Method of Test for Conformance to BACnet *Errata*
ANSI/ASHRAE STANDARD 135.1-2007

February 10, 2010

This document lists all known *errata* to ANSI/ASHRAE 135.1-2007 as of the above date. Each entry is cited first by clause, then page number. The outside back cover marking identifying the first printing of Standard 135.1-2007 is "86446 9/07" and "Product Code: 86446 2/09" for the second printing. Items 2) through 7) have been added since the previously published errata sheet dated June 27, 2008 was distributed. Items 8) through 14) have been added since the previously published errata sheet dated January 24, 2009 was distributed.

1) Unpublished material was inadvertently added to the standard while in draft. Revise as follows (strikeout indicates text to be removed, italic text to be re-inserted):

[Clause **4.5.4**, p. 9, remove new object references]

…
The standard objects may be any of:

~~Access Door~~	Command	Multi-state Output
~~Accumulator~~	Device	Multi-state Value
Analog Input	Event Enrollment	Notification Class
Analog Output	File	Program
Analog Value	Group	~~Pulse Converter~~
Averaging	Life Safety Point	Schedule
Binary Input	Life Safety Zone	~~Structured View~~
Binary Output	~~Load Control~~	Trend Log
Binary Value	Loop	
Calendar	Multi-state Input	

[Clause **4.5.10**, p. 11-12, remove added material]

…
Properties in the test database that are writable shall have a "W" following the property value, as shown in the example below:

```
{
object-identifier: (analog-value, 6)
object-name: "❑"
object-type: analog-value
present-value: 23.4 W
other properties...
}
```

~~Properties in the test database that are conditionally writable shall have a "C" following the property value, as shown in the example below. It is recommended that the governing mechanism be identified in a comment:~~

~~{~~
~~object-identifier: (analog-input, 6)~~
~~object-name: "❑"~~
~~object-type: analog-input~~
~~present-value: 12.3 C -- Writable when Out_Of_Service is TRUE~~
~~other properties...~~
~~}~~

The following sections show templates for each of the standard object types. To improve readability the carriage return/linefeed pairs are not explicitly shown in the examples.

[Remove the following clauses and renumber as follows:]

[Remove **4.5.10.1 Accumulator**, pp. 12-13]

[Renumber **4.5.10.2 Analog Input** through **4.5.10.22 Program** to **4.5.10.1** through **4.5.10.21**]

[Remove **4.5.10.23 Pulse Converter**, p. 22-23]

[Renumber **4.5.10.24 Schedule** through **4.5.10.25 Trend Log** to **4.5.10.22** through **4.5.10.23**]

[Remove **4.5.10.26 Access Door**, p. 24]

[Remove **4.5.10.27 Load Control**, p. 25]

[Remove **4.5.10.28 Structured View**, p. 25]

[Clause **7.3.2.4.2**, p. 56, Test Concept paragraph, remove final sentence]

Test Concept: An Averaging object is configured to monitor a property that can be controlled manually by the testing agent or by the TD. The TD initializes the sample and then monitors the Minimum_Value, Average_Value, Maximum_Value, Attempted_Samples, and Valid_Samples properties after each sampling interval to verify that their values are properly tracking the monitored value. This requires the ability to manipulate the values of the monitored property value and a slow enough sampling interval to permit the analysis. This continues until after the sample window is full. ~~If the IUT does not support Averaging object configuration, then this test shall be omitted.~~

[Clause **7.3.2.9.1**, p. 61, Test Concept paragraph, remove final sentence]

Test Concept: The IUT is configured with an action list that includes manipulating a sequence of externally visible outputs with a time delay between each output. The TD triggers this action list and the tester observes the external changes. ~~If the IUT does not support Post Delay, then this test shall be omitted. If the IUT does not support action list configuration, then this test shall be omitted.~~

[Clause **7.3.2.9.2**, p. 62, Test Concept paragraph, remove final sentence]

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, then this test shall be omitted.~~

[Clause **7.3.2.9.4**, p. 63, remove Test Concept paragraph]

~~Test Concept: The IUT is configured with at least one empty action list. The TD triggers the action list. The external outputs are observed to verify that no changes occurred. If the IUT does not support action list configuration, then this test shall be omitted.~~

[Clause **7.3.2.9.7**, pp. 64-65, remove Test Concept paragraph]

~~Test Concept: The IUT is configured with two action lists that include a sequence of externally visible outputs with post delays for each action. The TD triggers the first action list. The external outputs are observed in order to trigger the second action list during the post delay of the first list. The TD triggers the second action list. The external outputs are observed to verify that the second action list is not executed. If the IUT does not support Post Delay, then this test shall be omitted. If the IUT does not support action list configuration, then this test shall be omitted.~~

[Clause **7.3.2.14**, p.70, remove additions to Test Step 6]

6. ~~IF (a property value of a group member is changeable) THEN~~

```
    ~~IF (the changeable group member property value is writable) THEN~~
.       WRITE (the writable property that is a member of the group) = (a value different from its current value)
    ~~*ELSE*~~
        ~~*MAKE (the changeable group member property value different from its current value)*~~
```

[Remove Clause **9.1.1.7,** pp. 196-197]

[Clause **9.1.2.3**, p.200, undo changes as shown]

9.1.2.3 Unsuccessful Alarm Acknowledgment of Confirmed Event Notifications Because the *'Event Object Identifier' is Invalid* ~~Referenced Object Does Not Exist~~

Purpose: This test case verifies that an alarm remains unacknowledged if the 'Event Object Identifier' represents an object that does not exist *or is not consistent with the other parameters that define the alarm being acknowledged.*

…

Test Steps: The test steps defined in 9.1.2.1 shall be followed except that in the first AcknowledgeAlarm request the 'Time Stamp' shall have the same value as the 'Time Stamp' from the event notification, and the 'Event Object Identifier' shall have a value that is different from the 'Event Object Identifier' in the event notification ~~and for which no object exists in the IUT.~~

Passing Result: A passing result is the same message sequence described in 9.1.2.1 except that the ~~Error Class in step 7 shall be OBJECT and th~~*e* Error Code in step 7 shall be *INCONSISTENT_PARAMETERS.* ~~UNKNOWN_OBJECT. For devices that claim a Protocol_Revision of 5 or prior, an Error Class of SERVICES with an Error Code of INCONSISTENT_PARAMETERS shall also be accepted.~~

[Clause **9.1.2.4**, p.200, undo changes as shown]

Test Steps: The test steps defined in 9.1.2.1 shall be followed except that in the first AcknowledgeAlarm request the 'Time Stamp' shall have the same value as the 'Time Stamp' from the event notification and the 'Event State Acknowledged' shall have a~~n off-normal~~ value ~~other than OFFNORMAL and other than the value of~~ *that is different from* the 'To State' ~~parameter~~ in the event notification.

Passing Result: A passing result is the same message sequence described in 9.1.2.1 except that the Error Code in step 7 shall be *INCONSISTENT_PARAMETERS.* ~~INVALID_EVENT_STATE. For devices that claim a Protocol_Revision of 5 or prior, an Error Code of INCONSISTENT_PARAMETERS shall also be accepted.~~

[Clause **9.1.2.6**, p.203, undo changes as shown]

9.1.2.6 Unsuccessful Alarm Acknowledgment of Unconfirmed Event Notifications Because the *'Event Object Identifier' is Invalid* ~~Referenced Object Does Not Exist~~

Purpose: This test case verifies that an alarm remains unacknowledged if the 'Event Object Identifier' represents an object that does not exist *or is not consistent with the other parameters that define the alarm being acknowledged.*

…

Test Steps: The test steps defined in 9.1.2.5 shall be followed except that in the first AcknowledgeAlarm request the 'Time Stamp' shall have the same value as the 'Time Stamp' from the event notification, and the 'Event Object Identifier' shall have a value that is different from the 'Event Object Identifier' in the event notification ~~and for which no object exists in the IUT.~~

Passing Result: A passing result is the same message sequence described as the passing result in 9.1.2.5 except that the ~~Error Class in step 7 shall be OBJECT and th~~*e* Error Code in step 7 shall be *INCONSISTENT_PARAMETERS.* ~~UNKNOWN_OBJECT. For devices that claim a Protocol_Revision of 5 or prior, an Error Class of SERVICES with an Error Code of INCONSISTENT_PARAMETERS shall also be accepted.~~

[Clause **9.1.2.7**, p.203, undo changes as shown]

Test Steps: The test steps defined in 9.1.2.1 shall be followed except that in the first AcknowledgeAlarm request the 'Time Stamp' shall have the same value as the 'Time Stamp' from the event notification and the 'Event State Acknowledged' shall have a~~n off-normal~~ value ~~other than OFFNORMAL and other than the value of~~ *that is different from* the 'To State' ~~parameter~~ in the event notification.

Passing Result: A passing result is the same message sequence described as the passing result in 9.1.2.5 except that the Error Code in step 7 shall be *INCONSISTENT_PARAMETERS.* ~~INVALID_EVENT_STATE. For devices that claim a Protocol_Revision of 5 or prior, an Error Code of INCONSISTENT_PARAMETERS shall also be accepted.~~

2) [This erratum removed – redundantly listed]

3) [This erratum removed – referred to wrong standard]

4) Clause 7.3.2.19.6, p. 73. The reference to Clause 7.3.2.15.5 is incorrect. It should be to Clause 7.3.2.18.5.

Tests to verify the Number_Of_States value and State_Text array size of Multi-state Output objects are defined in ~~7.3.2.15.5~~ *7.3.2.18.5*. Run the tests using a Multi-state Output object.

5) Clause 7.3.2.20.5, p. 74. The reference to Clause 7.3.2.15.5 is incorrect. It should be to Clause 7.3.2.18.5.

Tests to verify the Number_Of_States value and State_Text array size of Multi-state Value objects are defined in ~~7.3.2.15.5~~ *7.3.2.18.5*. Run the tests using a Multi-state Value object.

6) Clause 7.3.2.21.3.2, p. 78. Test steps 10 and 11 should be numbered 5 and 6. Correct as shown:

7.3.2.21.3.2 FromTime and ToTime Test

```
…
4.  IF (X is writable) THEN
            WRITE X = (a value that is OFFNORMAL)
    ELSE
            MAKE (X have a value that is OFFNORMAL)
~~10.~~ *5.*  WAIT (Time_Delay + Notification Fail Time)
~~11.~~ *6.*  CHECK (verify that no notification message was transmitted)
```

7) Clause 8.2.8 p. 116. Step 6 incorrectly refers to Step 4. The correct reference is to Step 5. Correct as shown:

8.2.8 Change of Value Notification from a Loop Object Status_Flags Property

```
…
6. IF (WriteProperty is used in ~~step 4~~ *step 5*) THEN
        RECEIVE BACnet-SimpleACK-PDU
```

8) Clause 8.29.3 p. 175, and Clause 8.29.4, p. 176. Step 1 incorrectly calls for the use of a ConfirmedTextMessage in a group of tests of UnconfirmedTextMessage services. Correct as shown:

8.29.3 Text Message with a CharacterString Message Class

…

1. RECEIVE ~~Confirmed~~*Unconfirmed*TextMessage-Request,

…

8.29.4 Text Message with an Urgent Priority
…
1. RECEIVE ~~Confirmed~~*Unconfirmed*TextMessage-Request,
…

9) Clause 7.3.2.24.9. p. 101, in the "Purpose" paragraph, a clause reference is incorrect. Correct as follows:

"Note: it is not reasonable to test for the requirement of BACnet Clause ~~12.23.16~~ *12.25.16* that…"

10) Clause 8.3.9, p. 118 is testing for initiations of unsubscribed UnconfirmedCOVNotification, but the test step incorrectly calls out for ConfirmedCOVNotification and does not specify the required zero value for Subscriber Process Identifier. Correct step 2 as follows:

2. RECEIVE ~~Confirmed~~*Unconfirmed*COVNotification-Request,
'Subscriber Process Identifier' = ~~(any valid process ID)~~ *0*,

11) Clause 9.16, p. 248. All the subclauses of 9.16 incorrectly use either 'Object Type' or 'Object Identifier' in place of 'Object Specifier' as the first parameter of a CreateObject-Request message. Correct as follows:

n. TRANSMIT CreateObject-Request,
~~'Object Type'~~ *'Object Specifier'* = …

or

n. TRANSMIT CreateObject-Request,
~~'Object Identifier'~~ *'Object Specifier'* = …

12) Clause 9.24.1.2, p. 281. There is a missing single quote character before 'Reinitialized State of Device'.

9.24.1.2 Indefinite Time Duration Restored by ReinitializeDevice

...
7. TRANSMIT ReinitializeDevice-Request,
~~Reinitialized State of Device' = WARMSTART,~~
'Reinitialized State of Device' = WARMSTART,

13) Clause 9.24.1.7, p. 283. There is a missing single quote character before 'Reinitialized State of Device'.

9.24.1.7 Indefinite Time Duration Restored by ReinitializeDevice

...
9. TRANSMIT ReinitializeDevice-Request,
~~Reinitialized State of Device' =WARMSTART,~~
'Reinitialized State of Device' = WARMSTART,

14) Clause 9.24.1.2, p. 281. Step 9 calls for a check of a COLDSTART reboot, but a WARMSTART was sent. Correct to match line 7 (see 9.24.1.5 and 9.24.1.7).

9.24.1.2 Indefinite Time Duration Restored by ReinitializeDevice

...

7. TRANSMIT ReinitializeDevice-Request,
   Reinitialized State of Device' =WARMSTART,
   'Password' = (any appropriate password as described in the Test Concept)
8. RECEIVE BACnet-Simple-ACK-PDU
9. CHECK (Did the IUT perform a ~~COLDSTART~~ *WARMSTART* reboot?)

...