



# **ASHRAE ADDENDA**

## **Method of Test for Conformance to BACnet<sup>®</sup>**

Approved by the ASHRAE Standards Committee on January 29, 2011; by the ASHRAE Board of Directors on February 2, 2011; and by the American National Standards Institute on February 3, 2011.

This addendum was approved by a Standing Standard Project Committee (SSPC) for which the Standards Committee has established a documented program for regular publication of addenda or revisions, including procedures for timely, documented, consensus action on requests for change to any part of the standard. The change submittal form, instructions, and deadlines may be obtained in electronic form from the ASHRAE Web site ([www.ashrae.org](http://www.ashrae.org)) or in paper form from the Manager of Standards.

The latest edition of an ASHRAE Standard may be purchased on the ASHRAE Web site ([www.ashrae.org](http://www.ashrae.org)) or from ASHRAE Customer Service, 1791 Tullie Circle, NE, Atlanta, GA 30329-2305. E-mail: [orders@ashrae.org](mailto:orders@ashrae.org). Fax: 404-321-5478. Telephone: 404-636-8400 (worldwide), or toll free 1-800-527-4723 (for orders in US and Canada). For reprint permission, go to [www.ashrae.org/permissions](http://www.ashrae.org/permissions).

© 2011 American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc.



ISSN 1041-2336

**American Society of Heating, Refrigerating  
and Air-Conditioning Engineers, Inc.**  
1791 Tullie Circle NE, Atlanta, GA 30329  
[www.ashrae.org](http://www.ashrae.org)

**ASHRAE Standing Standard Project Committee 135**  
**Cognizant TC: TC 1.4, Control Theory and Application**  
**SPLS Liaison: Richard L. Hall**

|                                   |                       |                      |
|-----------------------------------|-----------------------|----------------------|
| David Robin, <i>Chair*</i>        | Daniel P. Giorgis     | David G. Shike       |
| Carl Neilson, <i>Vice-Chair</i>   | David G. Holmberg     | Ted Sunderland       |
| Bernhard Isler, <i>Secretary*</i> | Robert L. Johnson     | William O. Swan, III |
| Donald P. Alexander*              | Stephen Karg*         | David B. Thompson*   |
| Barry B. Bridges*                 | Simon Lemaire         | Daniel A. Traill     |
| Coleman L. Brumley, Jr.           | J. Damian Ljungquist* | Stephen J. Treado*   |
| Ernest C. Bryant                  | James G. Luth         | Klaus Wagner         |
| James F. Butler                   | John J. Lynch         | J. Michael Whitcomb* |
| A. J. Capowski                    | Brian Meyers          | David F. White       |
| Clifford H. Copass                | Dana Petersen         | Grant N. Wichenko*   |
| Sharon E. Dinges*                 | Carl J. Ruther        | Christoph Zeller     |
| Craig P. Gemmill                  | Frank Schubert        | Scott Ziegenfus      |

*\*Denotes members of voting status when the document was approved for publication.*

---

**ASHRAE STANDARDS COMMITTEE 2010–2011**

|                                      |                    |                                  |
|--------------------------------------|--------------------|----------------------------------|
| H. Michael Newman, <i>Chair</i>      | Allan B. Fraser    | Janice C. Peterson               |
| Carol E. Marriott, <i>Vice-Chair</i> | Krishnan Gowri     | Douglas T. Reindl                |
| Douglass S. Abramson                 | Maureen Grasso     | Boggarm S. Setty                 |
| Karim Amrane                         | Cecily M. Grzywacz | James R. Tauby                   |
| Robert G. Baker                      | Richard L. Hall    | James K. Vallort                 |
| Hoy R. Bohanon, Jr.                  | Nadar R. Jayaraman | William F. Walter                |
| Steven F. Bruning                    | Byron W. Jones     | Michael W. Woodford              |
| Kenneth W. Cooper                    | Jay A. Kohler      | Craig P. Wray                    |
| Martin Dieryckx                      | Frank Myers        | Hugh F. Crowther, <i>BOD ExO</i> |
|                                      |                    | William P. Bahnfleth, <i>CO</i>  |

Stephanie Reiniche, *Manager of Standards*

---

**SPECIAL NOTE**

This American National Standard (ANS) is a national voluntary consensus standard developed under the auspices of the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE). *Consensus* is defined by the American National Standards Institute (ANSI), of which ASHRAE is a member and which has approved this standard as an ANS, as “substantial agreement reached by directly and materially affected interest categories. This signifies the concurrence of more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that an effort be made toward their resolution.” Compliance with this standard is voluntary until and unless a legal jurisdiction makes compliance mandatory through legislation.

ASHRAE obtains consensus through participation of its national and international members, associated societies, and public review.

ASHRAE Standards are prepared by a Project Committee appointed specifically for the purpose of writing the Standard. The Project Committee Chair and Vice-Chair must be members of ASHRAE; while other committee members may or may not be ASHRAE members, all must be technically qualified in the subject area of the Standard. Every effort is made to balance the concerned interests on all Project Committees.

The Manager of Standards of ASHRAE should be contacted for:

- a. interpretation of the contents of this Standard,
- b. participation in the next review of the Standard,
- c. offering constructive criticism for improving the Standard, or
- d. permission to reprint portions of the Standard.

**DISCLAIMER**

ASHRAE uses its best efforts to promulgate Standards and Guidelines for the benefit of the public in light of available information and accepted industry practices. However, ASHRAE does not guarantee, certify, or assure the safety or performance of any products, components, or systems tested, installed, or operated in accordance with ASHRAE's Standards or Guidelines or that any tests conducted under its Standards or Guidelines will be nonhazardous or free from risk.

**ASHRAE INDUSTRIAL ADVERTISING POLICY ON STANDARDS**

ASHRAE Standards and Guidelines are established to assist industry and the public by offering a uniform method of testing for rating purposes, by suggesting safe practices in designing and installing equipment, by providing proper definitions of this equipment, and by providing other information that may serve to guide the industry. The creation of ASHRAE Standards and Guidelines is determined by the need for them, and conformance to them is completely voluntary.

In referring to this Standard or Guideline and in marking of equipment and in advertising, no claim shall be made, either stated or implied, that the product has been approved by ASHRAE.

[This foreword and the “rationales” on the following pages are not part of this standard. They are merely informative and do not contain requirements necessary for conformance to the standard.]

## FOREWORD

Addendum 135.1g to ANSI/ASHRAE Standard 135.1-2009 contains a number of changes to the current standard. These modifications are the result of change proposals made pursuant to the ASHRAE continuous maintenance procedures and of deliberations within Standing Standard Project Committee 135. The changes are summarized below.

- 135.1-2009g-1 Correct Test Step Indention, p. 2.**
- 135.1-2009g-2 Remove Recipient Test, p. 4.**
- 135.1-2009g-3 Correct Errors in Routing Tests, p. 5.**
- 135.1-2009g-4 Change the Ignore Process ID Test, p. 9.**
- 135.1-2009g-5 Add Max Info Frames Check, p. 15.**
- 135.1-2009g-6 Add Test for Device Identifier Recipients, p. 16.**
- 135.1-2009g-7 Add Test for Network Address Recipients, p. 17.**
- 135.1-2009g-8 Add Tests for Disable Initiation, p. 18.**
- 135.1-2009g-9 Change Tests for Out\_Of\_Service, Status\_Flags, and Reliability, p. 20.**
- 135.1-2009g-10 Add Tests for Non-router Network Layer Messages, p. 22.**
- 135.1-2009g-11 Remove Time Delay in TO-FAULT Tests, p. 29.**
- 135.1-2009g-12 Make Additions to the TCSL Language, p. 31.**
- 135.1-2009g-13 Change Acknowledge Alarm Initiation Tests, p. 33.**
- 135.1-2009g-14 Add New Tests for Reading and Presenting Properties, p. 34.**
- 135.1-2009g-15 Add New Event Notification Tests, p. 36.**
- 135.1-2009g-16 Update Trending Tests for Revision 3, p. 38.**
- 135.1-2009g-17 Add New Tests for Revision 4 Schedules, p. 48.**
- 135.1-2009g-18 Add New Test for Event Notification Network Priority, p. 57.**
- 135.1-2009g-19 Add Device and Network Mapping Tests, p. 59.**
- 135.1-2009g-20 Add Device Restart Notification Tests, p. 60.**
- 135.1-2009g-21 Add Schedule Written Datatypes Tests, p. 62.**

In the following document, language added to existing clauses of ANSI/ASHRAE 135.1-2009 and addenda is indicated through the use of *italics*, while deletions are indicated by ~~strike through~~. Where entirely new subclauses are added, plain type is used throughout.

### 135.1-2009g-1 Correct Test Step Indentation.

#### Rationale

The test step 8 is incorrectly numbered and incorrectly indented.

[Change **Clause 8.2.7**, p 113]

#### 8.2.7 Change of Value Notification from Loop Object Present\_Value Property

Purpose: To verify that the IUT can initiate ConfirmedCOVNotification service requests conveying a change of the Present\_Value property of a loop object.

Test Concept: A subscription for COV notifications is established. The Present\_Value of the monitored object is changed by an amount less than the COV increment and it is verified that no COV notification is received. The Present\_Value is then changed by an amount greater than the COV increment and a notification shall be received.

The Present\_Value may be changed by placing the Loop Out\_Of\_Service and writing directly to the Present\_Value. For implementations where this option is not ~~possible~~ *an alternative*, an alternative trigger mechanism shall be provided to accomplish this task, such as changing the Setpoint or the Setpoint\_Reference. All of these methods are equally acceptable.

The object identifier of the Loop object being tested is designated as L in the test steps below.

Test Steps:

1. TRANSMIT SubscribeCOV-Request,  
     'Subscriber Process Identifier' = (any value > 0 chosen by the TD),  
     'Monitored Object Identifier' = L,  
     'Issue Confirmed Notifications' = TRUE,  
     'Lifetime' = 0
2. RECEIVE BACnet-SimpleACK-PDU
3. RECEIVE ConfirmedCOVNotification-Request,  
     'Subscriber Process Identifier' = (the same value used in step 1),  
     'Initiating Device Identifier' = IUT,  
     'Monitored Object Identifier' = L,  
     'Time Remaining' = 0,  
     'List of Values' = (the initial Present\_Value, initial Status\_Flags, initial Setpoint,  
                             and initial Controlled\_Variable\_Value)
4. TRANSMIT BACnet-SimpleACK-PDU
5. TRANSMIT ReadProperty-Request,  
     'Object Identifier' = L,  
     'Property Identifier' = COV\_Increment
6. RECEIVE BACnet-ComplexACK-PDU,  
     'Object Identifier' = L,  
     'Property Identifier' = COV\_Increment,  
     'Property Value' = (a value "increment" that will be used below)
7. IF (Out\_Of\_Service is writable) THEN  
     WRITE X, Out\_Of\_Service = TRUE  
     ~~RECEIVE BACnet-SimpleACK-PDU~~  
     BEFORE **Notification Fail Time**  
     RECEIVE ConfirmedCOVNotification-Request,  
         'Subscriber Process Identifier' = (the same value used in step 1),  
         'Initiating Device Identifier' = IUT,  
         'Monitored Object Identifier' = L,  
         'Time Remaining' = 0,  
         'List of Values' = (the initial Present\_Value, new Status\_Flags, initial Setpoint,

and initial Controlled\_Variable\_Value)

8. TRANSMIT BACnet-SimpleACK-PDU  
[ the following steps have been renumbered ]

8. IF (Present\_Value is now writable) THEN  
    WRITE X, Present\_Value = (any value that differs from "initial Present\_Value" by less than "increment")  
    ~~RECEIVE BACnet-SimpleACK-PDU~~  
    ELSE  
        MAKE (Present\_Value = any value that differs from "initial Present\_Value" by less than "increment")

9. WAIT **NotificationFailTime**

10. CHECK (verify that no COV notification was transmitted)

11. IF (Present\_Value is now writable) THEN  
    WRITE X, Present\_Value = (any value that differs from "initial Present\_Value" by an amount greater than "increment")  
    ~~RECEIVE BACnet-SimpleACK-PDU~~  
    ELSE  
        MAKE (Present\_Value = any value that differs from "initial Present\_Value" by an amount greater than "increment")

12. BEFORE **NotificationFailTime**  
    RECEIVE ConfirmedCOVNotification-Request,  
        'Subscriber Process Identifier' = (the same value used in step 1),  
        'Initiating Device Identifier' = IUT,  
        'Monitored Object Identifier' = L,  
        'Time Remaining' = 0,  
        'List of Values' = (the new Present\_Value, new Status\_Flags, initial Setpoint, and initial Controlled\_Variable\_Value)

13. TRANSMIT BACnet-SimpleACK-PDU

14. TRANSMIT SubscribeCOV-Request,  
    'Subscriber Process Identifier' = (the same value used in step 1),  
    'Monitored Object Identifier' = L

15. RECEIVE BACnet-SimpleACK-PDU

16. IF (Out\_Of\_Service is writable) THEN  
    WRITE L, Out\_Of\_Service = FALSE  
    ~~RECEIVE BACnet-SimpleACK-PDU~~

### 135.1-2009g-2 Remove Recipient Test.

#### Rationale

The Recipient property was previously removed from the Event Enrollment object. The corresponding test should be removed from 135.1.

[Change **Clause 9.4.4**, p. 206]

#### 9.4.4 ConfirmedEventNotification Without a Notification Class Parameter

*This test has been removed.*

##### Test Steps:

1. ~~TRANSMIT ConfirmedEventNotification Request,~~
  - ~~'Process Identifier' = (any valid process identifier),~~
  - ~~'Initiating Device Identifier' = TD,~~
  - ~~'Event Object Identifier' = (any Event Enrollment object),~~
  - ~~'Time Stamp' = (current time using the DateTime format),~~
  - ~~'Priority' = (any valid priority),~~
  - ~~'Event Type' = (any standard event type),~~
  - ~~'Notify Type' = ALARM | EVENT,~~
  - ~~'AckRequired' = FALSE,~~
  - ~~'From State' = NORMAL,~~
  - ~~'To State' = (any non-normal state appropriate to the event type),~~
  - ~~'Event Values' = (any values appropriate to the event type)~~
2. ~~RECEIVE BACnet SimpleACK PDU~~
3. ~~CHECK (for any vendor defined observable actions)~~

### 135.1-2009g-3 Correcting Errors in Routing Tests.

#### Rationale

Errors have identified in a number of routing tests in ANSI/ASHRAE Standard 135.1-2009. The most common types of errors are the following:

- Hop Count field present in a message when it should not be present
- Use of the DESTINATION keyword instead of DA
- Use of the SOURCE keyword instead of SA

Note that DA and SA specify the MAC addresses that should be present in a packet. It is often important to specify the MAC addresses in routing tests.

[Change **Clause 10.2.2.3**, p. 316]

#### 10.2.2.3 Forward I-Could-Be-Router-To-Network

BACnet Reference Clause: 6.6.3.4.

Purpose: To verify that the IUT will forward a received I-Could-Be-Router-To-Network message to the intended recipient.

Test Steps:

1. TRANSMIT PORT B,  
~~DESTINATION~~ DA = IUT,  
SOURCE SA = HR2-4,  
DNET = 1,  
DADR = D1A,  
Hop Count = 255,  
I-Could-Be-Router-To-Network,  
Network Number = 4,  
Performance Index = 6
2. RECEIVE PORT A,  
~~DESTINATION~~ DA = D1A,  
~~SOURCE~~ SA = IUT,  
SNET = 2,  
SADR = HR2-4,  
~~Hop Count = (any integer x: 0 < x < 255),~~  
I-Could-Be-Router-To-Network,  
Network Number = 4,  
Performance Index = 6

[Change **Clause 10.2.2.7.2**, p. 324]

#### 10.2.2.7.2 Unknown Network Layer Message Type

Purpose: To verify that the IUT will reject a network layer message ~~with an unknown message type~~ directed to the IUT that contains an unknown message type in the range of message types reserved for use by ASHRAE.

Test Steps:

1. TRANSMIT PORT A,  
DESTINATION = IUT,  
SOURCE = ~~D1A~~ TD,  
Message Type = (any value from X'0A' to X'7F in the range reserved for use by ASHRAE that is undefined in the protocol revision claimed by the device)

2. RECEIVE PORT A,  
DESTINATION = ~~D1A~~ TD,  
SOURCE = IUT,  
Reject-Message-To-Network,  
Reject Reason = 3 (unknown network layer message type),  
DNET = ~~+~~ any value

[Change Clause 10.2.3.2, pp. 325]

### 10.2.3.2 Route Message from a Local Device to a Local Device

Purpose: To verify that the IUT can route a unicast message from a local device on Network 1 to a device on Network 2.

Test Steps:

1. TRANSMIT PORT A,  
~~DESTINATION~~ DA = IUT,  
~~SOURCE~~ SA = D1A,  
DNET = 2,  
DADR = D2C,  
Hop Count = 255,  
BACnet-Confirmed-Request-PDU,  
'Service Choice' = ReadProperty-Request,  
'Object Identifier' = (any object identifier),  
'Property Identifier' = (any property of the specified object)
2. RECEIVE PORT B,  
~~DESTINATION~~ DA = D2C,  
~~SOURCE~~ SA = IUT,  
SNET = 1,  
SADR = D1A,  
~~Hop Count = (any integer x: 0 < x < 255),~~  
BACnet-Confirmed-Request-PDU,  
'Service Choice' = ReadProperty-Request,  
'Object Identifier' = (the object identifier used in step 1),  
'Property Identifier' = (the property identifier used in step 1)
3. TRANSMIT PORT B,  
~~DESTINATION~~ DA = IUT,  
~~SOURCE~~ SA = D2C,  
DNET = 1,  
DADR = D1A,  
Hop Count = 255,  
BACnet-Confirmed-Request-PDU,  
'Service Choice' = ReadProperty-Request,  
'Object Identifier' = (any object identifier),  
'Property Identifier' = (any property of the specified object)
4. RECEIVE PORT A,  
~~DESTINATION~~ DA = D1A,  
~~SOURCE~~ SA = IUT,  
SNET = 2,  
SADR = D2C,  
~~Hop Count = (any integer x: 0 < x < 255),~~  
BACnet-Confirmed-Request-PDU,  
'Service Choice' = ReadProperty-Request,  
'Object Identifier' = (the object identifier used in step ~~+~~ 3),  
'Property Identifier' = (the property identifier used in step ~~+~~ 3)

[Change **Clause 10.2.3.5**, p. 327]

#### **10.2.3.5 Route Message from a Router to a Local Device**

Purpose: To verify that the IUT can route a unicast message from a peer router to the destination device on a local network.

Test Steps:

1. TRANSMIT PORT A,  
~~DESTINATION~~  $DA = IUT$ ,  
~~SOURCE~~  $SA = R1-5$ ,  
 $DNET = 2$ ,  
 $DADR = D2C$ ,  
 $SNET = 5$ ,  
 $SADR = D5F$ ,  
 $Hop\ Count = 254$ ,  
BACnet-Confirmed-Request-PDU,  
'Service Choice' = ReadProperty-Request,  
'Object Identifier' = (any object identifier),  
'Property Identifier' = (any property of the specified object)
2. RECEIVE PORT B,  
~~DESTINATION~~  $DA = D2C$ ,  
~~SOURCE~~  $SA = IUT$ ,  
 $SNET = 5$ ,  
 $SADR = D5F$ ,  
~~Hop Count = (any integer x:  $0 < x < 254$ ),~~  
BACnet-Confirmed-Request-PDU,  
'Service Choice' = ReadProperty-Request,  
'Object Identifier' = (the object identifier used in step 1),  
'Property Identifier' = (the property identifier used in step 1)

[Change **Clause 10.2.4.4**, p. 330]

#### **10.2.4.4 Remote Broadcast from a Local Device to a Directly-Connected Network**

Purpose: To verify that the IUT properly forwards remote broadcast messages that originate on a local network and are directed to another local network.

Test Steps:

1. TRANSMIT PORT B,  
~~DESTINATION~~  $DA = LOCAL\ BROADCAST$ ,  
~~SOURCE~~  $SA = D2C$ ,  
 $DNET = 1$ ,  
 $DLEN = 0$ ,  
 $Hop\ Count = 255$ ,  
BACnet-Unconfirmed-Request-PDU,  
'Service Choice' = Who-Is
2. RECEIVE PORT A,  
~~DESTINATION~~  $DA = LOCAL\ BROADCAST$ ,  
~~SOURCE~~  $SA = IUT$ ,  
 $SNET = 2$ ,  
 $SADR = D2C$ ,  
~~Hop Count = (any integer x:  $0 < x < 255$ ),~~

BACnet-Unconfirmed-Request-PDU,  
'Service Choice' = Who-Is

[Change **Clause 10.2.4.6**, p. 331]

#### **10.2.4.6 Remote Broadcast from a Remote Device to a Directly-Connected Network**

Purpose: To verify that the IUT properly forwards remote broadcast messages that originate on a remote network and are directed to a directly-connected network.

Test Steps:

1. TRANSMIT PORT B,  
~~DESTINATION~~ *DA* = IUT,  
~~SOURCE~~ *SA* = R2-3,  
~~DNET~~ = 1,  
~~DLEN~~ = 0,  
~~SNET~~ = 3,  
~~SADR~~ = D3D,  
~~Hop Count~~ = 254,  
BACnet-Unconfirmed-Request-PDU,  
'Service Choice' = Who-Is
2. RECEIVE PORT A,  
~~DESTINATION~~ *DA* = LOCAL BROADCAST,  
~~SOURCE~~ *SA* = IUT,  
~~SNET~~ = 3,  
~~SADR~~ = D3D,  
~~Hop Count~~ = (any integer *x*,:  $0 < x < 254$ ),  
BACnet-Unconfirmed-Request-PDU,  
'Service Choice' = Who-Is

[Change **Clause 10.2.4.8**, p. 332]

#### **10.2.4.8 Remote Broadcast that Should Be Ignored**

Purpose: To verify that the IUT ignores broadcast messages ~~that are~~ intended for a remote network that is reachable through the same port ~~that~~ the message was received from.

Test Steps:

1. TRANSMIT PORT B,  
~~DESTINATION~~ *DA* = LOCAL BROADCAST,  
~~SOURCE~~ *SA* = D2C,  
~~DNET~~ = 3,  
~~DLEN~~ = (any valid MAC address length) 0,  
~~DADR~~ = (any valid MAC address that agrees with DLEN),  
~~Hop Count~~ = 255,  
BACnet-Unconfirmed-Request-PDU,  
'Service Choice' = Who-Is  
ReadProperty-Request,  
'Object Identifier' = \_\_\_\_\_ (any BACnet object),  
'Property Identifier' = \_\_\_\_\_ (any property of the specified object)
2. CHECK (verify that the IUT does not forward this message)

[Change **Clause 10.2.6**, pp. 333]

## 10.2.6 Network Layer Priority

BACnet Reference Clauses: 6.1, 6.2.2, and 6.5.4.

Purpose: To verify that the IUT can process and forward messages with all network priorities.

Test Steps:

1. TRANSMIT PORT A,  
~~DESTINATION~~  $DA = IUT$ ,  
~~SOURCE~~  $SA = D1A$ ,  
 Priority = B'00',  
 DNET = 2,  
 DADR = D2C,  
 Hop Count = 255,  
 BACnet-Confirmed-Request-PDU,  
 'Service Choice' = ReadProperty-Request,  
 'Object Identifier' = (any object identifier),  
 'Property Identifier' = (any property of the specified object)
2. RECEIVE PORT B,  
~~DESTINATION~~  $DA = D2C$ ,  
~~SOURCE~~  $SA = IUT$ ,  
 Priority = B'00',  
 SNET = 1,  
 SDR = D1A,  
~~Hop Count = (any integer x:  $0 < x < 255$ ),~~  
 BACnet-Confirmed-Request-PDU,  
 'Service Choice' = ReadProperty-Request,  
 'Object Identifier' = (the object identifier used in step 1),  
 'Property Identifier' = (the property identifier used in step 1)
3. TRANSMIT PORT A,  
~~DESTINATION~~  $DA = IUT$ ,  
~~SOURCE~~  $SA = D1A$ ,  
 Priority = B'01',  
 DNET = 2,  
 DADR = D2C,  
 Hop Count = 255,  
 BACnet-Confirmed-Request-PDU,  
 'Service Choice' = ReadProperty-Request,  
 'Object Identifier' = (any object identifier),  
 'Property Identifier' = (any property of the specified object)
4. RECEIVE PORT B,  
~~DESTINATION~~  $DA = D2C$ ,  
~~SOURCE~~  $SA = IUT$ ,  
 Priority = B'01',  
 SNET = 1,  
 SDR = D1A,  
~~Hop Count = (any integer x:  $0 < x < 255$ ),~~  
 BACnet-Confirmed-Request-PDU,  
 'Service Choice' = ReadProperty-Request,  
 'Object Identifier' = (the object identifier used in step 3),  
 'Property Identifier' = (the property identifier used in step 3)
5. TRANSMIT PORT A,  
~~DESTINATION~~  $DA = IUT$ ,

- ~~SOURCE~~  $SA = D1A$ ,  
 Priority = B'10',  
 DNET = 2,  
 DADR = D2C,  
 Hop Count = 255,  
 BACnet-Confirmed-Request-PDU,  
 'Service Choice' = ReadProperty-Request,  
 'Object Identifier' = (any object identifier),  
 'Property Identifier' = (any property of the specified object)
6. RECEIVE PORT B,  
~~DESTINATION~~  $DA = D2C$ ,  
~~SOURCE~~  $SA = IUT$ ,  
 Priority = B'10',  
 SNET = 1,  
 SDR = D1A,  
~~Hop Count = (any integer  $x$ :  $0 < x < 255$ )~~,  
 BACnet-Confirmed-Request-PDU,  
 'Service Choice' = ReadProperty-Request,  
 'Object Identifier' = (the object identifier used in step 5),  
 'Property Identifier' = (the property identifier used in step 5)
7. TRANSMIT PORT A,  
~~DESTINATION~~  $DA = IUT$ ,  
~~SOURCE~~  $SA = D1A$ ,  
 Priority = B'11',  
 DNET = 2,  
 DADR = D2C,  
 Hop Count = 255,  
 BACnet-Confirmed-Request-PDU,  
 'Service Choice' = ReadProperty-Request,  
 'Object Identifier' = (any object identifier),  
 'Property Identifier' = (any property of the specified object)
8. RECEIVE PORT B,  
~~DESTINATION~~  $DA = D2C$ ,  
~~SOURCE~~  $SA = IUT$ ,  
 Priority = B'11',  
 SNET = 1,  
 SDR = D1A,  
~~Hop Count = (any integer  $x$ :  $0 < x < 255$ )~~,  
 BACnet-Confirmed-Request-PDU,  
 'Service Choice' = ReadProperty-Request,  
 'Object Identifier' = (the object identifier used in step 7),  
 'Property Identifier' = (the property identifier used in step 7)

### 135.1-2009g-4 Change the Ignore Process ID Test.

#### Rationale

Tests need to be updated as a result of changes made by Addendum 135-2004b-8.

[Delete **Clause 9.1.2.2**, Unsuccessful Alarm Acknowledgment of Confirmed Event Notifications Because the 'Acknowledging Process Identifier' is Invalid, p 196, renumbering subsequent clauses]

[Insert new **Clause 9.1.1.X1**, p.194, renumbering subsequent clauses]

#### **9.1.1.X1 Successful Alarm Acknowledgment of Confirmed Event Notifications Using an Unknown 'Acknowledging Process Identifier' Parameter**

Purpose: To verify the successful acknowledgment of an alarm signaled by a ConfirmedEventNotification when the acknowledgement contains a mismatched or unknown 'Acknowledging Process Identifier' parameter.

Test Concept: An alarm is triggered that causes the IUT to notify the TD and at least one other device. The TD acknowledges the alarm with a mismatched 'Acknowledging Process Identifier' (the Process Identifier associated with another recipient), or an unknown 'Acknowledging Process Identifier' (a Process Identifier not associated with any recipient), and verifies that the acknowledgment is properly noted by the IUT. This test should be performed twice, once with a mismatched Process Identifier and once with an unknown Process Identifier.

Configuration Requirements: The IUT shall be configured with at least one object, Object1, that can detect alarm conditions and send confirmed notifications. The Acked\_Transitions property shall have the value (TRUE,TRUE,TRUE), indicating that all transitions have been acknowledged. The TD and at least one other BACnet device shall be recipients of the alarm notification and shall use different Process Identifiers.

This test is recommended for all BACnet devices that execute AcknowledgeAlarm but is required only for those that claim conformance to Protocol\_Revision 5 or greater.

#### Test Steps:

1. VERIFY (Object1), Acked\_Transitions = (TRUE,TRUE,TRUE)
2. MAKE (a change that triggers the detection of an alarm event in the IUT)
3. BEFORE **Notification-Fail-Time**
  - RECEIVE ConfirmedEventNotification-Request,
 

|                                  |                                                     |
|----------------------------------|-----------------------------------------------------|
| 'Process Identifier' =           | (any Process ID),                                   |
| 'Initiating Device Identifier' = | IUT,                                                |
| 'Event Object Identifier' =      | Object1,                                            |
| 'Time Stamp' =                   | (the current time or sequence number),              |
| 'Notification Class' =           | (the Notification Class configured for this event), |
| 'Priority' =                     | (the priority configured for this event),           |
| 'Event Type' =                   | (any valid event type),                             |
| 'Notify Type' =                  | ALARM or EVENT,                                     |
| 'AckRequired' =                  | TRUE,                                               |
| 'From State' =                   | (any appropriate event state),                      |
| 'To State' =                     | (any appropriate event state),                      |
| 'Event Values' =                 | (values appropriate to the event type)              |
4. TRANSMIT BACnet-SimpleACK-PDU
5. RECEIVE
  - DESTINATION = (at least one device other than the TD),
  - SOURCE = IUT,
  - ConfirmedEventNotification-Request,
  - 'Process Identifier' = (any Process ID),

- |  |                                  |                                                     |
|--|----------------------------------|-----------------------------------------------------|
|  | 'Initiating Device Identifier' = | IUT,                                                |
|  | 'Event Object Identifier' =      | Object1,                                            |
|  | 'Time Stamp' =                   | (the current time or sequence number),              |
|  | 'Notification Class' =           | (the notification class configured for this event), |
|  | 'Priority' =                     | (the priority configured for this event),           |
|  | 'Event Type' =                   | (any valid event type),                             |
|  | 'Notify Type' =                  | ALARM   EVENT,                                      |
|  | 'AckRequired' =                  | TRUE,                                               |
|  | 'From State' =                   | (any appropriate event state),                      |
|  | 'To State' =                     | (any appropriate event state),                      |
|  | 'Event Values' =                 | (values appropriate to the event type)              |
6. TRANSMIT
 

|                                   |  |
|-----------------------------------|--|
| DESTINATION = IUT,                |  |
| SOURCE = (DESTINATION in step 5), |  |
| BACnet-SimpleACK-PDU              |  |
  7. VERIFY (Object1), Acked\_Transitions = (one bit FALSE, the others TRUE)
  8. TRANSMIT AcknowledgeAlarm-Request,
 

|                                      |                                                                        |
|--------------------------------------|------------------------------------------------------------------------|
| 'Acknowledging Process Identifier' = | (Any mismatched or unknown value),                                     |
| 'Event Object Identifier' =          | Object1,                                                               |
| 'Event State Acknowledged' =         | (the state specified in the 'To State' parameter of the notification), |
|                                      | (the timestamp conveyed in the notification),                          |
|                                      | (the current timestamp)                                                |
  9. RECEIVE BACnet-SimpleACK-PDU
  10. BEFORE **Notification-Fail-Time**

|                                             |                                                     |
|---------------------------------------------|-----------------------------------------------------|
| RECEIVE ConfirmedEventNotification-Request, |                                                     |
| 'Process Identifier' =                      | (any Process ID),                                   |
| 'Initiating Device Identifier' =            | IUT,                                                |
| 'Event Object Identifier' =                 | Object1,                                            |
| 'Time Stamp' =                              | (the current time or sequence number),              |
| 'Notification Class' =                      | (the Notification Class configured for this event), |
| 'Priority' =                                | (the priority configured for this event),           |
| 'Event Type' =                              | (any valid event type),                             |
| 'Notify Type' =                             | ACK_NOTIFICATION,                                   |
| 'To State' =                                | (any appropriate event state)                       |
  11. TRANSMIT BACnet-SimpleACK-PDU
  12. RECEIVE
 

|                                                        |                                                     |
|--------------------------------------------------------|-----------------------------------------------------|
| DESTINATION = (at least one device other than the TD), |                                                     |
| SOURCE = IUT,                                          |                                                     |
| ConfirmedEventNotification-Request,                    |                                                     |
| 'Process Identifier' =                                 | (any Process ID),                                   |
| 'Initiating Device Identifier' =                       | IUT,                                                |
| 'Event Object Identifier' =                            | Object1,                                            |
| 'Time Stamp' =                                         | (the current time or sequence number),              |
| 'Notification Class' =                                 | (the notification class configured for this event), |
| 'Priority' =                                           | (the priority configured for this event),           |
| 'Event Type' =                                         | (any valid event type),                             |
| 'Notify Type' =                                        | ACK_NOTIFICATION,                                   |
| 'To State' =                                           | (any appropriate event state)                       |
  13. TRANSMIT
 

|                                   |  |
|-----------------------------------|--|
| DESTINATION = IUT,                |  |
| SOURCE = (DESTINATION in step 5), |  |
| BACnet-SimpleACK-PDU              |  |
  14. VERIFY (Object1), Acked\_Transitions = (TRUE,TRUE,TRUE)

Notes to Tester: The ConfirmedEventNotification-Request messages can be received in either order.

[Insert new **Clause 9.1.1.X2**, p.194]

### **9.1.1.X2 Successful Alarm Acknowledgment of Unconfirmed Event Notifications Using an Unknown 'Acknowledging Process Identifier' Parameter**

**Purpose:** To verify the successful acknowledgment of an alarm signaled by an UnconfirmedEventNotification when the acknowledgement contains a mismatched or unknown 'Acknowledging Process Identifier' parameter.

**Test Concept:** An alarm is triggered that causes the IUT to notify the TD and at least one other device. The TD acknowledges the alarm with a mismatched 'Acknowledging Process Identifier' (the Process Identifier associated with another recipient), or unknown (a Process Identifier not associated with any recipient), and verifies that the acknowledgment is properly noted by the IUT. This test should be performed twice, once with a mismatched Process Identifier and once with an unknown Process Identifier.

**Configuration Requirements:** The IUT shall be configured with at least one object, Object1, that can detect alarm conditions and send unconfirmed notifications. The Acked\_Transitions property shall have the value (TRUE,TRUE,TRUE), indicating that all transitions have been acknowledged. The TD and at least one other BACnet device shall be recipients of the alarm notification, configured to receive different Process Identifiers.

This test is recommended for all BACnet devices that execute AcknowledgeAlarm but is required only for those that claim conformance to Protocol\_Revision 5 or greater.

**Test Steps:**

1. VERIFY (Object1), Acked\_Transitions = (TRUE,TRUE,TRUE)
2. MAKE (a change that triggers the detection of an alarm event in the IUT)
3. BEFORE **Notification-Fail-Time**
  - RECEIVE UnconfirmedEventNotification-Request,
 

|                                  |                                                     |
|----------------------------------|-----------------------------------------------------|
| 'Process Identifier' =           | (any Process ID),                                   |
| 'Initiating Device Identifier' = | IUT,                                                |
| 'Event Object Identifier' =      | Object1,                                            |
| 'Time Stamp' =                   | (the current time or sequence number),              |
| 'Notification Class' =           | (the Notification Class configured for this event), |
| 'Priority' =                     | (the priority configured for this event),           |
| 'Event Type' =                   | (any valid event type),                             |
| 'Notify Type' =                  | ALARM or EVENT,                                     |
| 'AckRequired' =                  | TRUE,                                               |
| 'From State' =                   | (any appropriate event state),                      |
| 'To State' =                     | (any appropriate event state),                      |
| 'Event Values' =                 | (values appropriate to the event type)              |
  - RECEIVE
 

|               |                                          |
|---------------|------------------------------------------|
| DESTINATION = | (at least one device other than the TD), |
| SOURCE =      | IUT,                                     |

 UnconfirmedEventNotification-Request,
 

|                                  |                                                     |
|----------------------------------|-----------------------------------------------------|
| 'Process Identifier' =           | (any Process ID),                                   |
| 'Initiating Device Identifier' = | IUT,                                                |
| 'Event Object Identifier' =      | (the object detecting the alarm),                   |
| 'Time Stamp' =                   | (the current time or sequence number),              |
| 'Notification Class' =           | (the notification class configured for this event), |
| 'Priority' =                     | (the priority configured for this event),           |
| 'Event Type' =                   | (any valid event type),                             |
| 'Notify Type' =                  | ALARM   EVENT,                                      |
| 'AckRequired' =                  | TRUE,                                               |
| 'From State' =                   | (any appropriate event state),                      |
| 'To State' =                     | (any appropriate event state),                      |

- |    |                                                        |                                                                        |
|----|--------------------------------------------------------|------------------------------------------------------------------------|
|    | 'Event Values' =                                       | (values appropriate to the event type)                                 |
| 5. | VERIFY (Object1), Acked_Transitions =                  | (one bit FALSE, the others TRUE)                                       |
| 6. | TRANSMIT AcknowledgeAlarm-Request,                     |                                                                        |
|    | 'Acknowledging Process Identifier' =                   | (Any mismatched or unknown value),                                     |
|    | 'Event Object Identifier' =                            | Object1,                                                               |
|    | 'Event State Acknowledged' =                           | (the state specified in the 'To State' parameter of the notification), |
|    | 'Time Stamp' =                                         | (the timestamp conveyed in the notification),                          |
|    | 'Time of Acknowledgment' =                             | (the current timestamp)                                                |
| 7. | RECEIVE BACnet-SimpleACK-PDU                           |                                                                        |
| 8. | <b>BEFORE Notification-Fail-Time</b>                   |                                                                        |
|    | RECEIVE UnconfirmedEventNotification-Request,          |                                                                        |
|    | 'Process Identifier' =                                 | (any Process ID),                                                      |
|    | 'Initiating Device Identifier' =                       | IUT,                                                                   |
|    | 'Event Object Identifier' =                            | Object1,                                                               |
|    | 'Time Stamp' =                                         | (the current time or sequence number),                                 |
|    | 'Notification Class' =                                 | (the Notification Class configured for this event),                    |
|    | 'Priority' =                                           | (the priority configured for this event),                              |
|    | 'Event Type' =                                         | (any valid event type),                                                |
|    | 'Notify Type' =                                        | ACK_NOTIFICATION,                                                      |
|    | 'To State' =                                           | (any appropriate event state)                                          |
|    | RECEIVE                                                |                                                                        |
|    | DESTINATION = (at least one device other than the TD), |                                                                        |
|    | SOURCE = IUT,                                          |                                                                        |
|    | UnconfirmedEventNotification-Request,                  |                                                                        |
|    | 'Process Identifier' =                                 | (any Process ID),                                                      |
|    | 'Initiating Device Identifier' =                       | IUT,                                                                   |
|    | 'Event Object Identifier' =                            | Object1,                                                               |
|    | 'Time Stamp' =                                         | (the current time or sequence number),                                 |
|    | 'Notification Class' =                                 | (the notification class configured for this event),                    |
|    | 'Priority' =                                           | (the priority configured for this event),                              |
|    | 'Event Type' =                                         | (any valid event type),                                                |
|    | 'Notify Type' =                                        | ACK_NOTIFICATION,                                                      |
|    | 'To State' =                                           | (any appropriate event state)                                          |
| 9. | VERIFY (Object1), Acked_Transitions = (TRUE,TRUE,TRUE) |                                                                        |

Note to Tester: The UnconfirmedEventNotification-Request messages can be received in either order.

### 135.1-2009g-5 Add Max Info Frames Check.

#### Rationale

There is no test that explicitly tests max-info-frame functionality. This test verifies that devices do not hog the token for too long.

[Add new **Clause 12.1.1.9.X**, p.389]

#### 12.1.1.9.X Max Info Frame Check

Dependencies: None

BACnet Reference Clauses: 9.5.3 and 9.5.6.5

Purpose: This check verifies that the MS/TP Master Node State Machine does not issue more than  $N_{\text{max\_info\_frames}}$  information frames between the time the IUT receives a Token and either the time it passes the Token or it initiates a Poll For Master. Unlike tests, checks are not constructed of test steps, but rather conditions that must hold true through the complete testing process. As such, checks are periodically verified during or after the execution of tests.

Configuration Recommendations: If the Max\_Info\_Frames property of the Device object is configurable, it is recommended that this property be set to its minimum setting for the performance of at least some tests involving the MS/TP port being tested.

Check conditions: Monitor the MS/TP LAN during operations where the IUT would be expected to issue a number of information frames; if the IUT emits more information frames than:

- a) the configured value for Max\_Info\_Frames in the interval between receiving and passing the Token (with multiple masters on the LAN), or
  - b) the configured value for Max\_Info\_Frames in the interval between receiving the Token and issuing PFM (with multiple masters on the LAN), or
  - c) the configured value for Max\_Info\_Frames in the interval between any two consecutive Poll For Master frames except the interval between the issuance of a Poll For Master to (TS-1) modulo Max\_Master and a Poll For Master to (TS+1) modulo Max\_Master, (with the IUT as the only master on the LAN), or
  - d) 52 times the configured value for Max\_Info\_Frames in the interval between a Poll For Master frame issued to (TS-1) modulo Max\_Master, and the subsequent Poll For Master frame to (TS+1) modulo Max\_Master (with the IUT as the only master on the LAN),
- then the IUT shall fail this check.

Note to Tester: The value 52 is used in d) because an error in the MS/TP state machine originally defined in Standard 135-1995 caused the Token to be passed 52 times between Poll For Master cycles, instead of 50 times.

### 135.1-2009g-6 Add Test for Device Identifier Recipients.

#### Rationale

This test should be added because no relevant test for this function exists in ASHRAE Standard 135.1.

[Add new **Clause 7.3.2.21.3.X**, p. 80]

#### 7.3.2.21.3.X Recipient\_List Property Supports Device Identifier Recipients Test

**Purpose:** To verify that the Recipient\_List property of the Notification Class object supports the device form of the Recipient component and that the IUT is able to associate a MAC address with the Device Identifier. The intent is to ensure that the IUT is able to locate the specified alarm recipient and send notification to the specified recipient. This test shall be run if the IUT's Notification Class object's Recipient\_List property supports the BACnet object identifier form of BACnetRecipient.

**Test Concept:** The tester shall select a single event-generating object E in the IUT that references Notification Class object N. The tester shall add an entry into the Recipient\_List of the associated Notification Class object that specifies a Device Identifier, D, for a device that the IUT is not already aware of. The TD, acting as device D, shall be located on a different network than the IUT to ensure that the IUT is capable of binding to recipients located on any network.

**Configuration Requirements:** The TD shall be configured so that it does not execute WhoHas.

#### Test Steps:

1. WRITE N.RecipientList = ( {all days, all times, D, any process ID, FALSE, all transitions} )
2. MAKE (the event generating object, E, transition)
3. BEFORE **Notification Fail Time** plus the amount of time the IUT takes to perform device discovery
  - RECEIVE UnconfirmedEventNotification-Request,
    - 'Process Identifier' = (the valid process ID from step 1),
    - 'Initiating Device Identifier' = IUT,
    - 'Event Object Identifier' = E,
    - 'Time Stamp' = (any valid time stamp),
    - 'Notification Class' = (N's instance),
    - 'Priority' = (any valid priority),
    - 'Event Type' = (any valid event type),
    - 'Notify Type' = ALARM | EVENT,
    - 'AckRequired' = TRUE | FALSE,
    - 'From State' = (any valid event state),
    - 'To State' = (any valid event state),
    - 'Event Values' = (values appropriate to the event type)

**Notes to Tester:** The IUT is expected to initiate one or more range-restricted WhoIs requests after the modification of the Recipient\_List but before the sending of the notification. The IUT might also need to perform other network discovery operations. Given that there are multiple approaches to the use of WhoIs for device discovery, the test only focuses on the IUT's ability to find device D and not on the specifics or timing of the WhoIs requests.

### 135.1-2009g-7 Add Test for Network Address Recipients.

#### Rationale

This test should be added because no relevant test for this function exists in ASHRAE Standard 135.1.

[Add new **Clause 7.3.2.21.3.X**, p. 80]

#### 7.3.2.21.3.X Recipient\_List Property Supports Network Address Recipients

**Purpose:** To verify that the Recipient\_List property of the Notification Class object supports the address form of the Recipient component. The intent is to ensure that the IUT is able to send notifications to the specified recipient.

**Test Concept:** The tester shall select a single event-generating object E in the IUT that references Notification Class object N. The tester shall add an entry into the Recipient\_List of the associated Notification Class object that specifies a BACnetAddress A, where A is a unicast or is a local, remote, or global broadcast address.

**Test Steps:**

1. WRITE N.RecipientList = ( {all days, all times, A, any process ID, FALSE, all transitions} )
2. MAKE (the event generating object, E, transition)
3. BEFORE **Notification Fail Time**
  - RECEIVE DESTINATION = A,
  - UnconfirmedEventNotification-Request,
  - 'Process Identifier' = (the valid process ID from step 1),
  - 'Initiating Device Identifier' = IUT,
  - 'Event Object Identifier' = E,
  - 'Time Stamp' = (the current local time),
  - 'Notification Class' = (N's instance),
  - 'Priority' = (any valid priority),
  - 'Event Type' = (any valid event type),
  - 'Notify Type' = ALARM | EVENT,
  - 'AckRequired' = TRUE | FALSE,
  - 'From State' = (any valid event state),
  - 'To State' = (any valid event state),
  - 'Event Values' = (values appropriate to the event type)

### 135.1-2009g-8 Add Tests for Disable Initiation.

#### Rationale

This test should be added because no relevant test for this function exists in ASHRAE Standard 135.1.

[Insert new **Clause 9.24.1.X1**, p 283]

#### 9.24.1.X1 Disable of Service Initiation Restored by Time Duration

BACnet Reference Clause: 16.1.1.1.2.

Purpose: To verify the correct execution of the DeviceCommunicationControl service when DISABLE\_INITIATION is requested with a finite time duration. Communication is restored when the DeviceCommunicationControl 'Time Duration' parameter expires.

Configuration Requirements: The IUT shall be configured to initiate client requests.

#### Test Steps:

1. MAKE (a condition that would normally cause the IUT to initiate requests)
2. CHECK (that the IUT is initiating requests)
3. TRANSMIT DeviceCommunicationControl-Request,  
    'Time Duration' = (a value  $T > 1$ , in minutes, selected by the tester),  
    'Enable/Disable' = DISABLE\_INITIATION,  
    'Password' = (any appropriate password if required)
4. RECEIVE BACnet-SimpleACK-PDU
5. MAKE (a condition that would normally cause the IUT to initiate requests)
6. CHECK (that the IUT has stopped initiating requests)
7. VERIFY (any supported property) = (any valid value)
8. TRANSMIT Who-Is-Request
9. RECEIVE I-Am-Request
10. WAIT (time T)
11. MAKE (a condition that would normally cause the IUT to initiate requests)
12. CHECK (that the IUT is initiating requests)

Notes to Tester: Steps 2 through 9 must be executed within time T.

[ Insert new **Clause 9.24.1.X2**, p 283 ]

#### 9.24.1.X2 Disable of Service Initiation Restored by DeviceCommunicationControl

BACnet Reference Clause: 16.1.1.1.2.

Purpose: To verify the correct execution of the DeviceCommunicationControl service when DISABLE\_INITIATION is requested for a finite time duration. Communication is restored using the DeviceCommunicationControl service.

Configuration Requirements: The IUT shall be configured to initiate client requests.

#### Test Steps:

1. MAKE (a condition that would normally cause the IUT to initiate requests)
2. CHECK (that the IUT is initiating requests)
3. TRANSMIT DeviceCommunicationControl-Request,  
    'Time Duration' = (a value in minutes  $>$  time required to execute all test steps),  
    'Enable/Disable' = DISABLE\_INITIATION,

- 'Password' = (any appropriate password if required)
4. RECEIVE BACnet-SimpleACK-PDU
  5. MAKE (a condition that would normally cause IUT to initiate requests)
  6. CHECK (that the IUT has stopped initiating requests)
  7. VERIFY (any supported property) = (any valid value)
  8. TRANSMIT Who-Is-Request
  9. RECEIVE I-Am-Request
  10. TRANSMIT DeviceCommunicationControl-Request,  
'Enable/Disable' = ENABLE,  
'Password' = (any appropriate password if required)
  11. MAKE (a condition that would normally cause the IUT to initiate requests)
  12. CHECK (that the IUT is initiating requests)

### 135.1-2009g-9 Change Tests for Out\_Of\_Service, Status\_Flags, and Reliability.

#### Rationale

There is a mistake in the Clause 7.3.11 test in 135.1-2009, and one step could be removed.

The test concept states "...If the Reliability property is not supported, then step 4 shall be omitted." This should actually state that "...If the Reliability property is not supported, then step 6 shall be omitted". (Note that in this modified version of the test, step 6 is renumbered to be step 5.)

Step 5 says to "WRITE Present\_Value = (any value that corresponds to an Event\_State of NORMAL)" but then step 6 asks you to "VERIFY Status\_Flags = (?, TRUE, ?, TRUE)". If the In\_Alarm bit of the Status\_Flags property is shown as "?" in step 6, then step 5 can be removed.

This test verifies that when the Out\_Of\_Service property = True, the OUT\_OF\_SERVICE flag of the Status\_Flags property is also true, but it does not test to see that when the Out\_Of\_Service property = False, the OUT\_OF\_SERVICE flag of the Status\_Flags property is also False. This change provides a test for this with the addition of two further steps that parallel steps 2 and 3.

[ Change **Clause 7.3.1.1**, p 32 ]

#### 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~~ step 5 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. WRITE Present\_Value = (any value that corresponds to an Event\_State of NORMAL)~~
- 6-5. 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)
    }
7.6. 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~~ *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.

### 135.1-2009g-10 Add Tests for Non-router Network Layer Messages.

#### Rationale

These tests should be added because no relevant tests exist in ASHRAE Standard 135.1.

[Insert new **Clause 10.X** and its subclauses, p 345 ]

#### 10.X Non-Router Functionality Tests

This subclause defines the tests necessary to demonstrate the portion of BACnet Network Layer functionality that is unique to non-router nodes. These tests verify that the node correctly ignores messages that are reserved for routers. If the IUT can only be configured as a BACnet router, then these tests shall be omitted.

The tests assume that the IUT is located on network DNET1 and the TD appears to be a router to network DNET2. The value DNET3 is assigned a unique network number. If the IUT can initiate requests, it will be configured to send those requests to a device (D2A) on network DNET2. The IUT will also be expected to respond to requests from device D2A. The test descriptions assume that the TD will mimic device D2A.

Note to Tester: Clauses 6.5.1 and 6.5.3 indicate that there are only two ways for a non-router to transmit a request (on the local network and destined for a remote network), neither of which include network layer source routing information. If the IUT emits any NPDU with SNET/SADR fields during the tests in this subclause, then it shall fail.

Note to Tester: Clauses 6.6 and 6.6.3.3 define BACnet routers and the network layer services reserved for routers. If the IUT emits any I-Am-Router-To-Network or I-Could-Be-Router-To-Network NPDUs during the tests in this subclause (including during test 10.X.2), then it shall fail.

##### 10.X.1 Ignore Remote packets

BACnet Reference Clauses: 6.5.2.1, 6.5.4

Purpose: This test case verifies that the non-router IUT will quietly accept and discard packets destined for remote networks.

Test Concept: The TD transmits both broadcast and directed requests to the IUT with DNET (not equal to x'FFFF') and DADR in the Network Layer header. The IUT is required to silently drop the requests because it is not a router.

#### Test Steps:

1. TRANSMIT  
DA = BROADCAST,  
SA = TD,  
DNET = DNET3,  
DADR = BROADCAST,  
Hop Count = 255,  
BACnet-Unconfirmed-Request-PDU,  
'Service Choice' = who-Is
2. WAIT **Internal Processing Fail Time**
3. CHECK (that the IUT did not send an I-Am)
4. TRANSMIT  
DA = IUT,  
SA = TD,  
DNET = DNET3,  
DADR = IUT,  
Hop Count = 255,  
BACnet-Confirmed-Request-PDU,  
'Service Choice' = ReadProperty-Request,  
'Object Identifier' = O2 (any BACnet standard object in IUT),  
'Property Identifier' = P2 (any required property of the specified object)

5. **WAIT Internal Processing Fail Time**
6. **CHECK** (that the IUT did not send a response to the ReadProperty)

### 10.X.2 Ignore Who-Is-Router-To-Network

BACnet Reference Clauses: 6.6, 6.6.3.2

Purpose: This test case verifies that the non-router IUT will quietly accept and discard the Who-Is-Router-To-Network service.

Test Concept: The TD transmits both the general query and the specific network number query form of the Who-Is-Router-To-Network service. The IUT is required to silently drop the requests because it is not a router.

Test Steps:

1. **TRANSMIT**  
DA = BROADCAST,  
SA = TD,  
Who-Is-Router-To-Network,  
DNET = DNET2
2. **WAIT Internal Processing Fail Time**
3. **CHECK** (that the IUT did not send any packets in response to the Who-Is-Router-To-Network)
4. **TRANSMIT**  
DA = BROADCAST,  
SA = TD,  
Who-Is-Router-To-Network
5. **WAIT Internal Processing Fail Time**
6. **CHECK** (that the IUT did not send any packets in response to the Who-Is-Router-To-Network)

### 10.X.3 Ignore Router Commands

BACnet Reference Clauses: 6.6, 6.6.3.8, 6.6.3.10, 6.6.3.11

Purpose: This test case verifies that the non-router IUT will quietly accept and discard network layer router services.

Test Concept: The TD transmits the Initialize-Routing-Table, Establish-Connection-To-Network, and Disconnect-Connection-To-Network services. The IUT is required to silently drop the requests because it is not a router.

Test Steps:

1. **TRANSMIT**  
DA = IUT,  
SA = TD,  
Initialize-Routing-Table  
Number of Ports = 0
2. **WAIT Internal Processing Fail Time**
3. **CHECK** (that the IUT did not send any packets in response to the Initialize-Routing-Table)
4. **TRANSMIT**  
DA = IUT,  
SA = TD,  
Establish-Connection-To-Network  
DNET = DNET3  
Termination Time Value = 0
5. **WAIT Internal Processing Fail Time**
6. **CHECK** (that the IUT did not send any packets in response to the Establish-Connection-To-Network)
7. **TRANSMIT**  
DA = IUT,  
SA = TD,

Disconnect-Connection-To-Network

DNET = NET3

8. WAIT Internal Processing Fail Time
9. CHECK(that the IUT did not send any packets in response to the Disconnect-Connection-To-Network)

[ Insert new **Clause 10.Y** and its subclauses, p 345 ]

## 10.Y Route Binding Tests

This subclause defines the tests necessary to demonstrate the portion of BACnet Network Layer functionality that is used to determine network routes. These tests are generic and are meant to be applied to both router and non-router nodes.

The tests assume that the IUT is located on network DNET1 and the TD appears to be a router to network DNET2. The value DNET3 is assigned a unique network number. If the IUT can initiate requests, it will be configured to send those requests to a device (D2A) on network DNET2. The IUT will also be expected to respond to requests from device D2A. The test descriptions assume that the TD will mimic device D2A.

Note: Clauses 6.5.1 and 6.5.3 indicate that there are only two ways for a non-router to transmit a request (on the local network and destined for a remote network), neither of which includes network layer source routing information. If the IUT is configured as a non-router and it emits any NPDU with SNET/SADR fields during the tests in this subclause, then it shall fail.

Note: Clauses 6.6 and 6.6.3.3 define BACnet routers and the network layer services reserved for routers. If the IUT is configured as a non-router and it emits any I-Am-Router-To-Network or I-Could-Be-Router-To-Network NPDUs during the tests in this subclause (including during test 10.X.2), then it shall fail.

### 10.Y.1 Static Router Binding

Dependencies: ReadProperty Service Initiation Tests, 8.15, ReadProperty Service Execution Tests, 9.15

BACnet Reference Clause: 6.5.3

Purpose: To verify that the IUT can initiate requests to a remote network when the IUT has been statically configured with the MAC address of the router to that remote network.

Test Concept: The IUT transmits a request to a device on the remote network without performing any form of dynamic router binding. If the IUT does not support static router binding or if the IUT cannot initiate a request, then this test shall be omitted. If the IUT cannot initiate a ReadProperty request, then another service can be substituted.

Test Configuration: The IUT is configured with the TD's MAC address as the router for network DNET2.

Test Steps:

1. MAKE (IUT transmit a ReadProperty request to the D2A device on the remote network)
2. RECEIVE
  - DA = TD,
  - SA = IUT,
  - DNET = DNET2,
  - DADR = D2A,
  - Hop Count = 255,
  - BACnet-Confirmed-Request-PDU,
  - 'Service Choice' = ReadProperty-Request,
  - 'Object Identifier' = O1 (any BACnet standard object in D2A),
  - 'Property Identifier' = P1 (any required property of the specified object)
3. TRANSMIT
  - DA = IUT,
  - SA = TD,

SNET = DNET2,  
SADR = D2A,  
BACnet-ComplexACK-PDU,  
'Service ACK Choice' = ReadProperty-ACK,  
'Object Identifier' = O1,  
'Property Identifier' = P1,  
'Property Value' = (any valid value)

## 10.Y.2 Router Binding via Application Layer Services

Dependencies: ReadProperty Service Initiation Tests, 8.18, ReadProperty Service Execution Tests, 9.18, Who-Is Service Initiation Tests, 8.34

BACnet Reference Clause: 6.5.3

Purpose: To verify that the IUT can initiate requests to a remote network and respond to requests from a remote network after the IUT uses the Who-Is and I-Am Application Layer services to discover the MAC address of the router to that remote network.

Test Concept: The IUT broadcasts a Who-Is request to discover device D2A and notes the MAC address of the intervening router in the corresponding I-Am reply. The TD transmits a request to a device on the remote network and responds to a request from the remote network without performing any further form of dynamic router binding. If the IUT does not support application layer router binding, then this test shall be omitted. If the IUT cannot initiate a ReadProperty request, then another confirmed service can be substituted. The IUT may use the deviceInstanceRange form of Who-Is.

Clause 6.5.3 specifically mentions router binding via Who-Is and does not mention router binding by initiating other application layer services (such as Who-Has) or by lurking and noting the router MAC addresses for incoming application layer requests. For this reason the test only allows for router binding via Who-Is.

Test Steps:

1. MAKE (IUT transmit Who-Is to discover the device on the remote network)
2. RECEIVE
  - DA = BROADCAST,
  - SA = IUT,
  - DNET = GLOBAL BROADCAST,
  - Hop Count = 255,
  - BACnet-Unconfirmed-Request-PDU,
  - 'Service Choice' = who-Is
3. TRANSMIT
  - DA = BROADCAST,
  - SA = TD,
  - SNET = DNET2,
  - SADR = D2A,
  - BACnet-Unconfirmed-Request-PDU,
  - 'Service Choice' = I-Am,
  - 'I Am Device Identifier' = (device object, instance number of D2A),
  - 'Max APDU Length Accepted' = (any valid value),
  - 'Segmentation Supported' = (any valid value),
  - 'Vendor ID' = (any valid value)

4. MAKE (IUT transmit a ReadProperty request to the D2A device on the remote network)
5. RECEIVE
  - DA = TD,
  - SA = IUT,
  - DNET = DNET2,
  - DADR = D2A,
  - Hop Count = 255,
  - BACnet-Confirmed-Request-PDU,
  - 'Service Choice' = ReadProperty-Request,
  - 'Object Identifier' = O1 (any BACnet standard object in D2A),
  - 'Property Identifier' = P1 (any required property of the specified object)
6. TRANSMIT
  - DA = IUT,
  - SA = TD,
  - SNET = DNET2,
  - SADR = D2A,
  - BACnet-ComplexACK-PDU,
  - 'Service ACK Choice' = ReadProperty-ACK,
  - 'Object Identifier' = O1,
  - 'Property Identifier' = P1,
  - 'Property Value' = (any valid value)

### 10.Y.3 Router Binding via Who-Is-Router-To-Network

Dependencies: ReadProperty Service Initiation Tests, 8.15, ReadProperty Service Execution Tests, 9.15, Locating Routers, 10.5.1

BACnet Reference Clause: 6.5.3

Purpose: To verify that the IUT can initiate requests to a remote network after the IUT uses the Who-Is-Router-To-Network Network Layer service to discover the MAC address of the router to that remote network.

Test Concept: The IUT broadcasts a Who-Is-Router-To-Network request to discover the router to the desired network. The TD transmits a request to a device on the remote network without performing any further form of dynamic router binding. If the IUT does not support Who-Is-Router-To-Network router binding, then this test shall be omitted. If the IUT cannot initiate a ReadProperty request, then another confirmed service can be substituted. The IUT may use either the general query or specific network number query form of the Who-Is-Router-To-Network service.

Note that Clause 6.5.3 specifically mentions router binding via Who-Is-Router-To-Network and does not mention router binding by lurking and noting unsolicited I-Am-Router-To-Network messages.

Test Steps:

1. MAKE (IUT transmit Who-Is-Router-To-Network to discover the router to DNET2)
2. RECEIVE
  - DA = BROADCAST,
  - SA = IUT,
  - Who-Is-Router-To-Network,
  - DA = BROADCAST,
  - SA = IUT,
  - Who-Is-Router-To-Network,
  - DNET = DNET2
3. TRANSMIT
  - DESTINATION = BROADCAST,
  - SOURCE = TD,
  - I-Am-Router-To-Network,
  - Network Numbers = DNET2

4. MAKE (IUT transmit a ReadProperty request to the D2A device on the remote network)
5. RECEIVE
  - DA = TD,
  - SA = IUT,
  - DNET = DNET2,
  - DADR = D2A,
  - Hop Count = 255,
  - BACnet-Confirmed-Request-PDU,
  - 'Service Choice' = ReadProperty-Request,
  - 'Object Identifier' = O1 (any BACnet standard object in D2A),
  - 'Property Identifier' = P1 (any required property of the specified object)
6. TRANSMIT
  - DA = IUT,
  - SA = TD,
  - SNET = DNET2,
  - SADR = D2A,
  - BACnet-ComplexACK-PDU,
  - 'Service ACK Choice' = ReadProperty-ACK,
  - 'Object Identifier' = O1,
  - 'Property Identifier' = P1,
  - 'Property Value' = (any valid value)

#### 10.Y.4 Router Binding via Broadcast

Dependencies: ReadProperty Service Initiation Tests, 8.15, ReadProperty Service Execution Tests, 9.15

BACnet Reference Clause: 6.5.3

Purpose: To verify that the IUT can initiate requests to a remote network when the IUT uses an initial broadcast to discover the MAC address of the router to that remote network.

Test Concept: The IUT broadcasts a request to a device on the remote network and notes the MAC address of the intervening router in the reply. If the IUT does not support router binding via broadcast, then this test shall be omitted. If the IUT cannot initiate a ReadProperty request, then another confirmed service can be substituted.

Test Steps:

1. MAKE (IUT transmit a ReadProperty request to the D2A device on the remote network)
2. RECEIVE
  - DA = BROADCAST,
  - SA = IUT,
  - DNET = DNET2,
  - DADR = D2A,
  - Hop Count = 255,
  - BACnet-Confirmed-Request-PDU,
  - 'Service Choice' = ReadProperty-Request,
  - 'Object Identifier' = O1 (any BACnet standard object in D2A),
  - 'Property Identifier' = P1 (any required property of the specified object)
3. TRANSMIT
  - DA = IUT,
  - SA = TD,
  - SNET = DNET2,
  - SADR = D2A,
  - BACnet-ComplexACK-PDU,
  - 'Service ACK Choice' = ReadProperty-ACK,
  - 'Object Identifier' = O1,
  - 'Property Identifier' = P1,

- 'Property Value' = (any valid value)
4. MAKE (IUT transmit a ReadProperty request to the D2A device on the remote network)
5. RECEIVE
- DA = TD,  
SA = IUT,  
DNET = DNET2,  
DADR = D2A,  
Hop Count = 255,  
BACnet-Confirmed-Request-PDU,  
'Service Choice' = ReadProperty-Request,  
'Object Identifier' = O1 (any BACnet standard object in D2A),  
'Property Identifier' = P1 (any required property of the specified object)
6. TRANSMIT
- DA = IUT,  
SA = TD,  
SNET = DNET2,  
SADR = D2A,  
BACnet-ComplexACK-PDU,  
'Service ACK Choice' = ReadProperty-ACK,  
'Object Identifier' = O1,  
'Property Identifier' = P1,  
'Property Value' = (any valid value)

### 135.1-2009g-11 Remove Time Delay in TO-FAULT Tests

#### Rationale

The BACnet Test Standard ANSI/ASHRAE 135.1-2009 defines a number of tests for TO-FAULT event transitions. In all these tests, Time\_Delay is respected when transitioning to a FAULT event state, but in BACnet Standard 135-2008, where transitions to FAULT are independent of Time\_Delay, these transitions take place immediately when the conditions are met.

[Change **Clause 7.3.1.10 Event\_Enable Tests**, Modify Test Step 11, p. 39]

```
...
11. IF (the event-triggering object can be placed into a fault condition) THEN {
    MAKE (the event-triggering object change to a fault condition)
    WAIT (Time_Delay)
    BEFORE Notification Fail Time
    ...
```

[Change **Clause 7.3.2.21.1 Priority Tests**, Remove Test Steps 13 and 17, Renumber following steps, p. 74]

```
...
11. IF (the event-triggering object can be placed into a fault condition) THEN {
12.     MAKE (the event-triggering object change to a fault condition)
13. WAIT (Time_Delay)
14./13.    BEFORE Notification Fail Time
            RECEIVE ConfirmedEventNotification-Request,
                'Process Identifier' = (any valid process ID),
                'Initiating Device Identifier' = IUT,
                'Event Object Identifier' = (the event-generating object configured for this test),
                'Time Stamp' = (any valid time stamp),
                'Notification Class' = (the class corresponding to the object being tested),
                'Priority' = (the value configured to correspond to a TO-FAULT transition),
                'Event Type' = (any valid event type),
                'Notify Type' = EVENT | ALARM,
                'AckRequired' = TRUE | FALSE,
                'From State' = NORMAL,
                'To State' = FAULT,
                'Event Values' = (values appropriate to the event type)
15./14.    VERIFY Event_State = FAULT
16./15.    MAKE (the event-triggering object change to a normal condition)
17. WAIT (Time_Delay)
18./16.    BEFORE Notification Fail Time
            RECEIVE ConfirmedEventNotification-Request,
            ...
```

[Renumber original steps 14 to 19 to become steps 13 to 17]

[Change **Clause 7.3.2.21.3.4 Transitions Test**, Modify Test Step 11, p. 80]

```
...
11. IF (the event-triggering object can be placed into a fault condition) THEN {
    MAKE (the event-triggering object change to a fault condition)
```

```

_____ WAIT (Time_Delay)
    BEFORE Notification Fail Time
        IF (the Transitions bit corresponding to the TO-FAULT transition is TRUE) THEN
            ...

```

[Change **Clause 8.4.2 CHANGE\_OF\_STATE Test**, Remove Test Step 18, Renumber following steps, p. 119]

```

...
16. IF (the object being tested is a multi-state object that supports intrinsic reporting) THEN
17.     IF (Present_Value is writable) THEN
            WRITE Present_Value = (a value x: x = one of the Fault_Values)
        ELSE
            MAKE (Present_Value have a value x: x = one of the Fault_Values)
18. _____ WAIT (Time_Delay)
19:18.    BEFORE Notification Fail Time
            RECEIVE ConfirmedEventNotification-Request,
            ...

25:24.    IF (Present_Value is writable) THEN
            WRITE Present_Value = (a value x: x corresponds to a NORMAL state)
        ELSE
            MAKE (Present_Value have a value x: x corresponds to a NORMAL state)
26. _____ WAIT (Time_Delay)
27:25.    BEFORE Notification Fail Time
            RECEIVE ConfirmedEventNotification-Request,
            ...

```

[Renumber original steps 19 to 31 to become steps 18 to 29]

[Change **Clause 8.4.8.14 TO-FAULT Transition Tests**, Remove Test Step 3, Renumber following steps, p. 148]

```

...
2. MAKE (Present_Value have a value x such that x corresponds to a FAULT state)
3. _____ WAIT Time_Delay
4.3. BEFORE Notification Fail Time
5:4.    RECEIVE ConfirmedEventNotification-Request,
            ...
10:9. MAKE (Present_Value have a value x such that x corresponds to a NORMAL state)
11. _____ WAIT Time_Delay
12:10. BEFORE Notification Fail Time
            ...

```

[Renumber original steps 4 to 17 to become steps 3 to 15]

## 135.1-2009g-12 Make Additions to the TCSL Language.

### Rationale

The TCSL as described by 135.1 can be improved to make the tests more readable through the addition and modification of some of the statements.

[Change **Clause 6.1.1**, p 24 ]

### 6.1.1 Common Symbols and Characters

The following definitions are used to represent common symbols and characters.

```
<binary digit> ::= '0' | '1'
<decimal digit> ::= '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'
<hex digit> ::= <decimal digit> | 'A' | 'B' | 'C' | 'D' | 'E' | 'F'
<single quote> ::= (the single quote character)
<double quote> ::= (the double quote character)
<conditional> ::= '=' | '<' | '>' | '<=' | '>=' | '~=' | '<>'
```

*The conditional '~=' is approximately equal to. The definition of approximate is dependant on the context of its use.*

[ Change **Clause 6.2**, p 25 ]

### 6.2 TCSL Statements

Statements in TCSL fall into two categories, those that control the flow and order of tests and those that tell the TD to send data, receive data, or both.

```
<statement> ::= <simple statement> | <compound statement>
<compound statement> ::= '{' <statement>... '}'
```

A <compound statement> can be used anywhere a <simple statement> can be used.

```
<simple statement> ::= <if statement> | <repeat statement> | <error statement>
| <say statement> | <check statement> | <make statement>
| <transmit statement> | <receive statement>
| <write statement> | <verify statement> | <before statement>
| <read statement>
```

[Change **Clause 6.2.10**, p 29]

### 6.2.10 VERIFY Statement

```
<verify statement> ::= VERIFY [ <object identifier> ',' ] <property identifier> '=' <conditional>
<property value> [ ',' ARRAY INDEX '=' <array index> ]
```

The verify *that the* procedure consists of the following steps:

1. WAIT **Internal Processing Fail Time**
2. TRANSMIT ReadProperty-Request,
  - 'Object Identifier' = <object identifier>,
  - 'Property Identifier' = <property identifier>,
  - 'Property Array Index' = <array index>
3. RECEIVE BACnet-ComplexACK-PDU,
  - 'Object Identifier' = <object identifier>,

'Property Identifier' = <property identifier>,  
'Property Array Index' = <array index>,  
'Property Value' = (any valid value *x*, where *x* <conditional> <property value> is TRUE  
subject to the resolution constraints of *Clause 4.4.2*)

Example: WRITE (Analog Output, 1), Present\_Value = 6.5, PRIORITY = 8  
VERIFY (Analog Output, 1), Present\_Value = 6.5

[ Insert new **Clause 6.2.13**, p 29 ]

### 6.2.13 READ Statement

<read statement> ::= READ <variable> '=' [ <object identifier> ',' ] <property identifier>  
[ ',' ARRAY INDEX '=' <array index> ]

The read procedure consists of the following steps:

1. WAIT **Internal Processing Fail Time**
2. TRANSMIT ReadProperty-Request,  
'Object Identifier' = <object identifier>,  
'Property Identifier' = <property identifier>,  
'Property Array Index' = <array index>
3. RECEIVE BACnet-ComplexACK-PDU,  
'Object Identifier' = <object identifier>,  
'Property Identifier' = <property identifier>,  
'Property Array Index' = <array index>,  
'Property Value' = (any valid value, <variable>)

Example: READ X = Priority\_Array, ARRAY\_INDEX=10

[ Insert new **Clause 6.5**, p 30 ]

### 6.5 TD Requirements

The tests defined in this test standard describe specific conversations between the TD and the IUT. While these test conversations take place, the IUT may initiate other conversations related to or unrelated to the current test. Unless otherwise noted in the test, for conversations initiated by the IUT that are not reflected in the test steps of the test, the TD shall generate a correct response consistent with the device capabilities and state that the TD is emulating.

## 135.1-2009g-13 Changes to Acknowledge Alarm Initiation Tests

### Rationale

The existing tests in Standard 135.1 need clarification.

[Change **Clause 8.1**, 106]

### 8.1 Acknowledge Alarm Service Initiation Tests

Dependencies: None.

BACnet Reference Clause: 13.5.

~~Purpose: To verify that the IUT can initiate an AcknowledgeAlarm service request.~~

*Purpose: To verify that the IUT is capable of acknowledging alarms and events that are reported to the IUT via the ConfirmedEventNotification and UnconfirmedEventNotification services.*

*Configuration: For this test, the tester shall choose 1 object, O1, in the TD, which is configured to send event notifications to the IUT. The tester places O1 into an alarm state such that the transition requires an acknowledgment.*

Test Steps:

1. TRANSMIT ConfirmedEventNotification-Request | UnconfirmedEventNotification-Request,
 

|                                   |                                                                                                                                     |
|-----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| 'Subscriber Process Identifier' = | <del>(any value selected by the TD);</del> (a value acceptable to the IUT configured in the Notification Class object for the IUT), |
| 'Initiating Device Identifier' =  | <del>(any value selected by the TD);</del> TD,                                                                                      |
| 'Event Object Identifier' =       | <del>(any value selected by the TD);</del> O1,                                                                                      |
| 'Time Stamp' =                    | <del>(any value selected by the TD);</del> (any valid value, T1),                                                                   |
| 'Notification Class' =            | <del>(any value selected by the TD);</del> (the value configured in O1),                                                            |
| 'Priority' =                      | (any value selected by the TD),                                                                                                     |
| 'Event Type' =                    | (any value selected by the TD),                                                                                                     |
| 'Notify Type' =                   | ALARM   EVENT,                                                                                                                      |
| 'AckRequired' =                   | TRUE,                                                                                                                               |
| 'From State' =                    | <del>NORMAL</del> (any valid value),                                                                                                |
| 'To State' =                      | <del>(any offnormal state appropriate to the event type);</del> (any valid value, S1)                                               |
| 'Event Values' =                  | (any event values appropriate to the event type)                                                                                    |
2. IF (the ConfirmedEventNotification choice was selected) THEN  
RECEIVE BACnet-SimpleACK-PDU
3. MAKE ~~(the IUT initiate an AcknowledgeAlarm service request with parameters appropriate to the event notification the IUT acknowledge O1)~~
4. RECEIVE AcknowledgeAlarm-Request,
 

|                                      |                                                                         |
|--------------------------------------|-------------------------------------------------------------------------|
| 'Acknowledging Process Identifier' = | (any process identifier),                                               |
| 'Event Object Identifier' =          | <del>(the 'Event Object Identifier' from the notification O1),</del>    |
| 'Event State Acknowledged' =         | S1,                                                                     |
| 'Time Stamp' =                       | <del>(the 'Time Stamp' from the notification T1),</del>                 |
| 'Acknowledgement Source' =           | (any CharacterString any valid value),                                  |
| 'Time of Acknowledgement' =          | <del>(the current time as determined by the IUT)</del> any valid value) |
5. TRANSMIT BACnet-SimpleACK-PDU

## 135.1-2009g-14 Add New Tests for Reading and Presenting Properties

### Rationale

Tests are needed for B-OWS functionality.

[Insert new **Clause 8.18.3**, p 165]

### 8.18.3 Reading and Presenting Properties

**Purpose:** This test case verifies that the IUT is capable of reading and presenting properties. It is a generic test used to test data presentation requirements.

**Configuration:** For this test, the tester shall choose a property, P1, from an object, O1. The TD shall be configured to not support the execution of ReadPropertyMultiple or the initiation of COV notifications.

**Test Steps:**

1. MAKE (the IUT read P1)
2. RECEIVE ReadProperty-Request,  
    'Object Identifier' = O1,  
    'Property Identifier' = P1
3. TRANSMIT BACnet-ComplexACK-PDU,  
    'Object Identifier' = O1,  
    'Property Identifier' = P1,  
    'Property Value' = (any valid value for P1)
4. CHECK (that the IUT presents a value that is consistent with the value received in step 3)

**Notes to Tester:** The value presented by the IUT may differ from the value transmitted on the wire due to rounding, truncation, formatting, language conversion, etc.

**Notes to Tester:** If the property being read and displayed is an array, the IUT may include an Array Index parameter in the ReadProperty-Request in step 2 and the TD will include it in the response in step 4.

**Notes to Tester:** If the IUT has not already determined that the TD does not support execution of ReadPropertyMultiple, SubscribeCOV, and SubscribeCOVProperty, the IUT may initiate any of these services. If this occurs, the IUT shall pass the test only if it automatically falls back to using ReadProperty upon receipt of the correct BACnetReject-PDU(s) from the TD, indicating that other service(s) is not supported.

[ Insert new **Clause 8.22.4**, p 169]

### 8.22.4 Accepting Input and Modifying Properties

**Purpose:** This test case verifies that the IUT is capable of accepting user input and using it to modify properties. It is a generic test used to test data-input requirements.

**Configuration:** For this test, the tester shall choose a property, P1, from an object, O1. The TD shall be configured to not support execution of WritePropertyMultiple.

**Test Steps:**

1. MAKE (the IUT accept a new value for P1 from the user)
2. RECEIVE WriteProperty-Request,  
    'Object Identifier' = O1,  
    'Property Identifier' = P1

'Property Value' = (the value provided to the IUT for P1)

3. TRANSMIT BACnet-SimpleACK-PDU

Notes to Tester: The value accepted by the IUT may differ from the value transmitted on the wire due to rounding, truncation, formatting, language conversion, etc.

Notes to Tester: If the property being modified is an array element, the IUT may include an Array Index parameter in the WriteProperty-Request in step 2.

Notes to Tester: If the IUT has not already determined that the TD does not support execution of WritePropertyMultiple, the IUT may initiate a WritePropertyMultiple. If this occurs, the IUT shall pass the test only if it automatically falls back to using WriteProperty upon receipt of the correct BACnetReject-PDU from the TD, indicating that WritePropertyMultiple is not supported.

[ Insert new **Clause 8.22.5**, p 169]

### **8.22.5 Accepting Input and Commanding/Relinquishing Properties**

Purpose: This test case verifies that the IUT is capable of accepting user input and using it to modify a commandable property at a specific priority. It also tests that the IUT is capable of relinquishing at that same priority.

Configuration: For this test, the tester shall choose a commandable property, P1, from an object, O1. The TD shall be configured to not support execution of WritePropertyMultiple. PR1 is the specific priority that will be tested.

Test Steps:

1. MAKE (the IUT accept a new value for P1 from the user, to be commanded at priority PR1)
2. RECEIVE WriteProperty-Request,  
    'Object Identifier' = O1,  
    'Property Identifier' = P1,  
    'Priority' = PR1,  
    'Property Value' = (the value provided to the IUT for P1)
3. TRANSMIT BACnet-SimpleACK-PDU
4. MAKE (the IUT relinquish P1 at priority PR1)
5. RECEIVE WriteProperty-Request,  
    'Object Identifier' = O1,  
    'Property Identifier' = P1,  
    'Priority' = PR1,  
    'Property Value' = NULL
6. TRANSMIT BACnet-SimpleACK-PDU

Notes to Tester: The value accepted by the IUT may differ from the value transmitted on the wire due to rounding, truncation, formatting, language conversion, etc.

Notes to Tester: If the IUT has not already determined that the TD does not support execution of WritePropertyMultiple, the IUT may initiate a WritePropertyMultiple. If this occurs, the IUT shall pass the test only if it automatically falls back to using WriteProperty upon receipt of the correct BACnetReject-PDU from the TD, indicating that WritePropertyMultiple is not supported.

### 135.1-2009g-15 Add New Event Notification Tests.

#### Rationale

Tests are needed for B-OWS functionality.

[Insert new **Clause 9.4.5**, p. 206]

#### 9.4.5 ConfirmedEventNotification Simple Presentation

Purpose: This test case verifies that the IUT is capable of minimally displaying ConfirmedEventNotifications.

Configuration: For this test, the tester shall choose one event-generating object, O1.

Test Steps:

1. TRANSMIT ConfirmedEventNotification-Request,
 

|                                  |                                                                     |
|----------------------------------|---------------------------------------------------------------------|
| 'Process Identifier' =           | (a valid process identifier specified by the IUT vendor),           |
| 'Initiating Device Identifier' = | TD,                                                                 |
| 'Event Object Identifier' =      | O1,                                                                 |
| 'Time Stamp' =                   | (current time in any format),                                       |
| 'Notification Class' =           | (any valid notification class),                                     |
| 'Priority' =                     | (any valid priority),                                               |
| 'Event Type' =                   | (any standard event type),                                          |
| 'Message Text' =                 | (any character string),                                             |
| 'Notify Type' =                  | ALARM   EVENT,                                                      |
| 'AckRequired' =                  | TRUE   FALSE,                                                       |
| 'From State' =                   | (state S1, any valid state for this event type),                    |
| 'To State' =                     | (state S2, any valid state for this event type that can follow S1), |
| 'Event Values' =                 | (any values appropriate to the event type)                          |
2. RECEIVE BACnet-SimpleACK-PDU
3. CHECK (that the IUT indicates the notification to the operator and that the indication includes identification of the event generating object or the monitored object, the event timestamp, and the event Message Text)
4. CHECK (that all information indicated to the user is consistent with the information provided in step 1)

Passing Result: The IUT shall truncate the message text if it is longer than the maximum length displayable by the IUT. The IUT is allowed to include characters in the displayed text that indicate the message has been truncated, even if the truncated message is then shorter than 32 characters. The IUT shall not truncate Message Text that is less than or equal to 32 characters in length.

[Insert new **Clause 9.4.6**, p 207]

#### 9.4.6 ConfirmedEventNotification Full Presentation

Purpose: This test case verifies that the IUT is capable of displaying ConfirmedEventNotifications.

Configuration: For this test, the tester shall choose one event generating object, O1.

Test Steps:

1. TRANSMIT ConfirmedEventNotification-Request,
 

|                                  |                                                           |
|----------------------------------|-----------------------------------------------------------|
| 'Process Identifier' =           | (a valid process identifier specified by the IUT vendor), |
| 'Initiating Device Identifier' = | TD,                                                       |
| 'Event Object Identifier' =      | O1,                                                       |
| 'Time Stamp' =                   | (current time in any format),                             |
| 'Notification Class' =           | (any valid notification class),                           |

|                  |                                                                     |
|------------------|---------------------------------------------------------------------|
| 'Priority' =     | (any valid priority),                                               |
| 'Event Type' =   | (any standard event type),                                          |
| 'Message Text' = | (any character string),                                             |
| 'Notify Type' =  | ALARM   EVENT,                                                      |
| 'AckRequired' =  | TRUE   FALSE,                                                       |
| 'From State' =   | (state S1, any valid state for this event type),                    |
| 'To State' =     | (state S2, any valid state for this event type that can follow S2), |
| 'Event Values' = | (any values appropriate to the event type)                          |

2. RECEIVE BACnet-SimpleACK-PDU
3. CHECK (that the IUT indicates the notification to the operator and that the indication includes identification of the event generating object or the monitored object, the event timestamp, the event Message Text, Notification Class, Priority, Notify Type, Ack Required, To State and Event Values)
4. CHECK (that all information indicated to the user is consistent with the information provided in step 1)

Passing Result: The IUT shall truncate the message text if it is longer than the maximum length displayable by the IUT. The IUT is allowed to include characters in the displayed text that indicate the message has been truncated, even if the truncated message is then shorter than 255 characters. The IUT shall not truncate Message Text that is less than or equal to 255 characters in length.

[Insert new **Clause 9.5.1**, p 206]

#### **9.5.1 UnconfirmedEventNotification Simple Presentation**

This test is identical to the one in Clause 9.4.5 ConfirmedEventNotification Simple Presentation, except that an UnconfirmedEventNotification is sent in step 1 and that no BACnet-SimpleACK-PDU is returned by the IUT in step 2.

[Insert new **Clause 9.5.2**, p 206]

#### **9.5.2 UnconfirmedEventNotification Full Presentation**

This test is identical to the one in Clause 9.4.6 ConfirmedEventNotification Simple Presentation, except that an UnconfirmedEventNotification is sent in step 1 and that no BACnet-SimpleACK-PDU is returned by the IUT in step 2.

### 135.1-2009g-16 Update Trending Tests for Revision 3

#### Rationale

A number of tests for Trending functionality need to be updated to reflect changes made in BACnet protocol revision 3 (Addendum 135-2001b) and to correct errors.

[Change **Clause 7.3.2.24.1**, p. 95]

[ Reason for Change: In the previous version of the test, Log\_Enable was being set to TRUE too late such that step 10 could pass incorrectly. ]

#### 7.3.2.24.1 Log\_Enable Test

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

BACnet Reference Clause: 12.25.5.

Purpose: To verify that the Log\_Enable property enables and disables the logging of data by the Trend Log object.

Test Concept: The Trend Log is configured to acquire data by each means (polling and COV subscription) available to the implementation. Log\_Enable is enabled and the collection of one or more records in the Log\_Buffer is confirmed. Log\_Enable is then disabled and non-collection of records is confirmed.

*The COV increment used is either the Client\_COV\_Increment property of the Trend Log or the COV\_Increment property of the monitored object, depending on the configuration of the Trend Log object being tested.*

Configuration Requirements: Start\_Time, if present, shall be configured with a date and time preceding the beginning of the test. Stop\_Time, if ~~present shall present~~, shall be configured with a time that will occur after the completion of the test. Stop\_When\_Full, if configurable, shall be set to FALSE.

Test Steps:

1. WRITE Log\_Enable = FALSE
2. WRITE Record\_Count = 0
3. WAIT ~~Internal Processing Fail Time~~
4. TRANSMIT ReadProperty Request,  
—— 'Object Identifier' = (the object being tested),  
—— 'Property Identifier' = Total\_Record\_Count
5. RECEIVE ReadProperty ACK,  
—— 'Object Identifier' = (the object being tested),  
—— 'Property Identifier' = Total\_Record\_Count  
—— 'Property Value' = (any valid value, X)
6. WRITE Log\_Enable = TRUE
7. WAIT ~~Internal Processing Fail Time~~
8. IF (COV subscription in use) THEN  
—— MAKE (monitored value change more than Client\_COV\_Increment)  
—— ELSE  
—— WAIT (Log\_Interval)
9. WAIT (Notification Fail Time + Internal Processing Fail Time)
10. VERIFY Total\_Record\_Count > (value X returned in step 5)
11. WRITE Log\_Enable = FALSE
12. WAIT ~~Internal Processing Fail Time~~
13. TRANSMIT ReadProperty Request,  
—— 'Object Identifier' = (the object being tested),  
—— 'Property Identifier' = Total\_Record\_Count
14. RECEIVE ReadProperty ACK,

```

———'Object Identifier' = (the object being tested),
———'Property Identifier' = Total_Record_Count
———'Property Value' = (any valid value, X)
15. IF (COV subscription in use) THEN
———MAKE (monitored value change more than Client_COV_Increment)
———ELSE
———WAIT (Log_Interval)
16. WAIT (Notification Fail Time + Internal Processing Fail Time)
17. VERIFY Total_Record_Count = (value X returned in step 14)

```

```

1. READ I = Log_Interval
2. WRITE Log_Enable = FALSE
3. WRITE Record_Count = 0
4. WAIT Internal Processing Fail Time
5. WRITE Log_Enable = TRUE
6. READ X = Total_Record_Count
7. IF (I = 0) THEN
———MAKE (monitored value change more than the COV increment)
———ELSE
———WAIT (I)
8. WAIT (Notification Fail Time + Internal Processing Fail Time)
9. VERIFY Total_Record_Count > X
10. WRITE Log_Enable = FALSE
11. READ Y = Total_Record_Count
12. IF (I = 0) THEN
———MAKE (monitored value change more than the COV increment)
———ELSE
———WAIT (I)
13. WAIT (Notification Fail Time + Internal Processing Fail Time)
14. VERIFY Total_Record_Count = Y

```

[Change **Clause 7.3.2.24.3**, p. 96]

[Reason for Change: In the previous version of the test, Log\_Enable was being set to TRUE too late such that step 10 could pass incorrectly. The test is also made clearer through the use of the new READ statement.]

### 7.3.2.24.3 Stop\_Time Test

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

BACnet Reference Clause: 12.25.7.

Purpose: To verify that logging is disabled at the time specified by Stop\_Time.

Test Concept: The Trend Log is configured to acquire data by each means (polling and COV subscription) available to the implementation. The test is begun at some time prior to the time specified in Start\_Time and collection of records is confirmed. Non-collection of records after the time specified by Stop\_Time is then confirmed.

Configuration Requirements: Stop\_Time shall be configured with a date and time such that steps 1 through 9 will be concluded before that time. Start\_Time, if present, shall be configured with a date and time preceding the initiation of the test. Stop\_When\_Full, if configurable, shall be set to FALSE.

Test Steps:

```
1. WRITE Log_Enable = FALSE
```

2. WAIT **Internal Processing Fail Time**

3. WRITE Record\_Count = 0

4. WRITE Log\_Enable = TRUE

[the following test steps have been renumbered appropriately]

4. TRANSMIT ReadProperty Request,

\_\_\_\_\_ 'Object Identifier' = \_\_\_\_\_ (the object being tested),

\_\_\_\_\_ 'Property Identifier' = Total\_Record\_Count

5. RECEIVE ReadProperty ACK,

\_\_\_\_\_ 'Object Identifier' = \_\_\_\_\_ (the object being tested),

\_\_\_\_\_ 'Property Identifier' = Total\_Record\_Count

\_\_\_\_\_ 'Property Value' = \_\_\_\_\_ (any valid value, X)

6. WRITE Log\_Enable = TRUE

5. READ X = Record\_Count

6. WAIT **Internal Processing Fail Time**

7. IF (COV subscription in use) THEN

MAKE (monitored value change more than Client\_COV\_Increment)

ELSE

WAIT (Log\_Interval)

8. WAIT (**Notification Fail Time** + **Internal Processing Fail Time**)

9. VERIFY Total\_Record\_Count > (value X returned in step 5)X

10. WHILE (IUT clock is earlier than Stop\_Time) DO {}

11. WAIT (**Notification Fail Time** + **Internal Processing Fail Time**)

13. TRANSMIT ReadProperty Request,

\_\_\_\_\_ 'Object Identifier' = \_\_\_\_\_ (the object being tested),

\_\_\_\_\_ 'Property Identifier' = Total\_Record\_Count

14. RECEIVE ReadProperty ACK,

\_\_\_\_\_ 'Object Identifier' = \_\_\_\_\_ (the object being tested),

\_\_\_\_\_ 'Property Identifier' = Total\_Record\_Count

\_\_\_\_\_ 'Property Value' = \_\_\_\_\_ (any valid value, X)

12. READ X = Total\_Record\_Count

13. IF (COV subscription in use) THEN

MAKE (monitored value change more than Client\_COV\_Increment)

ELSE

WAIT (Log\_Interval)

14. WAIT (**Notification Fail Time** + **Internal Processing Fail Time**)

15. VERIFY Total\_Record\_Count = (value X returned in step 14)X

[Change **Clause 7.3.2.24.5**, p. 98]

### 7.3.2.24.5 COV\_Resubscription\_Interval Test

Dependencies: Confirmed Notifications Subscription, 8.10.1; ~~Unconfirmed Notifications Subscription, 8.10.2.~~

BACnet Reference Clause: 12.25.10.

Purpose: To verify that a Trend Log acquiring data via COV notification reissues its subscription at the interval set by COV\_Resubscription\_Interval.

Test Concept: The Trend Log is configured to acquire data from the TD by COV notification. The TD verifies the resubscription interval.

(a) Configuration Requirements: Start\_Time, if present, shall be configured with a date and time preceding the beginning of the test. Stop\_Time, if present, shall be configured with the latest possible date and time, in order that it occur after the end of the test. Stop\_When\_Full, if configurable, shall be set to FALSE. Log\_Enable shall be set to TRUE. Non-zero values shall be chosen for COV\_Resubscription\_Interval in accordance with the range and resolution specified by the manufacturer for this property.

# Test Steps:

```

1. IF (the IUT uses SubscribeCOV) THEN
  RECEIVE SubscribeCOV Request,
  'Subscriber Process Identifier' = (any value);
  'Monitored Object Identifier' = (the object to be monitored);
  'Issue Confirmed Notifications' = (TRUE or FALSE);
  'Lifetime' = (2 * COV_Resubscription_Interval)
ELSE
  RECEIVE SubscribeCOVProperty Request,
  'Subscriber Process Identifier' = (any value);
  'Monitored Object Identifier' = (the object to be monitored);
  'Issue Confirmed Notifications' = (TRUE | FALSE);
  'Lifetime' = (2 * COV_Resubscription_Interval);
  'Monitored Property Identifier' = (the property to be monitored);
  'COV Increment' = (Client_COV_Increment optional)
2. IF ('Issue Confirmed Notification' = TRUE) THEN
  TRANSMIT ConfirmedCOVNotification Request,
  'Subscriber Process Identifier' = (corresponding value in step 1);
  'Initiating Object Identifier' = (Device object identifier of the TD);
  'Monitored Object Identifier' = (corresponding value in step 1);
  'Issue Confirmed Notifications' = (corresponding value in step 1);
  'Time Remaining' = (2 * COV_Resubscription_Interval);
  'List of Values' = (appropriate BACnetPropertyValue(s))
ELSE
  TRANSMIT UnconfirmedCOVNotification,
  'Subscriber Process identifier' = (corresponding value in step 1);
  'Initiating Object Identifier' = (Device object identifier of the TD);
  'Monitored Object Identifier' = (corresponding value in step 1);
  'Issue Confirmed Notifications' = (corresponding value in step 1);
  'Time Remaining' = (2 * COV_Resubscription_Interval);
  'List of Values' = (appropriate BACnetPropertyValue(s))
3. WAIT (COV_Resubscription_Interval - Notification Fail Time)
4. BEFORE (2 * Notification Fail Time)
  IF (the IUT uses SubscribeCOV)
    RECEIVE SubscribeCOV Request,
    'Subscriber Process Identifier' = (corresponding value in step 1);
    'Monitored Object Identifier' = (corresponding value in step 1);
    'Issue Confirmed Notifications' = (corresponding value in step 1);
    'Lifetime' = (corresponding value in step 1)
  ELSE
    RECEIVE SubscribeCOVProperty Request,
    'Subscriber Process Identifier' = (corresponding value in step 1);
    'Monitored Object Identifier' = (corresponding value in step 1);
    'Issue Confirmed Notifications' = (corresponding value in step 1);
    'Lifetime' = (corresponding value in step 1);
    'Monitored Property Identifier' = (corresponding value in step 1);
    'COV Increment' = (corresponding value in step 1)

1. IF (the IUT uses SubscribeCOV) THEN
  RECEIVE SubscribeCOV-Request,
  'Subscriber Process Identifier' = (any value),
  'Monitored Object Identifier' = (the object to be monitored),
  'Issue Confirmed Notifications' = (TRUE),
  'Lifetime' = (any value >= COV_Resubscription_Interval)

```

```

ELSE
    RECEIVE SubscribeCOVProperty-Request,
        'Subscriber Process Identifier' = (any value),
        'Monitored Object Identifier' = (the object to be monitored),
        'Issue Confirmed Notifications' = (TRUE),
        'Lifetime' = (any value >= COV_Resubscription_Interval),
        'Monitored Property Identifier' = (the property to be monitored),
        'COV Increment' = (Client_COV_Increment -- optional)
2. TRANSMIT BACnet-SimpleACK-PDU
3. TRANSMIT ConfirmedCOVNotification-Request,
    'Subscriber Process Identifier' = (corresponding value in step 1),
    'Initiating Device Identifier' = (Device object identifier of the TD),
    'Monitored Object Identifier' = (corresponding value in step 1),
    'Issue Confirmed Notifications' = (corresponding value in step 1),
    'Time Remaining' = (any value <= the Lifetime from step 1),
    'List of Values' = (appropriate BACnetPropertyValue(s))
4. RECEIVE BACnet-SimpleACK-PDU
5. BEFORE (the lesser of COV_Resubscription_Interval + Re-subscription Interval Tolerance and
    LifeTime from step 1)
    IF (the IUT uses SubscribeCOV)
        RECEIVE SubscribeCOV-Request,
            'Subscriber Process Identifier' = (corresponding value in step 1),
            'Monitored Object Identifier' = (corresponding value in step 1),
            'Issue Confirmed Notifications' = (TRUE),
            'Lifetime' = (any value >= COV_Resubscription_Interval)
    ELSE
        RECEIVE SubscribeCOVProperty-Request,
            'Subscriber Process Identifier' = (corresponding value in step 1),
            'Monitored Object Identifier' = (corresponding value in step 1),
            'Issue Confirmed Notifications' = (TRUE),
            'Lifetime' = (any value >= COV_Resubscription_Interval),
            'Monitored Property Identifier' = (corresponding value in step 1),
            'COV Increment' = (corresponding value in step 1)
6. TRANSMIT BACnet-SimpleACK-PDU
7. TRANSMIT ConfirmedCOVNotification-Request,
    'Subscriber Process Identifier' = (corresponding value in step 1),
    'Initiating Device Identifier' = (Device object identifier of the TD),
    'Monitored Object Identifier' = (corresponding value in step 1),
    'Issue Confirmed Notifications' = (corresponding value in step 1),
    'Time Remaining' = (any value <= the Lifetime from step 5),
    'List of Values' = (appropriate BACnetPropertyValue(s))
8. RECEIVE BACnet-SimpleACK-PDU
9. WAIT (COV_Resubscription_Interval - Re-subscription Interval Tolerance)
10. BEFORE (2 * Re-subscription Interval Tolerance)
    IF (the IUT uses SubscribeCOV)
        RECEIVE SubscribeCOV-Request,
            'Subscriber Process Identifier' = (corresponding value in step 1),
            'Monitored Object Identifier' = (corresponding value in step 1),
            'Issue Confirmed Notifications' = (TRUE),
            'Lifetime' = (corresponding value in step 1)
    ELSE
        RECEIVE SubscribeCOVProperty-Request,
            'Subscriber Process Identifier' = (corresponding value in step 1),
            'Monitored Object Identifier' = (corresponding value in step 1),
            'Issue Confirmed Notifications' = (TRUE),
            'Lifetime' = (corresponding value in step 1),

```

'Monitored Property Identifier' = (corresponding value in step 1),  
'COV Increment' = (corresponding value in step 1)

#### 11. TRANSMIT BACnet-SimpleACK-PDU

*Passing Result: Where the Lifetime parameter of a SubscribeCOV request is less than COV\_Resubscription\_Interval + Re-subscription Interval Tolerance, the IUT shall send the subsequent SubscribeCOV request within Lifetime seconds even though this is a smaller time window than defined by the test. If the IUT does not meet this stricter time window, then the IUT shall fail the test.*

[Change **Clause 7.3.2.24.9**, p. 101]

[Reason for Change: It is not clear whether or not log records should be added to a Trend Log when Record\_Count is set to 0 and the Trend Log is disabled. As such, the test steps were re-worked to ignore this issue.]

#### 7.3.2.24.9 Total\_Record\_Count Test

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

BACnet Reference Clause: 12.25.16.

Purpose: To verify that the Total\_Record\_Count property increments for each record added to the Log\_Buffer, even after Buffer\_Size records have been added. (Note: it is not reasonable to test for the requirement of BACnet Clause 12.23.16 that the value wrap from  $2^{32}-1$  to 0; even if a record was collected every 100<sup>th</sup> of a second it could take more than 497 days to complete the test.)

Test Concept: The Trend Log is configured to acquire data by whatever means. Record\_Count is set to zero and Total\_Record\_Count is read. Collection of data proceeds until Record\_Count changes, collection is halted and Total\_Record\_Count is checked that it has incremented by Record\_Count. If, for whatever reason, the IUT cannot be configured such that the TD is able to halt collection before Buffer\_Size records are collected this test shall not be performed.

Configuration Requirements: Start\_Time, if present, shall be configured with a date and time preceding the beginning of the test. Stop\_Time, if present, shall be configured with the latest possible date and time in order that it occur after the end of the test. Log\_Enable shall be set to FALSE.

Test Steps:

```

1. WRITE Record_Count = 0
2. WAIT Internal Processing Fail Time
3. TRANSMIT ReadProperty Request,
   —— 'Object Identifier' = (the object being tested),
   —— 'Property Identifier' = Total_Record_Count
4. RECEIVE ReadProperty ACK,
   —— 'Object Identifier' = (the object being tested),
   —— 'Property Identifier' = Total_Record_Count,
   —— 'Property Value' = (any valid value, X)
3. READ X = Total_Record_Count
4. READ Y = Record_Count
5. WRITE Log_Enable = TRUE
6. WHILE ( Record_Count = 0Y + 1) DO { }
7. WRITE Log_Enable = FALSE
8. WAIT Internal Processing Fail Time
9. IF (Record_Count = Buffer_Size) THEN
   ERROR "Buffer full; cannot verify Total_Record_Count value."
— ELSE {
— IF ( Total_Record_Count != Record_Count + (value X returned in step 4)) THEN
   IF (Total_Record_Count - X != Record_Count - Y) THEN

```

ERROR "Total\_Record\_Count has incorrect value."

†

[Change **Clause 9.21**, p. 263]

### 9.21 ReadRange Service Execution Tests

This clause defines the tests necessary to demonstrate support for executing ReadRange service requests.

Dependencies: None.

BACnet Reference Clause: 15.8.

Configuration Requirements: The IUT shall be configured with a Trend Log object that contains a set of known trend data. The TD must have exact knowledge of the trend data in order to evaluate the results of the tests. The value of the Log\_Enable property shall be FALSE so that the Log\_Buffer does not change during the tests.

*The following sample buffer is used as explanation for the tests in this section.*

**Sample Log\_Buffer, (Trend Log, Instance 1)**

| <i>Arbitrary<br/>Record<br/>Designation</i> | <i>Position<br/>(index)</i> | <i>Implied<br/>Sequence #</i> | <i>Timestamp<br/>(Date excluded<br/>for clarity)</i> | <i>LogDatum</i>                         |
|---------------------------------------------|-----------------------------|-------------------------------|------------------------------------------------------|-----------------------------------------|
| <i>a</i>                                    | <i>1</i>                    | <i>16</i>                     | <i>13:01:00.00</i>                                   | <i>Log-status, buffer-purged</i>        |
| <i>b</i>                                    | <i>2</i>                    | <i>17</i>                     | <i>13:02:00.00</i>                                   | <i>log-status, log-disabled = FALSE</i> |
| <i>c</i>                                    | <i>3</i>                    | <i>18</i>                     | <i>13:05:00.00</i>                                   | <i>real-value = 5.0</i>                 |
| <i>d</i>                                    | <i>4</i>                    | <i>19</i>                     | <i>13:10:00.00</i>                                   | <i>real-value = 10.0</i>                |
| <i>e</i>                                    | <i>5</i>                    | <i>20</i>                     | <i>13:15:00.00</i>                                   | <i>real-value = 15.0</i>                |
| <i>f</i>                                    | <i>6</i>                    | <i>21</i>                     | <i>13:16:00.00</i>                                   | <i>log-status, log-disabled = TRUE</i>  |
| <i>g</i>                                    | <i>7</i>                    | <i>22</i>                     | <i>13:21:00.00</i>                                   | <i>log-status, log-disabled = FALSE</i> |
| <i>h</i>                                    | <i>8</i>                    | <i>23</i>                     | <i>13:25:00.00</i>                                   | <i>real-value = 25.0</i>                |
| <i>i</i>                                    | <i>9</i>                    | <i>24</i>                     | <i>13:30:00.00</i>                                   | <i>real-value = 30.0</i>                |
| <i>j</i>                                    | <i>10</i>                   | <i>25</i>                     | <i>13:35:00.00</i>                                   | <i>real-value = 35.0</i>                |
| <i>k</i>                                    | <i>11</i>                   | <i>26</i>                     | <i>13:36:00.00</i>                                   | <i>log-status, log-disabled = TRUE</i>  |

[Change **Clause 9.21.1.2**, p. 264]

#### 9.21.1.2 Reading Items by Position with Positive Count

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 after that position to return.

Test Concept: A ReadRange request is transmitted by the TD requesting a range of items known to be in the Log\_Buffer. This range is specified using the 'By Position' option and a positive value for 'Count'. The 'Reference Index' and 'Count' are selected so that the results can be conveyed in a single acknowledgment.

Test Steps:

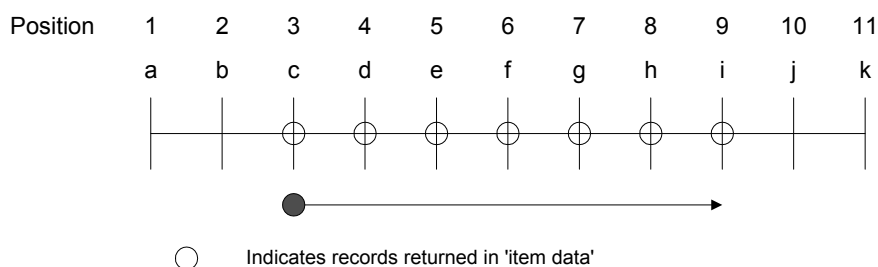
1. TRANSMIT ReadRange-Request,  
'Object Identifier' = (the Trend Log object configured for this test),

'Property Identifier' = Log\_Buffer,  
'Reference Index' = (any value  $x$ :  $1 \leq x \leq \text{Record\_Count}$  ~~(the number of trend records in the buffer — the 'Count' used below)~~),  
'Count' = (any value  $y$ :  $0 < y \leq \text{Record\_Count} - x + 1$ )  ~~$x$ :  $0 < x \leq$  the number of trend records remaining beyond "Reference Index"~~)

2. RECEIVE ReadRange-ACK,  
'Object Identifier' = (the Trend Log object configured for this test),  
'Property Identifier' = Log\_Buffer,  
'Result Flags' = {TRUE, TRUE, FALSE},  
'Item Count' =  $y$  ~~(the same value used in the 'Count' parameter in step 1)~~,  
'Item Data' = (all of the specified trend records in order of increasing position. The items specified include the item at the index specified by  $x$ , plus  $(y-1)$  items following.)

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' = 3,  
'Count' = 7
2. RECEIVE ReadRange-ACK,  
'Object Identifier' = (Trend Log, Instance 1),  
'Property Identifier' = Log\_Buffer,  
'Result Flags' = {FALSE, FALSE, FALSE},  
'Item Count' = 7,  
'Item Data' = Records < c, d, e, f, g, h, i > in that order.



[Change **Clause 9.21.1.3**, p. 264]

### 9.21.1.3 Reading Items by Position with Negative Count

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 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.

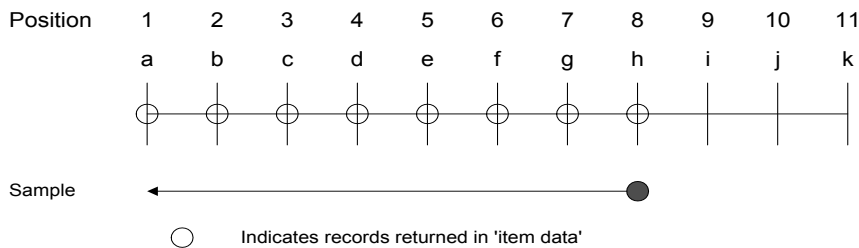
Test Steps:

1. TRANSMIT ReadRange-Request,  
'Object Identifier' = (the Trend Log object configured for this test),  
'Property Identifier' = Log\_Buffer,  
'Reference Index' = (any value  $x$ :  $1 \leq x \leq \text{Record\_Count}$   ~~$2 < x <$  the number of trend records in the buffer~~),

- 'Count' = (any value  $y$ :  $y < 0$  AND  $|y| \leq x$ ;  $x < 0$  AND  $|x| < \text{the 'Reference Index'}$ )
2. ~~RECEIVE ReadRange-ACK~~ ~~Read Range ACK~~,  
 'Object Identifier' = (the Trend Log object configured for this test),  
 'Property Identifier' = Log\_Buffer,  
 'Result Flags' = {~~TRUE~~, ?TRUE, FALSE},  
 'Item Count' =  $|y|$  ~~(the same value used in the 'Count' parameter in step 1)~~,  
 'Item Data' = (all of the 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.)

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  $\langle a, b, c, d, e, f, g, h \rangle$  in that order.



[Change **Clause 9.21.1.4**, p. 264]

#### 9.21.1.4 Reading Items by Time

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

Test Concept: A ReadRange request is transmitted by the TD requesting a range of items known to be in the Log\_Buffer. This range is specified using the 'By Time' option and a positive value for 'Count'. The 'Reference Index Time' and 'Count' are selected so that the results can be conveyed in a single acknowledgement.

Test Steps:

1. TRANSMIT ReadRange-Request,  
     'Object Identifier' = (the Trend Log object configured for this test),  
     'Property Identifier' = Log\_Buffer,  
     'Reference Time' = (any value  $x$ :  $x$  is older (of earlier time) than the time of an entry in the buffer which has a sequence number of  $S$ , and  $x$  is newer than or equal to the time of any preceding entry), ~~(any value older than (earlier time) the last time in the buffer)~~;

'Count' = (any value  $y$ :  $0 < y \leq \text{Total\_Record\_Count} - S + 1$ ) (any value  $> 0$ )

2. ~~RECEIVE ReadRange-ACK~~ ~~ReadRange-ACK~~,

'Object Identifier' = (the Trend Log object configured for this test),

'Property Identifier' = Log\_Buffer,

'Result Flags' = {?TRUE, ?TRUE, FALSE},

'Item Count' =  ~~$y$  (the same value used in the 'Count' parameter in step 1),~~

'Item Data' = (all of the specified trend records in order of increasing sequence number. The items specified include the first item with a timestamp newer than  $x$ , plus  $(y-1)$  items following.)

'First Sequence Number' =  $S$

~~Notes to Tester: The first item returned shall be the entry in the Log\_Buffer with a timestamp newer (later time) than the time specified by the 'Reference Time' parameter.~~

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

1. TRANSMIT ReadRange-Request,

'Object Identifier' = (Trend Log, Instance 1),

'Property Identifier' = Log\_Buffer,

'Reference Time' = 13:21:00.00

'Count' = 4

2. RECEIVE ReadRange-ACK,

'Object Identifier' = (Trend Log, Instance 1),

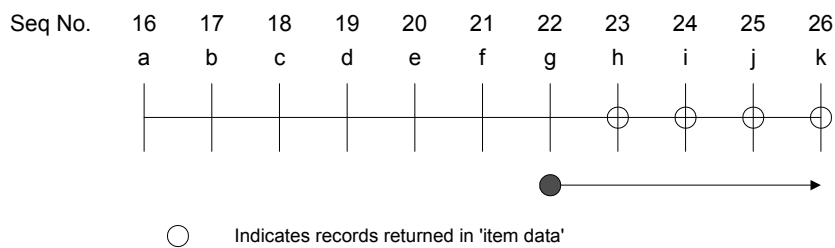
'Property Identifier' = Log\_Buffer,

'Result Flags' = {FALSE, TRUE, FALSE},

'Item Count' = 4,

'Item Data' = Records  $\langle h, i, j, k \rangle$  in that order.

'First Sequence Number' = 23



### 135.1-2009g-17 Add New Tests for Revision 4 Schedules.

#### Rationale

These tests have been added because no tests exist for revision 4 functionality schedules in 135.1.

[Add new **Clause 7.3.2.23.X** and its subclauses, p. 94]

#### 7.3.2.23.X Schedule Object Protocol\_Revision 4 Tests

The Schedule object was revised in Addendum a to ANSI/ASHRAE 135-2001, which increased Protocol\_Revision to 4. Although the basic structure of the Schedule object changed little, its operations are sufficiently different that the existing tests for the original Schedule object need revision in some cases and complete replacement in others, and new tests for some additional changes were needed. This clause presents specific tests to be run for Schedule objects in devices that claim Protocol\_Revision 4 or higher.

The Schedule object has no properties required to be writable or otherwise configurable. The following tests are designed to be performed on such a Schedule object. However, if the Schedule object is in any way configurable, then it shall be configured to accommodate as many of the following tests as is possible for the implementation. If it is impossible to configure the IUT in the manner required for a particular test, then that test shall be omitted. If the IUT supports Schedule objects that can write outside the device this shall be demonstrated in one of the Schedule tests.

Tests of the Schedule object center upon observing the write operations scheduled to occur at specific dates and times, verified by reading the Schedule object's Present\_Value property. For the test to be performed in a reasonable amount of time it is necessary to be able to alter settings of the device's clock and calendar.

For each test, a date and (as required) time ("Date") for the test is determined beforehand. The tables in the Configuration Requirements section of each test give the criteria for the Date/Time value, designated D1, D2, and so on, to be used in the tests. Date/Time values meeting these criteria may be chosen from existing schedules, or a schedule may be developed by the manufacturer to meet these criteria.

Associated with each Date/Time value, D<sub>n</sub>, and defining the time of a schedule write operation is a value V<sub>n</sub>, which is the value associated with the time member of D<sub>n</sub> in the BACnetTimeValue pair. V<sub>n</sub> may take on any primitive datatype.

##### 7.3.2.23.X.1 Revision 4 Effective\_Period Test

**Purpose:** To verify that Effective\_Period controls the range of dates during which the Schedule object is active.

**Test Concept:** Two Date values are chosen by the TD based on the criteria in Table 7-X1 such that one is outside of the Effective\_Period and the other corresponds to a known scheduled state inside the Effective\_Period. The IUT's local date and time are changed between these dates and a property referenced by the List\_Of\_Object\_Property\_References property is monitored to verify that write operations occur only within the Effective\_Period.

**Configuration Requirements:** The IUT shall be configured with a schedule object such that the time periods defined in Table 7-X1 have uniquely scheduled values. The local date and time shall be set such that the Present\_Value property has a value other than V<sub>1</sub>. The List\_Of\_Object\_Property\_References property shall contain at least one reference either to a property within the IUT alterable by the Schedule object or a writable property in another device (in either case: the "referenced property"); if the List\_Of\_Object\_Property\_References property cannot be thus configured, then this test shall be skipped.

**Table 7-X1.** Criteria for Effective Period Test Dates and Values

| <b>Date:</b>   | <b>Criteria:</b>                                                                                                        | <b>Value:</b>                                  |
|----------------|-------------------------------------------------------------------------------------------------------------------------|------------------------------------------------|
| D <sub>1</sub> | 1. Date occurs during Effective_Period, and<br>2. Date appears either in Weekly_Schedule or Exception_Schedule.         | V <sub>1</sub>                                 |
| D <sub>2</sub> | 1. Date does not occur during Effective_Period, and<br>2. Date appears either in Weekly_Schedule or Exception_Schedule. | V <sub>2</sub> different from V <sub>1</sub> . |

Test Steps:

1. VERIFY "referenced property" = (any value other than V<sub>1</sub>)
2. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>1</sub>)  
| (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>1</sub>)  
| MAKE (the local date and time = D<sub>1</sub>)
3. **WAIT Schedule Evaluation Fail Time**
4. VERIFY "referenced property" = V<sub>1</sub>
5. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>2</sub>)  
| (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>2</sub>)  
| MAKE (the local date and time = D<sub>2</sub>)
6. **WAIT Schedule Evaluation Fail Time**
7. VERIFY "referenced property" = V<sub>1</sub>

**7.3.2.22.X.2 Revision 4 Weekly\_Schedule Property Test**

Purpose: To verify that Weekly\_Schedule contains distinguishable schedules for each day of the week and that a day's entire schedule can be executed.

Test Concept: The IUT's local date and time are changed sequentially to represent each day of the week as shown in Table 7-X2. The Present\_Value property is monitored to verify that write operations occur for each separately scheduled day.

Configuration Requirements: The IUT shall be configured with a schedule object containing a weekly schedule with seven distinguishable daily schedules meeting the requirements of Table 7-X2. The local date and time shall be set such that the Present\_Value property has a value other than V<sub>1</sub>. If no schedule exists that meets these requirements and none can be configured, then this test shall be omitted. An "active period" is defined as a period of time when the Exception\_Schedule determines the value appearing in Present\_Value.

**Table 7-X2.** Criteria for Weekly Schedule Test Dates and Values

| <b>Date:</b>   | <b>Criteria:</b>                                                                                                                                       | <b>Value:</b>                                     |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------|
| D <sub>1</sub> | 1. Date occurs during Effective_Period,<br>2. Date occurs on a Monday, and<br>3. Date does not occur during an active period in Exception_Schedule.    | V <sub>1</sub>                                    |
| D <sub>2</sub> | 1. Date occurs during Effective_Period,<br>2. Date occurs on a Tuesday, and<br>3. Date does not occur during an active period in Exception_Schedule.   | V <sub>2</sub> is different from V <sub>1</sub> . |
| D <sub>3</sub> | 1. Date occurs during Effective_Period,<br>2. Date occurs on a Wednesday, and<br>3. Date does not occur during an active period in Exception_Schedule. | V <sub>3</sub> is different from V <sub>2</sub> . |
| D <sub>4</sub> | 1. Date occurs during Effective_Period,<br>2. Date occurs on a Thursday, and<br>3. Date does not occur during an active period in Exception_Schedule.  | V <sub>4</sub> is different from V <sub>3</sub> . |
| D <sub>5</sub> | 1. Date occurs during Effective_Period,<br>2. Date occurs on a Friday, and<br>3. Date does not occur during an active period in Exception_Schedule.    | V <sub>5</sub> is different from V <sub>4</sub> . |
| D <sub>6</sub> | 1. Date occurs during Effective_Period,<br>2. Date occurs on a Saturday, and<br>3. Date does not occur during an active period in Exception_Schedule.  | V <sub>6</sub> is different from V <sub>5</sub> . |
| D <sub>7</sub> | 1. Date occurs during Effective_Period,<br>2. Date occurs on a Sunday, and<br>3. Date does not occur during an active period in Exception_Schedule.    | V <sub>7</sub> is different from V <sub>6</sub> . |

**Test Steps:**

1. VERIFY Present\_Value = (any value other than V<sub>1</sub>)
2. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>1</sub>)  
    | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>1</sub>)  
    | MAKE (the local date and time = D<sub>1</sub>)
3. WAIT Schedule Evaluation Fail Time
4. VERIFY Present\_Value = V<sub>1</sub>
5. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>2</sub>)  
    | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>2</sub>)  
    | MAKE (the local date and time = D<sub>2</sub>)
6. WAIT Schedule Evaluation Fail Time
7. VERIFY Present\_Value = V<sub>2</sub>
8. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>3</sub>)  
    | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>3</sub>)  
    | MAKE (the local date and time = D<sub>3</sub>)
9. WAIT Schedule Evaluation Fail Time
10. VERIFY Present\_Value = V<sub>3</sub>
11. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>4</sub>)  
    | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>4</sub>)  
    | MAKE (the local date and time = D<sub>4</sub>)
12. WAIT Schedule Evaluation Fail Time
13. VERIFY Present\_Value = V<sub>4</sub>
14. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>5</sub>)  
    | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>5</sub>)  
    | MAKE (the local date and time = D<sub>5</sub>)
15. WAIT Schedule Evaluation Fail Time
16. VERIFY Present\_Value = V<sub>5</sub>
17. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>6</sub>)  
    | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>6</sub>)  
    | MAKE (the local date and time = D<sub>6</sub>)
18. WAIT Schedule Evaluation Fail Time

19. VERIFY Present\_Value = V<sub>6</sub>
20. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>7</sub>)  
     | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>7</sub>)  
     | MAKE (the local date and time = D<sub>7</sub>)
21. WAIT Schedule Evaluation Fail Time
22. VERIFY Present\_Value = V<sub>7</sub>
23. REPEAT X = (the time portion of the BACnetTimeValue entries for one of the daily schedules in Table 7-X2) DO {  
     (TRANSMIT TimeSynchronization-Request, 'Time' = X)  
     | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = X)  
     | MAKE (the local date and time = X)  
     WAIT Schedule Evaluation Fail Time  
     VERIFY Present\_Value = (the scheduled value corresponding to time X)  
   }

### 7.3.2.23.X.3 Revision 4 Exception\_Schedule Property Tests

If the IUT cannot be made to meet the configuration requirements of one or more of the tests in this clause, then that test shall be omitted. The inability to make such a configuration may be due to an absent or immutable Exception\_Schedule property to limited numbers of available BACnetSpecialEvent records in the Exception\_Schedule or to the unavailability of Calendar objects.

#### 7.3.2.23.X.3.1 Revision 4 Calendar Reference Test

Purpose: To verify that a date appearing in a referenced Calendar object enables the referencing Schedule object. This test applies to Protocol\_Revision 3 and prior, as well as to Protocol\_Revision 4 and later schedule objects.

Test Concept: The IUT's local date and time are changed to values that are selected by the TD based on the criteria in Table 7-X3. The value of the Present\_Value property is monitored to verify that the scheduled write operations occur.

Configuration Requirements: The IUT shall be configured to contain a Schedule object that references a Calendar object with a non-empty Date\_List. The criteria for the dates used are given in Table 7-X3. The local date and time shall be set such that the Present\_Value property has a value other than V<sub>1</sub>.

**Table 7-X3. Criteria for Calendar Reference Dates and Values**

| Date           | Criteria                                                                                                                                                                                                                                                                                                                                                                                                                   | Value                                        |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------|
| D <sub>1</sub> | 1. Date occurs during Effective_Period,<br>2A. BACnetSpecialEvent references Calendar object via calendarReference,<br>2B. Date appears in that Calendar's Date_List property,<br>2C. Time is on or after the time of the entry with V <sub>1</sub> , but before any other entry in the Exception_Schedule, and<br>2D. BACnetSpecialEvent has a higher eventPriority than any other coincident BACnetSpecialEvent records. | V <sub>1</sub>                               |
| D <sub>2</sub> | 1. Date occurs during Effective_Period,<br>2. Date does not appear in any BACnetSpecialEvent records, and<br>3. Date occurs on the same date as, but with time following, an entry in a BACnetDailySchedule in the referencing Schedule object.                                                                                                                                                                            | V <sub>2</sub> different from V <sub>1</sub> |

Test Steps:

1. VERIFY Present\_Value = (any value other than V<sub>1</sub>)
2. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>1</sub>)  
     | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>1</sub>)  
     | MAKE (the local date and time = D<sub>1</sub>)
3. **WAIT Schedule Evaluation Fail Time**
4. VERIFY Present\_Value = V<sub>1</sub>
5. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>2</sub>)

- | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>2</sub>)
- | MAKE (the local date and time = D<sub>2</sub>)
- 6. **WAIT Schedule Evaluation Fail Time**
- 7. **VERIFY** Present\_Value = V<sub>2</sub>

### 7.3.2.23.X.3.2 Revision 4 Calendar Entry Date Test

**Purpose:** To verify that a specified date appearing in an Exception\_Schedule enables the referencing Schedule object.

**Test Concept:** The IUT's local date and time are changed to values that are selected by the TD based on the criteria in Table 7-X4. The value of the Present\_Value property is monitored to verify that the scheduled write operations occur.

**Configuration Requirements:** The IUT shall be configured to contain a Schedule object, S, with an Exception\_Schedule containing a BACnetCalendarEntry with a specific date. The criteria for the dates used in the test are given in Table 7-X4. The local date and time shall be set such that the Present\_Value property has a value other than V<sub>1</sub>.

**Table 7-X4.** Criteria for Calendar Entry Date Test Dates and Values

| Date           | Criteria                                                                                                                                                                                                                                                                                                                                                                           | Value                                        |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------|
| D <sub>1</sub> | 1. Date occurs during Effective_Period,<br>2A. BACnetSpecialEvent incorporates calendarEntry: Date,<br>2B. Date matches calendarEntry: Date,<br>2C. Time is on or after the time of the entry with V <sub>1</sub> , but before any other entry in the Exception_Schedule, and<br>2D. BACnetSpecialEvent has a higher eventPriority than any coincident BACnetSpecialEvent records. | V <sub>1</sub>                               |
| D <sub>2</sub> | 1. Date occurs during Effective_Period,<br>2. Date does not appear in any BACnetSpecialEvent records, and<br>3. Date occurs on the same date as, but with time following, an entry in a BACnetDailySchedule in the referencing Schedule object.                                                                                                                                    | V <sub>2</sub> different from V <sub>1</sub> |

**Test Steps:**

- 1. **VERIFY** Present\_Value = (any value other than V<sub>1</sub>)
- 2. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>1</sub>) |  
| (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>1</sub>)  
| MAKE (the local date and time = D<sub>1</sub>)
- 3. **WAIT Schedule Evaluation Fail Time**
- 4. **VERIFY** S, Present\_Value = V<sub>1</sub>
- 5. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>2</sub>) |  
| (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>2</sub>)  
| MAKE (the local date and time = D<sub>2</sub>)
- 6. **WAIT Schedule Evaluation Fail Time**
- 7. **VERIFY** S, Present\_Value = V<sub>2</sub>

### 7.3.2.23.X.3.3 Revision 4 Calendar Entry DateRange Test

**Purpose:** To verify that a date appearing in an Exception\_Schedule's date range enables the referencing Schedule object.

**Test Concept:** The IUT's local date and time are changed to values that are selected by the TD based on the criteria in Table 7-X5. The value of the Present\_Value property is monitored to verify that the scheduled write operations occur.

Configuration Requirements: The IUT shall be configured to contain a Schedule object, S, with an Exception\_Schedule containing a BACnetCalendarEntry with a date range. The criteria for the dates used in the test are given in Table 7-X5. The local date and time shall be set such that the Present\_Value property has a value other than V<sub>1</sub>.

**Table 7-X5. Criteria for Calendar Entry DateRange Test Dates and Values**

| <b>Date</b>    | <b>Criteria</b>                                                                                                                                                                                                                                                                                                                                                                                                                                  | <b>Value</b>                                 |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------|
| D <sub>1</sub> | 1. Date occurs during Effective_Period,<br>2A. BACnetSpecialEvent incorporates calendarEntry: DateRange,<br>2B. Date matches BACnetCalendarEntry: DateRange,<br>2C. Time is on or after the time of the entry with V <sub>1</sub> , but before any other entry in the Exception_Schedule,<br>2D. BACnetSpecialEvent has a higher eventPriority than any coincident BACnetSpecialEvent records, and<br>2E. DateRange shall not include wildcards. | V <sub>1</sub>                               |
| D <sub>2</sub> | 1. Date occurs during Effective_Period,<br>2. Date does not appear in any BACnetSpecialEvent records,<br>3. Date occurs on the same date as, but with time following, an entry in a BACnetDailySchedule in the referencing Schedule object, and<br>4. DateRange shall not include wildcards.                                                                                                                                                     | V <sub>2</sub> different from V <sub>1</sub> |

Test Steps:

1. VERIFY (S), Present\_Value = any value other than V<sub>1</sub>
2. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>1</sub>) |  
     | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>1</sub>)  
     MAKE (the local date and time = D<sub>1</sub>)
3. **WAIT Schedule Evaluation Fail Time**
4. VERIFY S, Present\_Value = V<sub>1</sub>
5. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>2</sub>) |  
     | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>2</sub>)  
     MAKE (the local date and time = D<sub>2</sub>)
6. **WAIT Schedule Evaluation Fail Time**
7. VERIFY S, Present\_Value = V<sub>2</sub>

#### **7.3.2.23.X.3.4 Revision 4 Calendar Entry WeekNDay Month Test**

Purpose: To verify that a date matching a WeekNDay's Month field, specifying a specific month in an Exception\_Schedule, enables the referencing Schedule object.

Test Concept: The IUT's local date and time are changed to values that are selected by the TD based on the criteria in Table 7-X6. The value of the Present\_Value property is monitored to verify that the scheduled write operations occur.

Configuration Requirements: The IUT shall be configured to contain a Schedule object, S, with an Exception\_Schedule containing a BACnetCalendarEntry with a WeekNDay entry specifying a specific month, from January (1) to December (12). The criteria for the dates used in the test are given in Table 7-X6. The local date and time shall be set such that the Present\_Value property has a value other than V<sub>1</sub>.

**Table 7-X6. Criteria for Calendar Entry WeekNDay Month Test Dates and Values**

| <b>Date</b>    | <b>Criteria</b>                                                                                                                                                                                                                                                                                                                                                                                                                             | <b>Value</b>                                 |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------|
| D <sub>1</sub> | 1. Date occurs during Effective_Period,<br>2A. BACnetSpecialEvent incorporates calendarEntry: WeekNDay,<br>2B. calendarEntry: WeekNDay: specifies Month,<br>2C. Date matches calendarEntry: WeekNDay,<br>2D. Time is on or after the time of the entry with V <sub>1</sub> , but before any other entry in the Exception_Schedule, and<br>2E. BACnetSpecialEvent has a higher eventPriority than any coincident BACnetSpecialEvent records. | V <sub>1</sub>                               |
| D <sub>2</sub> | 1. Date occurs during Effective_Period,<br>2. Date does not appear in any BACnetSpecialEvent records, and<br>3. Date occurs on the same date as, but with time following, an entry in a BACnetDailySchedule in the referencing Schedule object.                                                                                                                                                                                             | V <sub>2</sub> different from V <sub>1</sub> |

**Test Steps:**

1. VERIFY (S), Present\_Value = any value other than V<sub>1</sub>
2. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>1</sub>) |  
| (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>1</sub>)  
MAKE (the local date and time = D<sub>1</sub>)
3. WAIT **Schedule Evaluation Fail Time**
4. VERIFY S, Present\_Value = V<sub>1</sub>
5. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>2</sub>) |  
| (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>2</sub>)  
MAKE (the local date and time = D<sub>2</sub>)
6. WAIT **Schedule Evaluation Fail Time**
7. VERIFY S, Present\_Value = V<sub>2</sub>

**7.3.2.23.X.3.5 Revision 4 Calendar Entry WeekNDay Week Of Month Test**

**Purpose:** To verify that a date matching a WeekNDay's WeekOfMonth field in an Exception\_Schedule enables the referencing Schedule object.

**Test Concept:** The IUT's local date and time are changed to values that are selected by the TD based on the criteria in Table 7-X7. The value of the Present\_Value property is monitored to verify that the scheduled write operations occur.

**Configuration Requirements:** The IUT shall be configured to contain a Schedule object, S, with an Exception\_Schedule containing a BACnetCalendarEntry with a WeekNDay entry specifying a week of the month. The criteria for the dates used in the test are given in Table 7-X7. The local date and time shall be set such that the Present\_Value property has a value other than V<sub>1</sub>.

**Table 7-X7. Criteria for Calendar Entry WeekNDay Week Of Month Test Dates and Values**

| Date           | Criteria                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | Value                                        |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------|
| D <sub>1</sub> | 1. Date occurs during Effective_Period,<br>2A. BACnetSpecialEvent incorporates calendarEntry: WeekNDay,<br>2B. calendarEntry: WeekNDay specifies WeekOfMonth,<br>2C. calendarEntry: WeekNDay: WeekOfMonth is in the range 1..5,<br>2D. Date matches calendarEntry: WeekNDay,<br>2E. Time is on or after the time of the entry with V <sub>1</sub> , but before any other entry in the Exception_Schedule, and<br>2F. BACnetSpecialEvent has a higher eventPriority than any coincident BACnetSpecialEvent records. | V <sub>1</sub>                               |
| D <sub>2</sub> | 1. Date occurs during Effective_Period,<br>2A. BACnetSpecialEvent incorporates calendarEntry: WeekNDay,<br>2B. calendarEntry: WeekNDay specifies WeekOfMonth,<br>2C. calendarEntry: WeekNDay: WeekOfMonth is in the range 1..5, and<br>2D. Date does not match calendarEntry: WeekNDay: WeekOfMonth.                                                                                                                                                                                                               | V <sub>2</sub> different from V <sub>1</sub> |

**Test Steps:**

1. VERIFY (S), Present\_Value = any value other than V<sub>1</sub>
2. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>1</sub>) |  
 | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>1</sub>)  
 MAKE (the local date and time = D<sub>1</sub>)
3. **WAIT Schedule Evaluation Fail Time**
4. VERIFY S, Present\_Value = V<sub>1</sub>
5. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>2</sub>) |  
 | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>2</sub>)  
 MAKE (the local date and time = D<sub>2</sub>)
6. **WAIT Schedule Evaluation Fail Time**
7. VERIFY S, Present\_Value = V<sub>2</sub>

**7.3.2.23.X.3.6 Revision 4 Calendar Entry WeekNDay Last Week Of Month Test**

**Purpose:** To verify that a date matching a WeekNDay's WeekOfMonth field in an Exception\_Schedule enables the referencing Schedule object.

**Test Concept:** The IUT's local date and time are changed to values that are selected by the TD based on the criteria in Table 7-X8. The value of the Present\_Value property is monitored to verify that the scheduled write operations occur.

**Configuration Requirements:** The IUT shall be configured to contain a Schedule object, S, with an Exception\_Schedule containing a BACnetCalendarEntry with a WeekNDay entry specifying the last week of the month. The criteria for the dates used in the test are given in Table 7-X8. The local date and time shall be set such that the Present\_Value property has a value other than V<sub>1</sub>.

**Table 7-X8. Criteria for Calendar Entry WeekNDay Last Week Of Month Test Dates and Values**

| Date           | Criteria                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | Value                                        |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------|
| D <sub>1</sub> | 1. Date occurs during Effective_Period,<br>2A. BACnetSpecialEvent incorporates calendarEntry: WeekNDay,<br>2B. calendarEntry: WeekNDay specifies WeekOfMonth,<br>2C. calendarEntry: WeekNDay: WeekOfMonth has the value 6,<br>2D. Date is in the last week of the month,<br>2E. WeekNDay:Month matches the specified month,<br>2F. WeekNDay:dayOfWeek matches the specified day of the week,<br>2G. Time is on or after the time of the entry with V <sub>1</sub> , but before any other entry in the Exception_Schedule, and<br>2H. BACnetSpecialEvent has a higher eventPriority than any coincident BACnetSpecialEvent records. | V <sub>1</sub>                               |
| D <sub>2</sub> | 1. Date occurs during Effective_Period,<br>2A. BACnetSpecialEvent incorporates calendarEntry: WeekNDay,<br>2B. calendarEntry: WeekNDay specifies WeekOfMonth,<br>2C. calendarEntry: WeekNDay: WeekOfMonth has the value 6, and<br>2D. Date is not in the last week of the month.                                                                                                                                                                                                                                                                                                                                                   | V <sub>2</sub> different from V <sub>1</sub> |

Test Steps:

1. VERIFY (S), Present\_Value = any value other than V<sub>1</sub>
2. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>1</sub>) |  
 | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>1</sub>)  
 MAKE (the local date and time = D<sub>1</sub>)
3. WAIT **Schedule Evaluation Fail Time**
4. VERIFY S, Present\_Value = V<sub>1</sub>
5. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>2</sub>) |  
 | (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D<sub>2</sub>)  
 MAKE (the local date and time = D<sub>2</sub>)
6. WAIT **Schedule Evaluation Fail Time**
7. VERIFY S, Present\_Value = V<sub>2</sub>

[Delete **Clause 7.3.2.23.3.1**, p. 84, and replace it with the following]

**7.3.2.23.3.1 Calendar Reference Test**

*See Clause 7.3.2.23.X.2.1 Revision 4 Calendar Reference Test.*

### 135.1-2009g-18 Add New Test for Event Notification Network Priority.

#### Rationale

The standard requires that EventNotifications be sent with a specific network priority based on the event priority of the notification. This new test verifies that the IUT does just that.

[ Add new **Clause 8.4.X**, p 149 ]

#### 8.4.X Network Priority Test

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

BACnet Reference Clause: 13.4.1

Purpose: To verify that the correct network priority is used when transmitting EventNotification requests.

Test Concept: The IUT is made to generate a ConfirmedEventNotification with a specific priority. The network priority in the NPCI is checked to ensure that it is correct based on the priority of the event.

Test Configuration: The IUT is configured with an event generating object E1 that refers to Notification Class object N1.

Test Steps:

1. IF (the IUT can generate ConfirmedEventNotifications with an event priority in the range 0..63) THEN
2. MAKE (The IUT initiate a ConfirmedEventNotification-Request with a event priority in the range 0..63)
3. BEFORE **Notification Fail Time**  
RECEIVE ConfirmedEventNotification-Request,  
    'Network Priority' = 3  
    'Process Identifier' = (any valid process ID),  
    'Initiating Device Identifier' = IUT,  
    'Event Object Identifier' = (the intrinsic reporting object or the Event Enrollment object being tested),  
    'Time Stamp' = (the current local time of the IUT),  
    'Notification Class' = (the configured notification class),  
    'Priority' = (a value in the range 0..63),  
    'Event Type' = (any valid value),  
    'Notify Type' = EVENT | ALARM,  
    'AckRequired' = TRUE | FALSE,  
    'From State' = (any valid value),  
    'To State' = (any valid value),  
    'Event Values' = (any valid valid)
4. TRANSMIT BACnet-SimpleACK-PDU
5. IF (the IUT can generate ConfirmedEventNotifications with an event priority in the range 64..127) THEN
6. MAKE (The IUT initiate a ConfirmedEventNotification-Request with an event priority in the range 64..127)
7. BEFORE **Notification Fail Time**  
RECEIVE ConfirmedEventNotification-Request,  
    'Network Priority' = 2  
    'Process Identifier' = (any valid process ID),  
    'Initiating Device Identifier' = IUT,  
    'Event Object Identifier' = (the intrinsic reporting object or the Event Enrollment object being tested),  
    'Time Stamp' = (the current local time of the IUT),  
    'Notification Class' = (the configured notification class),  
    'Priority' = (a value in the range 64..127),  
    'Event Type' = (any valid value),  
    'Notify Type' = EVENT | ALARM,  
    'AckRequired' = TRUE | FALSE,

- 'From State' = (any valid value),
- 'To State' = (any valid value),
- 'Event Values' = (any valid valid)
- 8. TRANSMIT BACnet-SimpleACK-PDU
- 9. IF (the IUT can generate ConfirmedEventNotifications with an event priority in the range 128..191) THEN
- 10. MAKE (The IUT initiate a ConfirmedEventNotification-Request with a event priority in the range 128..191)
- 11. BEFORE **Notification Fail Time**
  - RECEIVE ConfirmedEventNotification-Request,
  - 'Network Priority' = 1
  - 'Process Identifier' = (any valid process ID),
  - 'Initiating Device Identifier' = IUT,
  - 'Event Object Identifier' = (the intrinsic reporting object or the Event Enrollment object being tested),
  - 'Time Stamp' = (the current local time of the IUT),
  - 'Notification Class' = (the configured notification class),
  - 'Priority' = (a value in the range 128..191),
  - 'Event Type' = (any valid value),
  - 'Notify Type' = EVENT | ALARM,
  - 'AckRequired' = TRUE | FALSE,
  - 'From State' = (any valid value),
  - 'To State' = (any valid value),
  - 'Event Values' = (any valid valid)
- 12. TRANSMIT BACnet-SimpleACK-PDU
- 13. IF (the IUT can generate ConfirmedEventNotifications with an event priority in the range 192..255) THEN
- 14. MAKE (The IUT initiate a ConfirmedEventNotification-Request with a event priority in the range 192..255)
- 15. BEFORE **Notification Fail Time**
  - RECEIVE ConfirmedEventNotification-Request,
  - 'Network Priority' = 0
  - 'Process Identifier' = (any valid process ID),
  - 'Initiating Device Identifier' = IUT,
  - 'Event Object Identifier' = (the intrinsic reporting object or the Event Enrollment object being tested),
  - 'Time Stamp' = (the current local time of the IUT),
  - 'Notification Class' = (the configured notification class),
  - 'Priority' = (a value in the range 192..255),
  - 'Event Type' = (any valid value),
  - 'Notify Type' = EVENT | ALARM,
  - 'AckRequired' = TRUE | FALSE,
  - 'From State' = (any valid value),
  - 'To State' = (any valid value),
  - 'Event Values' = (any valid valid)
- 16. TRANSMIT BACnet-SimpleACK-PDU

### 135.1-2009g-19 Add Device and Network Mapping Tests.

#### Rationale

Tests for this functionality are not in 135.1.

[Insert new **Clause 13.X1**, p 456]

#### **13.X1 Automatic Network Mapping**

Purpose: To verify that an IUT can find all devices connected to the BACnet internetwork and present the list of devices to the user.

Test Configuration: The IUT shall be connected to a network consisting of devices that are distributed on multiple networks. The TD shall monitor the IUT's generation of Who-Is service requests and shall verify that the requests cover the complete range of BACnet device instances and that Who-Is requests are used that contain device instance ranges.

Test Steps:

1. IF (the IUT caches device information) THEN  
MAKE (the IUT's cache clear so that is unaware of the existence of any other devices)
2. MAKE (the IUT initiate the network mapping function while monitoring the network for Who-Is requests initiated by the IUT)
3. CHECK (that the IUT identifies all of the devices on the network to the user)

Passing Result: The IUT sends global broadcast Who-Is requests or directed broadcasts to each network, the complete range of device instances is covered by the Who-Is requests, and the IUT does not solely rely on Who-Is requests with no device instance ranges.

Passing Result: The IUT presents a list of all devices on the network to the user. The list shall not indicate devices as present that do not exist on the internetwork. The IUT is not required to include itself in the list.

[Insert new **Clause 13.X2**, p 456]

#### **13.X2 Automatic Device Mapping**

Purpose: To verify that an IUT can find all objects in an arbitrary BACnet device and present the objects to the user.

Configuration: Configure the TD to emulate a device with an arbitrary set of self-consistent BACnet characteristics with a collection of objects in it.

Test Steps:

1. MAKE (the IUT initiate its device mapping function for the TD)
2. CHECK (that the IUT correctly identifies all of the objects in the TD)

### 135.1-2009g-20 Add Device Restart Notification Tests.

#### Rationale

Tests for this functionality are not currently included in 135.1.

[Insert new **Clause 8.3.X**, p. 118]

#### 8.3.X Device Restart Notifications

Purpose: To verify that the IUT initiates UnconfirmedCOVNotification service requests to each entry in its Restart\_Notification\_Recipients property when it resets.

Test Concept: The IUT is configured to send restart notifications and is then reset. The TD checks for the restart notifications.

Device restart notifications differ from subscribed COV notifications that use the UnconfirmedCOVNotification service in two respects. First, subscription is made through the Restart\_Notification\_Recipients property instead of SubscribeCOV. Second, the 'Subscriber Process Identifier' parameter always has a value of zero.

Configuration Requirements: For each Recipient of the Restart\_Notification\_Recipients property in the IUT which is of the device form, there shall be a device on the network that will answer Who-Is requests so that the IUT can determine addressing information before sending the restart notification.

Test Steps:

1. IF (Restart\_Notification\_Recipients is writable) THEN  
    WRITE(Restart\_Notification\_Recipients = any non-empty list of Recipients)  
ELSE  
    MAKE (Restart\_Notification\_Recipients contain any non-empty list of Recipients)
2. READ T1 = Local\_Time
3. MAKE(the IUT reset)
4. REPEAT X = (each entry in the Restart\_Notification\_Recipients) DO {  
    BEFORE **Notification Fail Time**  
    RECEIVE UnconfirmedCOVNotification-Request,  
        DESTINATION = X,  
        'Subscriber Process Identifier' = 0,  
        'Initiating Device Identifier' = IUT,  
        'Monitored Object Identifier' = (the IUT Device Identifier),  
        'Time Remaining' = 0,  
        'List of Values' = (System\_Status=OPERATIONAL,  
                            Time\_Of\_Device\_Restart = (T2),  
                            Last\_Restart\_Reason=(any valid restart reason, R))  
    }  
5. VERIFY Time\_Of\_Device\_Restart = T2
6. CHECK (T1 ~ T2)
7. VERIFY Last\_Restart\_Reason = R

Note to tester: Not all IUTs can accurately differentiate between the types of restart reasons and thus no requirements are placed on the value returned in step 4. The test shall pass regardless of the order in which the IUT generates the UnconfirmedCOVNotification-Requests in step 4. The value of T2 shall be the same for each notification sent out in step 4.

[Change **Clause 9.3**, p 204]

#### 9.3 UnconfirmedCOVNotification Service Execution Tests

~~BACnet does not define a service procedure for executing the UnconfirmedCOVNotification service and thus no tests are needed.~~

### **9.3.1 Device Restart Notifications**

*Purpose: To verify that the IUT executes UnconfirmedCOVNotification service requests that convey restart notifications.*

*Test Concept: The TD sends a restart notification and the tester verifies that the IUT processes the notification in a vendor-specified manner.*

*Test Configuration: A valid BACnetDeviceStatus and a valid Last\_Restart\_Reason pair,  $DS_1$  and  $LRR_1$ , shall be selected for which the IUT will take some action.*

*Device restart notifications differ from subscribed COV notifications that use the UnconfirmedCOVNotification service in two respects. First, subscription is made through the Restart\_Notification\_Recipients property instead of SubscribeCOV. Second, the 'Subscriber Process Identifier' parameter always has a value of zero.*

*Test Steps:*

1. TRANSMIT UnconfirmedCOVNotification-Request,  
    'Subscriber Process Identifier' = 0,  
    'Initiating Device Identifier' = TD,  
    'Monitored Object Identifier' = (the TD Device Identifier),  
    'Time Remaining' = 0,  
    'List of Values' = (System\_Status =  $DS_1$ ,  
                      Time\_Of\_Device\_Restart = (any valid time),  
                      Last\_Restart\_Reason =  $LRR_1$ )
2. CHECK(that the IUT processes the notification as described by the vendor)

### 135.1-2009g-21 Add Schedule Written Datatypes Tests.

#### Rationale

Tests for this functionality are not currently included in 135.1.

[Add new **Clause 7.3.2.23.Y** and its subclauses, p. 94]

#### 7.3.2.23.Y Written Datatypes Tests

The following tests verify that the Schedule properly writes datatypes that it claims to support.

##### 7.3.2.23.Y.1 Internally Written Datatypes Test, non-NULL values

BACnet Reference Clauses: 12.24, 12.24.10

Purpose: This test verifies that the Schedule object within the IUT writes to properties in the same device for the non-NULL datatype being tested.

Test Concept: Two Date/Time, values,  $D_1$  and  $D_2$ , are chosen by the TD based on the criteria in Table 7-Y1 such that  $D_1$  is sufficiently different from, and later than, the current time to cause a Schedule evaluation when the time is changed to  $D_1$ , and such that setting the time to  $D_2$  (later than  $D_1$ ) from  $D_1$  will cause a Schedule evaluation that will cause it to write value  $V_2$ . These values may be chosen based on the Schedule object's existing configuration, or the Schedule object,  $S$ , may be configured with such values.

Configuration Requirements: The IUT shall be configured with a Schedule object,  $S$ , such that the time periods defined in Table 7-Y1 can be configured with uniquely scheduled values. The Schedule object shall be configured with a List\_Of\_Object\_Property\_References, including at least one reference to a writable property within the device. If the IUT cannot be configured to these requirements, then this test shall be omitted. Other properties shall be consistent in both datatypes and values in a manner permitting this test to be executed.

**Table 7-Y1. Criteria for Test Date and Times**

| Date and Time: | Value:                       |
|----------------|------------------------------|
| $D_1$          | $V_1$                        |
| $D_2$          | $V_2$ different from $V_1$ . |

Test Steps:

1. (TRANSMIT TimeSynchronization-Request, 'Time' =  $D_1$ )  
     | (TRANSMIT UTCTimeSynchronization-Request, 'Time'= $D_1$ )  
     | MAKE (the local date and time =  $D_1$ )
2. WAIT(**Schedule\_Evaluation\_Fail\_Time**)
3. VERIFY  $S$ , Present\_Value =  $V_1$
4. REPEAT  $P$  = (writable property in List\_Of\_Property\_References)  
     VERIFY  $P$  =  $V_1$
5. (TRANSMIT TimeSynchronization-Request, 'Time' =  $D_2$ )  
     | (TRANSMIT UTCTimeSynchronization-Request, 'Time'= $D_1$ )  
     | MAKE (the local date and time =  $D_2$ )
6. WAIT(**Schedule\_Evaluation\_Fail\_Time**)
7. VERIFY  $S$ , Present\_Value =  $V_2$
8. REPEAT  $P$  = (writable property in List\_Of\_Property\_References)  
     VERIFY  $P$  =  $V_2$

Note to Tester: In the context of this test definition, writable means that the Schedule object is capable of modifying the property. It does not necessarily indicate that the property is modifiable via BACnet services.

### 7.3.2.23.Y.2 Internally Written Datatypes Test, NULL Values and Priority\_Arrays

Purpose: This test verifies that the Schedule object writes NULLs to priority arrays (via Present\_Value) within the same device.

Dependencies: TimeSynchronization Service Execution Tests, 9.30, UTCTimeSynchronization Service Execution Tests, 9.31

BACnet Reference Clauses: 12.24, 12.24.9

Test Concept: Two Date/Time values,  $D_1$  and  $D_2$ , are chosen by the TD based on the criteria in Table 7-Y2 such that  $D_1$  is sufficiently different from current time to cause a Schedule evaluation when the time is changed to  $D_1$ , and such that setting the time to  $D_2$  from  $D_1$  will cause a Schedule evaluation that will cause it to write value  $V_2$ . These values may be chosen based on the Schedule object's existing configuration, or the Schedule object, S, may be configured with such values, and either  $V_1$  or  $V_2$ , but not both, has datatype NULL. The values are written to a Present\_Value property with the priority designated by the Schedule object's Priority\_For\_Writing property, X. For devices of protocol revision 4 or higher, the Schedule\_Default shall be set to NULL and the schedule shall be configured such that at time  $D_1$  or  $D_2$ , but not both, the schedule shall take on the value of Schedule\_Default.

Configuration Requirements: The IUT shall be configured with a Schedule object, S, such that the time periods defined in Table 7-Y2 can be configured with uniquely scheduled values. The Schedule object shall be configured with a List\_Of\_Object\_Property\_References, including at least one reference within the device to a Present\_Value property, P, in an object containing a Priority\_Array property, PA. If the IUT cannot be configured to these requirements, then this test shall be omitted.

**Table 7-Y2. Criteria for Test Date and Times**

| Date and Time: | Value:                       |
|----------------|------------------------------|
| $D_1$          | $V_1$                        |
| $D_2$          | $V_2$ different from $V_1$ . |

Test Steps:

1. (TRANSMIT TimeSynchronization-Request, 'Time' =  $D_1$ )  
| (TRANSMIT UTCTimeSynchronization-Request, 'Time'=D1)  
| MAKE (the local date and time = D1)
2. WAIT(Schedule\_Evaluation\_Fail\_Time)
3. VERIFY S, Present\_Value =  $V_1$
4. VERIFY PA[X] =  $V_1$
5. (TRANSMIT TimeSynchronization-Request, 'Time' =  $D_2$ )  
| (TRANSMIT UTCTimeSynchronization-Request, 'Time'=D2)  
| MAKE (the local date and time = D2)
6. WAIT(Schedule\_Evaluation\_Fail\_Time)
7. VERIFY S, Present\_Value =  $V_2$
8. VERIFY PA[X] =  $V_2$

### 7.3.2.23.Y.3 Externally Written Datatypes Test, non-NULL values

Dependencies: TimeSynchronization Service Execution Tests, 9.26, UTCTimeSynchronization Service Execution Tests, 9.31

BACnet Reference Clauses: 12.24, 12.24.10

Purpose: This test verifies that the Schedule object writes to properties in other devices with all datatypes required and claimed for the external write operation. If the IUT supports Schedule objects that have differences in supported datatypes, then this test should be performed on at least one example of each type.

Test Concept: Two Date/Time values,  $D_1$  and  $D_2$ , are chosen by the TD based on the criteria in Table 7-Y3 such that  $D_1$  is sufficiently different from, and later than, the current time to cause a Schedule evaluation when the time is changed to  $D_1$ , and such that setting the time to  $D_2$  (later than  $D_1$ ) from  $D_1$  will cause a Schedule evaluation that will cause it to write value  $V_2$ . These values may be chosen based on the Schedule object's existing configuration, or the Schedule object may be configured with such values.

Configuration Requirements: The TD shall be configured to support the WriteProperty-Request service but not WritePropertyMultiple-Request in the Protocol\_Services\_Supported property of its Device object. The IUT shall be configured with a Schedule object, S, such that the time periods defined in Table 7-Y3 can be configured with uniquely scheduled values. The Schedule object shall be configured with a List\_Of\_Object\_Property\_Reference property, including at least one reference to a property in the TD. Other properties shall be consistent in both datatypes and values in a manner permitting this test to be executed.

**Table 7-Y3. Criteria for Test Date and Times**

| Date and Time: | Value:                                         |
|----------------|------------------------------------------------|
| D <sub>1</sub> | V <sub>1</sub>                                 |
| D <sub>2</sub> | V <sub>2</sub> different from V <sub>1</sub> . |

Test Steps:

1. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>1</sub>)  
 | (TRANSMIT UTCTimeSynchronization-Request, 'Time'=D1)  
 | MAKE (the local date and time = D1)
2. BEFORE **Schedule\_Evaluation\_Fail\_Time**  
 REPEAT X = (every reference to the TD in  
 List\_Of\_Object\_Property\_References) DO {  
 RECEIVE WriteProperty-Request,  
     'Object Identifier' = (the object identifier of X),  
     'Property Identifier' = (the property of X),  
     'Property Value' = V<sub>1</sub>  
     'Priority' = (the value of the Schedule object's  
                   Priority\_For\_Writing property)  
     TRANSMIT BACnet-SimpleACK-PDU  
 }  
 3. VERIFY S, Present\_Value = V<sub>1</sub>
4. (TRANSMIT TimeSynchronization-Request, 'Time' = D<sub>2</sub>)  
 | (TRANSMIT UTCTimeSynchronization-Request, 'Time'=D2)  
 | MAKE (the local date and time = D2)
5. BEFORE **Schedule\_Evaluation\_Fail\_Time**  
 REPEAT X = (every reference to the TD in  
 List\_Of\_Object\_Property\_References) DO {  
 RECEIVE WriteProperty-Request,  
     'Object Identifier' = (the object identifier of X),  
     'Property Identifier' = (the property of X),  
     'Property Value' = V<sub>2</sub>,  
     'Priority' = (the value of the Schedule object's  
                   Priority\_For\_Writing property)  
     TRANSMIT BACnet-SimpleACK-PDU  
 }  
 6. VERIFY S, Present\_Value = V<sub>2</sub>

Note to Tester: The Priority parameter for the WriteProperty-Request may be left out if the Schedule is configured with a value of 16 in its Priority\_For\_Writing property. The test shall pass regardless of the order in which the IUT generates the WriteProperty-Requests in steps 2 and 5.

#### 7.3.2.23.Y.4 Externally Written Datatypes Test, NULL values and Priority\_Arrays

Dependencies: TimeSynchronization Services Execution Tests, 9.26; UTCTimeSynchronization Service Execution Tests, 9.31

BACnet Reference Clauses: 12.24, 12.24.9

Purpose: This test verifies that the Schedule object writes NULLs to priority arrays (via Present\_Value) in other devices.

Test Concept: Two Date/Time values,  $D_1$  and  $D_2$ , are chosen by the TD based on the criteria in Table 7-Y4 such that  $D_1$  is sufficiently different from current time to cause a Schedule evaluation when the time is changed to  $D_1$ , and such that setting the time to  $D_2$  from  $D_1$  will cause a Schedule evaluation that will cause it to write value  $V_2$ . These values may be chosen based on the Schedule object's existing configuration, or the Schedule object may be configured with such values, and either  $V_1$  or  $V_2$ , but not both, has datatype NULL. The values are written to a Present\_Value property with the priority designated by the Schedule object's Priority\_For\_Writing property. For devices of protocol revision 4 or higher, the Schedule\_Default shall be set to NULL and the schedule shall be configured such that at time  $D_1$  or  $D_2$ , but not both, the schedule shall take on the value of Schedule\_Default.

Configuration Requirements: The TD shall be configured to support the WriteProperty-Request service but not WritePropertyMultiple-Request in the Protocol\_Services\_Supported property of its Device object. The IUT shall be configured with a Schedule object, S, such that the time periods defined in Table 7-Y4 can be configured with uniquely scheduled values. The Schedule object shall be configured with a Priority\_For\_Writing value other than 16, and with a List\_Of\_Object\_Property\_References, including at least one reference to a Present\_Value property in an object in the TD containing a Priority\_Array property.

**Table 7-Y4. Criteria for Test Date and Times**

| Date and Time: | Value:                       |
|----------------|------------------------------|
| $D_1$          | $V_1$                        |
| $D_2$          | $V_2$ different from $V_1$ . |

Test Steps:

1. (TRANSMIT TimeSynchronization-Request, 'Time' =  $D_1$ )  
     | (TRANSMIT UTCTimeSynchronization-Request, 'Time'= $D_1$ )  
     | MAKE (the local date and time =  $D_1$ )
2. BEFORE **Schedule\_Evaluation\_Fail\_Time**  
     REPEAT X = (every reference to the TD in List\_Of\_Object\_Property\_References) DO {  
         RECEIVE WriteProperty-Request,  
             'Object Identifier' = (the object identifier of X),  
             'Property Identifier' = (the property of X),  
             'Property Value' =  $V_1$ ,  
             'Priority' = (the value of the Schedule object's  
                             Priority\_For\_Writing property)  
         TRANSMIT BACnet-SimpleACK-PDU  
     }
3. VERIFY S, Present\_Value =  $V_1$
4. (TRANSMIT TimeSynchronization-Request, 'Time' =  $D_2$ )  
     | (TRANSMIT UTCTimeSynchronization-Request, 'Time'= $D_2$ )  
     | MAKE (the local date and time =  $D_2$ )
5. BEFORE **Schedule\_Evaluation\_Fail\_Time**  
     REPEAT X = (every reference to the TD in List\_Of\_Object\_Property\_References) DO {  
         RECEIVE WriteProperty-Request,  
             'Object Identifier' = (the object identifier of X),  
             'Property Identifier' = (the property of X),  
             'Property Value' =  $V_2$ ,  
             'Priority' = (the value of the Schedule object's  
                             Priority\_For\_Writing property)  
         TRANSMIT BACnet-SimpleACK-PDU  
     }
6. VERIFY S, Present\_Value =  $V_2$

Note to Tester: The Priority parameter for the WriteProperty-Request may be left out if the Schedule is configured with a value of 16 in its Priority\_For\_Writing property. The test shall pass regardless of the order in which the IUT generates the WriteProperty-Requests in steps 2 and 5.

[Add a new entry to **History of Revisions**, p. 489]

**(This History of Revisions is not part of this standard. It is merely informative and does not contain requirements necessary for conformance to the standard.)**

## HISTORY OF REVISIONS

| <i>Summary of Changes to the Standard</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |  |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|
| ...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |  |
| <b>Addendum g to ANSI/ASHRAE 135.1-2009</b><br>Approved by the ASHRAE Standards Committee January 29, 2011; by the ASHRAE Board of Directors February 2, 2011; and by the American National Standards Institute February 3, 2011.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |  |
| <ol style="list-style-type: none"> <li>1. Correct Test Step Indention.</li> <li>2. Remove Recipient Test.</li> <li>3. Correct Errors in Routing Tests.</li> <li>4. Change the Ignore Process ID Test.</li> <li>5. Add Max Info Frames Check.</li> <li>6. Add Test for Device Identifier Recipients.</li> <li>7. Add Test for Network Address Recipients.</li> <li>8. Add Tests for Disable Initiation.</li> <li>9. Change Tests for Out_Of_Service, Status_Flags, and Reliability.</li> <li>10. Add Tests for Non-router Network Layer Messages.</li> <li>11. Remove Time Delay in TO-FAULT Tests.</li> <li>12. Make Additions to the TCSL Language.</li> <li>13. Change Acknowledge Alarm Initiation Tests.</li> <li>14. Add New Tests for Reading and Presenting Properties.</li> <li>15. Add New Event Notification Tests.</li> <li>16. Update Trending Tests for Revision 3.</li> <li>17. Add New Tests for Revision 4 Schedules.</li> <li>18. Add New Test for Event Notification Network Priority.</li> <li>19. Add Device and Network Mapping Tests.</li> <li>20. Add Device Restart Notification Tests.</li> <li>21. Add Schedule Written Datatypes Tests.</li> </ol> |  |

**POLICY STATEMENT DEFINING ASHRAE'S CONCERN  
FOR THE ENVIRONMENTAL IMPACT OF ITS ACTIVITIES**

ASHRAE is concerned with the impact of its members' activities on both the indoor and outdoor environment. ASHRAE's members will strive to minimize any possible deleterious effect on the indoor and outdoor environment of the systems and components in their responsibility while maximizing the beneficial effects these systems provide, consistent with accepted standards and the practical state of the art.

ASHRAE's short-range goal is to ensure that the systems and components within its scope do not impact the indoor and outdoor environment to a greater extent than specified by the standards and guidelines as established by itself and other responsible bodies.

As an ongoing goal, ASHRAE will, through its Standards Committee and extensive technical committee structure, continue to generate up-to-date standards and guidelines where appropriate and adopt, recommend, and promote those new and revised standards developed by other responsible organizations.

Through its *Handbook*, appropriate chapters will contain up-to-date standards and design considerations as the material is systematically revised.

ASHRAE will take the lead with respect to dissemination of environmental information of its primary interest and will seek out and disseminate information from other responsible organizations that is pertinent, as guides to updating standards and guidelines.

The effects of the design and selection of equipment and systems will be considered within the scope of the system's intended use and expected misuse. The disposal of hazardous materials, if any, will also be considered.

ASHRAE's primary concern for environmental impact will be at the site where equipment within ASHRAE's scope operates. However, energy source selection and the possible environmental impact due to the energy source and energy transportation will be considered where possible. Recommendations concerning energy source selection should be made by its members.

