests

Date of BTL-WG Response: 09-Sept-2019

All Actions Necessitated have been Completed

Background:

IC135-2010-14 clarifies Out_Of_Service for Value objects. Test 7.3.1.1 appears to not properly test Value objects.

Interpretation #2 states:

Interpretation No.2: BACnet objects within the local device are considered to be local programming and as such cannot write / command the Present_Value of an object in which Out_Of_Service is TRUE.

Question No.2: Is this interpretation correct?

Answer No.2: Yes.

7.3.1.1 Out_Of_Service, Status_Flags, and Reliability Tests

Dependencies: ReadProperty Service Execution Tests, 9.18; WriteProperty Service Execution Tests, 9.22.

BACnet Reference Clauses: 12.1.7, 12.1.9, 12.1.10, 12.2.7, 12.2.9, 12.2.10, 12.3.7, 12.3.9, 12.3.10, 12.4.6, 12.4.8, 12.4.9, -12.6.7, 12.6.9, 12.6.10, 12.7.7, 12.7.9, 12.7.10, 12.8.6, 12.8.8, 12.8.9, 12.15.8, 12.15.10, 12.15.11, 12.16.8, 12.16.10, 12.16.11, 12.17.6, 12.17.8, 12.17.9, 12.18.7, 12.18.9, 12.18.10, 12.19.7, 12.19.9, 12.19.10, 12.20.6, 12.20.8, 12.20.9, 12.23.7, 12.23.9, and 12.23.10.

Purpose: This test case verifies that Present_Value is writable when Out_Of_Service is TRUE. It also verifies the interrelationship between the Out_Of_Service, Status_Flags, and Reliability properties. If the PICS indicates that the Out_Of_Service property of the object under test is not writable, and if the value of the property cannot be changed by other means, then this test shall be omitted. This test applies to Accumulator, Analog Input, Analog Output, Analog Value, Binary Input, Binary Output, Binary Value, Life Safety Point, Life Safety Zone, Multi-state Input, Multi-state Output, Multi-state Value, Loop and Pulse Converter objects.

Test Concept: The IUT will select one instance of each appropriate object type and test it as described. If the Reliability property is not supported then step 4 shall be omitted.

Test Steps:

1. IF (Out_Of_Service is writable) THEN

```

    WRITE Out_Of_Service = TRUE
  ELSE
    MAKE (Out_Of_Service TRUE)
2. VERIFY Out_Of_Service = TRUE
3. VERIFY Status_Flags = (?, FALSE, ?, TRUE)
4. REPEAT X = (all values meeting the functional range requirements of 7.2.1) DO {
    WRITE Present_Value = X
    VERIFY Present_Value = X
  }
5. IF (Reliability is present and writable) THEN
    REPEAT X = (all values of the Reliability enumeration appropriate to the object
type except
        NO_FAULT_DETECTED) DO {
    WRITE Reliability = X
    VERIFY Reliability = X
    VERIFY Status_Flags = (?, TRUE, ?, TRUE)
    WRITE Reliability = NO_FAULT_DETECTED
    VERIFY Reliability = NO_FAULT_DETECTED
    VERIFY Status_Flags = (?, FALSE, ?, TRUE)
  }
6. IF (Out_Of_Service is writable) THEN
    WRITE Out_Of_Service = FALSE
  ELSE
    MAKE (Out_Of_Service FALSE)
7. VERIFY Out_Of_Service = FALSE
8. VERIFY Status_Flags = (?, ?, ?, FALSE)

```

Notes to Tester: If the object being tested is commandable and there is an internal process writing to the Present_Value property, then each WriteProperty request shall contain a priority sufficient to override the internal process. After step 4 the priority array slot shall be relinquished.

Problem:

With respect to value objects, the Notes to Tester in 7.3.1.1 allows internal operations to continue to write to the Present_Value/Priority_Array when Out_Of_Service is TRUE.

Question:

Should the test be modified to ensure Out_Of_Service is tested correctly based on the type of object?

Response:

Yes. For Value Objects, the highlighted portion of Notes to Tester should be ignored.