



BACnet TESTING LABORATORIES

# DEVICE IMPLEMENTATION GUIDELINES

V 0.17 - Revised April 4, 2007

# Table Of Contents

1	Introduction.....	1
2	General.....	2
2.1	Beware of assumptions.....	2
2.2	Be prepared to fall back when utilizing optional features.....	2
2.3	All BACnet devices shall execute ReadProperty-Request.....	2
2.4	A device shall execute all forms of a service.....	2
2.5	Max_APDU_Length_Accepted might be too large for an intervening connection.....	2
2.6	Polling intervals should be configurable.....	3
2.7	Do not flood the network on startup.....	3
2.8	For proprietary extensions, use higher numbers.....	3
2.9	Be prepared for large MAC addresses of at least 18 octets.....	3
2.10	Be prepared for future changes.....	3
2.11	Don't give up forever.....	3
2.12	DeviceCommunicationControl(disable-initiation) does not require Protocol Revision 4.....	3
2.13	Units properties may support values from higher Protocol Revision levels.....	4
2.14	The Accumulator and Pulse Converter objects require Protocol Revision 4.....	4
2.15	Be prepared to handle MAC addresses of any size less than or equal to 7 bytes.....	4
3	The Device object.....	5
3.1	All BACnet devices shall contain a Device object.....	5
3.2	Be prepared to encounter a BACnet device without a Device object.....	5
3.3	All Device objects shall contain a non-empty Object_List property.....	5
3.4	Be prepared to read the Object_List array element by element.....	5
3.5	The Device instance shall be configurable in the range of 0 to 4194302.....	5
3.6	Client devices should expect to see Device instances across the entire range of 0 to 4194302.....	5
3.7	Device instance number 4194303 is reserved for reading a Device object's Object_Identifier.....	5
3.8	The length of Device object Bit String properties might be different than expected.....	6
3.9	The value of Max_APDU_Length_Accepted might be different on different ports.....	6
3.10	The Protocol_Services_Supported property identifies only executable services.....	6
4	MS/TP.....	7
4.1	Support 38.4 and 76.8k bps.....	7
4.2	Support auto-bauding.....	7
4.3	Some device types may watch the LAN to determine factors such as baud rate.....	7
4.4	MS/TP MAC addresses should be configurable.....	7
5	Segmentation.....	8
5.1	Support all services with all segments full.....	8
5.2	The maximum number of segments that can be accepted is advertised in two places.....	8
5.3	Indicate the current max-segments-accepted limitation in the request APDU.....	8
5.4	If Max_Segments_Accepted is not present, the maximum number of segments might be two.....	8
5.5	If 'max-segmented-accepted' is '000', the maximum number of segments might be two.....	8
5.6	Devices supporting segmentation shall be able to use non-segmented methods to retrieve large amounts of data.....	8
5.7	Devices shall be prepared to interoperate with implementations that claim segmentation in one direction only.....	9
6	Device Binding.....	10
6.1	Implement I-Am and I-Have.....	10
6.2	Broadcast an I-Am on start-up.....	10
6.3	Use Who-Is and I-Am for device binding.....	10
6.4	Do not periodically broadcast I-Am and I-Have.....	10
6.5	Restrict the Who-Is device instance range.....	10
6.6	Space out Who-Is broadcasts.....	10
6.7	Reduce the rate of repeated Who-Is broadcasts.....	10
6.8	Client devices should support static binding.....	11
6.9	Support for proxy I-Am responses for MS/TP devices does not require Protocol Revision 4.....	11
7	Data Sharing.....	12

7.1	Support ReadPropertyMultiple .....	12
7.2	Clients should be able to fall back to smaller ReadPropertyMultiple requests.....	12
7.3	Clients should be able to fall back to ReadProperty requests.....	12
7.4	Do not poll as fast as the network allows, and polling intervals should be configurable .....	12
7.5	If multiple devices are being polled, alternate the polls .....	12
7.6	Do not subscribe for COV notifications with 'Lifetime' set to zero (indefinite lifetime) .....	12
7.7	If a COV subscription fails, poll for data and/or inform the operator.....	12
7.8	COV notifications from proprietary objects should include Present_Value and Status_Flags.....	13
7.9	List properties should be modifiable using WriteProperty .....	13
7.10	List properties should be modified using AddListElement and RemoveListElement .....	13
7.11	ReadPropertyMultiple execution shall support ALL, REQUIRED, and OPTIONAL.....	13
7.12	ReadRange execution shall support all list and array of lists properties .....	13
7.13	Inter-related properties should be read and written together for consistency in their values .....	13
7.14	Writable Character String properties should support strings of useful length.....	13
7.15	Client devices should be able to handle Signed and INTEGER values up to 32 bits .....	13
7.16	Client devices should be able to handle Enumerated values up to 16 bits .....	13
7.17	Time and Date wildcard values should only be used in service requests .....	14
7.18	Workstation devices should be able to read all basic datatypes .....	14
7.19	Support as-yet-undefined and proprietary object properties with primitive datatypes .....	14
7.20	Use detailed error reporting when all ReadPropertyMultiple accesses fail .....	14
7.21	ReadPropertyMultiple data values are returned in the order requested.....	14
7.22	Support NaN and +/-INF values for REAL and Double datatypes .....	14
7.23	The absence of a Priority_Array property does not indicate the Present_Value property is read only.....	14
7.24	Server devices are required to support all COV lifetime values up to 24 hours (86400 seconds). 14	
7.25	Client devices are required to support a COV lifetime value within 1 to 86400 seconds (up to 24 hours).....	15
8	Arrays.....	16
8.1	Support for array element 0 is required.....	16
8.2	Arrays shall be readable in their entirety (except...) .....	16
8.3	Resizable arrays should be resizable per Addendum 135a-8 to BACnet-2001 .....	16
8.4	Client devices should be prepared for array indexes of at least 16 bits .....	16
8.5	All 16 priority array levels shall be present.....	16
8.6	Priority array level 6 is not writable .....	16
8.7	If the device cannot afford a priority array, in out "Output" object, use a "Value" object.....	16
8.8	Re-evaluate priority array as soon as possible upon write .....	16
9	Alarms .....	17
9.1	Use the standard event algorithms if at all possible .....	17
9.2	Parse all event notifications.....	17
9.3	Do not implement the Event Enrollment object's Recipient property .....	17
9.4	Support the 'device' form of BACnetRecipient and the DM-DDB-A BIBB .....	17
9.5	Do not rely on the GetAlarmSummary service for acknowledging unseen alarms .....	17
9.6	Support the GetEventInformation service.....	17
9.7	GetEventInformation requires APDU sizes greater than 50 or segmentation .....	17
9.8	The Notification Class object's Recipient_List property should be persistent.....	18
9.9	B-OVS devices should be able to modify standard alarm properties of standard objects .....	18
9.10	Unless fixed by the application, standard alarm parameters should be writable .....	18
9.11	Recipient_List shall be writable .....	18
9.12	Support both ConfirmedEventNotification and UnconfirmedEventNotification .....	18
9.13	UnconfirmedEventNotification Execution requires APDU sizes greater than 50.....	18
9.14	Accept any Acknowledging Process Identifier in the AcknowledgeAlarm service request.....	18
9.15	Life Safety objects may advertise supported modes despite Protocol_Revision .....	19
9.16	LifeSafetyOperation execution Unsilence operations require Protocol_Revision 4 .....	19
9.17	The Event_Parameters property sets the event type, the Event_Type property indicates it .....	19
9.18	Use the event priorities found in Annex M .....	19
9.19	The Life Safety object behaviour in Addendum 135-2004a-1 requires Protocol_Revision 5 .....	19

9.20	To indicate “all day” in BACnetDestination use 00:00:00.00 to 23:59:59.99 .....	19
9.21	If the alarm support properties are present in an object, it shall support intrinsic alarming .....	19
10	Scheduling .....	20
10.1	Schedules should be modifiable .....	20
10.2	Schedules should be capable of numerous weekday entries and exception entries .....	20
10.3	Schedule objects should not be used to schedule complex or proprietary datatypes .....	20
10.4	Schedule objects should schedule certain basic datatypes .....	20
10.5	Workstations shall be able to configure schedules with NULL datatype.....	20
10.6	Schedule objects should at a minimum be able to schedule BACnetBinaryPV .....	20
10.7	Workstations should, at a minimum, support Schedules that schedule BACnetBinaryPV .....	20
10.8	Schedule objects scheduling Unsigned and INTEGER values should support 32 bit values .....	20
10.9	Schedule objects scheduling Enumerated values should support 16 bit values .....	21
10.10	Support for Protocol Revision 4 Schedule objects requires at least Protocol Revision 4 .....	21
10.11	NULL is a valid value for the Schedule_Default property.....	21
11	Trending.....	22
11.1	Implement the most recent Trend Log and ReadRange .....	22
11.2	Support for Protocol Revision 4 Trend Log objects requires at least Protocol Revision 3 .....	22
11.3	ReadRange clients should keep the 'Count' parameter in the range of Signed16 .....	22
12	Routing.....	23
12.1	Drop messages with a Hop Count of zero.....	23
12.2	Drop messages if Hop Count is decremented past zero .....	23
12.3	Initialize -Routing-Table might not operate as expected.....	23
12.4	Add known networks to empty Router-Busy-To-Network messages.....	23
12.5	Routers should redirect unicast messages .....	23
12.6	Routers should not redirect broadcast messages.....	23
12.7	Routers should forward unknown network message types .....	23
12.8	Routers should support the maximum frame size for each supported medium.....	23
13	Backup and Restore.....	24
13.1	Size the configuration File objects before restoring.....	24
13.2	A device should be able to handle an interrupted Restore operation .....	24
13.3	Empty files shall be backed up and restored if listed in the Configuration_Files property.....	24
14	Annex J BACnet/IP .....	25
14.1	A non-BBMD BACnet/IP device shall be able to register as a Foreign Device .....	25
14.2	A Broadcast Distribution Table should be able to contain more than four entries .....	25
14.3	Two-hop BBMD entries are required – one-hop support is not required.....	25
14.4	BBMDs shall support Foreign Device registrations .....	25
14.5	A Foreign Device Table should be able to contain at least two entries .....	25
14.6	A BBMD shall be capable of forwarding Forwarded-NPDU broadcasts to its local subnet .....	25
14.7	BACnet/IP devices should support classless addressing.....	25
14.8	Routers supporting BACnet/IP shall support fragmentation.....	25
15	Miscellaneous.....	26
15.1	Context tags greater than 14 shall be properly handled by a parser.....	26
15.2	Implement the Abort reasons of Clause 5.4.5.3 (BACnet-2004) .....	26
15.3	Use error codes from higher Protocol_Revisions levels may be supported.....	26
15.4	BACnet-EIB/KNX mapping may be done independent of Protocol_Revision.....	26
16	Known Issues .....	27
16.1	Rapid transitions in the OUT_OF_RANGE and FLOATING_LIMIT algorithms .....	27
16.2	Interoperation of the Limit_Enable, Event_Enable and Event_State properties.....	27
17	Resources .....	28
17.1	BACnet standards:.....	28
17.2	The BACnet website.....	28
17.3	The BACnet Testing Labs website.....	28
17.4	Public e-mail lists:.....	28

## **1 Introduction**

The BACnet standard offers many options in the implementation of BACnet devices. Members of the BACnet Testing Labs (BTL) have agreed on specific options of capabilities that should be implemented in BACnet devices to maximize their interoperability with devices that are tested and listed by the BTL.

This document presents those options and capabilities as recommended guidelines for all BACnet implementers. It also presents guidelines to avoid mistakes made in the past by new BACnet implementers.

References to the BACnet standard, unless otherwise specified, are to *ANSI/ASHRAE Standard 135-2001, BACnet – A Data Communication Protocol for Building Automation and Control Networks*, “BACnet-2001.”

## 2 General

The following guidelines apply to all devices.

### 2.1 Beware of assumptions

Don't make assumptions about the behaviour of a communicating peer beyond what the BACnet standard specifies. The peer device might not implement some functionality, property or service that is designated by the standard as being optional. (Common examples of such are the basis of specific recommendations in this guide.)

In particular, be prepared to communicate with devices implemented to an earlier protocol revision. (Additions are listed in the "History of Revisions" at the end of the BACnet standard. The referenced Addenda detailing the changes are available online from ASHRAE See **17.1**.)

### 2.2 Be prepared to fall back when utilizing optional features

Do not rely on other devices supporting an optional capability of BACnet; it might not be there. Whenever possible provide a fallback position by using features that are required. For example, if one wants reasonably timely data updates from a peer device and seeks to acquire them by subscribing for change of value reports (COV), fall back to polling for the data if the peer device does not support COV subscription, or if it rejects the subscription request for some other reason.

### 2.3 All BACnet devices shall execute ReadProperty-Request

All BACnet devices shall be able to execute the ReadProperty service request, even BACnet client devices. The BACnet standard requires this for all devices with an application layer. (This requirement may soon include BACnet routers; see note in section 3.)

### 2.4 A device shall execute all forms of a service

When a device is able to execute a service request, it shall be able to execute all forms of the service. This allows a client device to select the form of the service request it chooses and expect successful interoperation. "All forms of the service" include all forms of the service parameters (see 7.11, for example), and its application to appropriate objects and services (see 7.12).

The only exceptions to this rule are the password parameters of the DeviceCommunicationControl and ReinitializeDevice services, where the BACnet standard allows restriction, by requiring the 'password' parameter to be present, for example.

### 2.5 Max\_APDU\_Length\_Accepted might be too large for an intervening connection

The maximum size APDU that a device can generate should be either no larger than the PTP datalink's maximum APDU size of 480 octets, or the device's Max\_APDU\_Length\_Accepted property should be configurable and this value used to restrict the size of the APDU that will be sent by the device. This will prevent messages from being dropped when they encounter intervening PTP links.<sup>1</sup>

An alternative recommendation is for a device, upon initiating communications with a peer device on a remote network, to determine how large an APDU can reach that device. This can be accomplished by sending service requests of different sizes to the peer device; if a response (of any kind) comes from the peer, the request reached it. If a Reject-Message-To-Network with a reject reason of 4 ("The message is too long to be routed to this DNET") is returned, the APDU is too large to reach that device. This process

---

<sup>1</sup> This guideline assumes that Ethernet LANs will not be interconnected through an MS/TP LAN, with its maximum APDU size of 480 octets; and that Ethernet, ARCNET and MS/TP LANs will not be interconnected through a LonTalk LAN, with its maximum size of 206 octets. These cases present the same difficulty as joining Ethernet LANs with a PTP link, however.

works even for cases where a device reports an incorrect value, such as an Ethernet to MS/TP router reporting a Max\_APDU\_Length\_Accepted value of 1476 on its MS/TP port (see 3.9).

## **2.6 Polling intervals should be configurable**

For devices that are capable of polling, the poll interval should be configurable and the default should be reasonable, not “as fast as possible.” This reduces the risk of overloading slower LANs and swamping routers; in addition, experience shows that not all Ethernet devices might be able to handle BACnet requests sent “as fast as possible,” perhaps due to internal processing delays.

## **2.7 Do not flood the network on startup**

Upon device startup, do not flood the network with messages. This avoids overloading networks and routers, and possibly recipient devices. In particular, alarm- and COV-notifying devices (i.e. alarm of COV servers) should be designed to suppress creation of multiple notifications immediately after power-up, if possible.

## **2.8 For proprietary extensions, use higher numbers**

When creating vendor-proprietary extensions (per Clause 23 of the standard), whether it is for proprietary object types, proprietary properties or extending enumerations, use higher-numbered values rather than assign sequentially from the lowest value allowed for vendor-proprietary extensions. This applies particularly to the BACnetEngineeringUnits enumeration, for which the reserved range may need to be extended.

## **2.9 Be prepared for large MAC addresses of at least 18 octets**

IPv6 uses 16-octet addressing. Coupling this with the two-byte UDP port, even a small MS/TP device might someday receive messages with a SADR field of 18 octets, considerably longer than the largest SADR field of 7 octets mentioned in clause 6.2.2.2 of the BACnet standard.

If provision is not made for the larger MAC addresses, existing devices will not be able to issue requests to, nor reply to requests from, devices with such larger MAC addresses. (Note: the BACnet committee is discussing this issue and is looking at a way around it.)

## **2.10 Be prepared for future changes**

In the future the available CHOICES in any particular ASN.1 production (in Clause 21) might be expanded beyond what is currently defined, as might enumerations (such as BACnetPropertyIdentifier). In addition, reserved fields and values might later be defined and used.

Devices (and parsers) should be implemented to handle these situations when encountered, without crashing the device.

## **2.11 Don't give up forever**

If a confirmed service request fails (all APDU timeouts and retries pass without a response), don't give up forever on communicating with the remote device. If dynamic binding (Who-Is or Who-Has) is used to locate the remote device or object, fall back to occasionally initiating the Who-Is or Who-Has to locate the remote device or object.

Similarly, if a Who-Is or Who-Has intended to locate a remote device or object is sent without receiving the corresponding I-Am or I-Have, consider re-sending the Who-Is or Who-Has occasionally until the I-Am or I-Have is seen.

## **2.12 DeviceCommunicationControl(disable-initiation) does not require Protocol Revision 4**

The BTL allows a device to support the 'disable-initiation' option for the 'enable-disable' parameter (introduced by Protocol\_Revision 4) regardless of the Protocol\_Revision claimed by the device.

**2.13 Units properties may support values from higher Protocol Revision levels**

The BTL allows a device to support values for Units property enumerations defined by Protocol\_Revision levels higher than that claimed for the device.

**2.14 The Accumulator and Pulse Converter objects require Protocol Revision 4**

The BTL requires that devices that support the Accumulator and Pulse Converter objects to claim Protocol\_Revision 4..

**2.15 Be prepared to handle MAC addresses of any size less than or equal to 7 bytes**

MAC addresses are not limited to 1, 2, 6 or 7 bytes as might be thought from Table 6-2 of ASHRAE 135-2004. Any size MAC address should be allowed which is less than or equal to 7 bytes.

As an example, gateways to proprietary systems are allowed to provide MAC addresses that are not included in the standard table.

### **3 The Device object**

The following guidelines cover devices that contain a Device object, or that access another device's Device object.

#### **3.1 All BACnet devices shall contain a Device object**

All BACnet devices with an application layer shall contain a Device object, even BACnet clients. This is required by the BACnet standard (clause 12.10 of the BACnet standard, "There shall be exactly one Device object in each BACnet Device.").

Only BACnet routers are exempt, and only if they do not contain an application layer. However, there is currently a proposal before the BACnet committee to require routers to have application layers.

#### **3.2 Be prepared to encounter a BACnet device without a Device object**

BACnet devices should be prepared to encounter a BACnet device without a Device object, despite item.

3.1. Such devices, typically client (workstation-like) devices, have been implemented and installed.

The problem most frequently observed is a device that stalls or locks up when requests to read another device's Device object's properties fail, or are ignored.

#### **3.3 All Device objects shall contain a non-empty Object\_List property**

All Device objects shall contain an Object\_List property, and this property shall include the Device object. This is required by the BACnet standard.

#### **3.4 Be prepared to read the Object\_List array element by element**

Some small devices that do not support segmentation have Object\_List properties that are too large to transmit unsegmented. If a device needs to read another's Object\_List property, be prepared to read it array element by array element.

#### **3.5 The Device instance shall be configurable in the range of 0 to 4194302**

The BTL requires the Device instance to be configurable across the entire valid range of 0 to 4194302. Device instance numbers are often used to encode information such as location (building-floor-room). Restricted configurable instance ranges hinder this practice.

Fixed, unchangeable, device instance numbers are a problem waiting to occur. Even if there would be, say, only one such device in a building, a fixed device instance can cause problems when several buildings are later joined on a campus, metropolitan or wide-area network.

The means by which the device instance can be changed is left to the implementer, whether through DIP switches, a proprietary tool or other means.

#### **3.6 Client devices should expect to see Device instances across the entire range of 0 to 4194302**

Client devices should expect to see Device instances across the entire range of 0 to 4194302. If the client device stores Device instances, it shall provide storage capability for the full instance range (22 bits).

#### **3.7 Device instance number 4194303 is reserved for reading a Device object's Object\_Identifier**

Device instance number 4194303 is a "wildcard" value for reading a Device object's Object\_Identifier property (to determine its Device instance). If a read request is received for the Object\_Identifier property of Device 4194303, the response shall convey the responding device's correct Device object instance.

The ability to respond to instance 4194303 as a "wildcard" value was added in Addendum 135a-5 to BACnet-2001; it might not be implemented in earlier devices.

### **3.8 The length of Device object Bit String properties might be different than expected**

The length of the Device object's Bit String properties, in particular the BACnet\_Services\_Supported and BACnet\_Object\_Types\_Supported properties, will vary depending upon the protocol revision to which the device was implemented. Client devices should be prepared to accept Bit String values with lengths longer or shorter than those defined for the Protocol\_Revision value to which the client device was implemented.

### **3.9 The value of Max\_APDU\_Length\_Accepted might be different on different ports**

The value of Max\_APDU\_Length\_Accepted should not exceed the maximum value defined for the datalink layer from which it is read; this could lead to attempts to send frames that are too large to be routed the device (see 2.5). For example, in an Ethernet to MS/TP router, even though a Max\_APDU\_Length\_Accepted value of 1476 is read from its Ethernet port the value 480 should be read from its MS/TP port.

### **3.10 The Protocol\_Services\_Supported property identifies only executable services**

Only bits representing services that are executable by the device shall be set to '1' in the Protocol\_Services\_Supported property . Bits representing services that are initiated by the device but not executed shall be set to '0'.

This property is a means for the device to advertise to other devices which services may be sent to the device with an expectation that they will be accepted and executed.

## **4 MS/TP**

The following guidelines cover devices that participate in MS/TP LAN communications. (See clause 9 of the BACnet standard.)

### **4.1 Support 38.4 and 76.8k bps**

Support for 9.6k bps is required by the BACnet standard. If at all possible, support 38.4k and 76.8k bps for better throughput for all devices sharing the MS/TP LAN.

### **4.2 Support auto-bauding**

All non-routing MS/TP devices should implement autobauding, to facilitate changing an MS/TP LAN's baud rate.

### **4.3 Some device types may watch the LAN to determine factors such as baud rate**

Devices that route to MS/TP, or that will spontaneously initiate service requests other than I-Am (excluding devices that might initiate COV notifications but which have no subscribers), are permitted to delay start-up for a time in order to observe traffic on the LAN to determine factors such as baud rate. They are prohibited (by the BTL) from observing indefinitely; this guarantees that the MS/TP LAN will start up.

Devices that do not route to MS/TP, and that will not spontaneously initiate service requests other than I-Am when they do start up, are permitted to observe traffic until they have sufficient information to complete their initialization and start.

### **4.4 MS/TP MAC addresses should be configurable**

MAC addresses of MS/TP devices should be configurable by some means, whether by DIP switches or a configuration tool. Fixed, unchangeable MS/TP MAC addresses can add difficulty in replacing a device.

## 5 Segmentation

The following guidelines cover devices that support segmentation (see clause 5.3, BACnet-2001).

It should be noted that there exists an implicit assumption: If a device supports segmentation for both transmit and receive, the maximum number of segments it can send and receive are identical.

### 5.1 Support all services with all segments full

If the device supports segmentation and advertises a maximum number of supported segments, the BTL requires that it shall be able to execute any service request (that could be large enough to require segmentation) with all segments filled.

This rule applies in general to service initiation as well, though there may be exceptions such as restricting the number of property requests (and the datatypes requested) contained in a ReadPropertyMultiple request so that the response, including worst-case error returns, will not overrun the number of supported segments.

### 5.2 The maximum number of segments that can be accepted is advertised in two places

The maximum number of segments that can be accepted by a device is "advertised" in two places: the Max\_Segments\_Accepted property of the Device object (clause 12.10.19 of BACnet-2001) and in the 'max-segments-accepted' parameter of the BACnet-Confirmed-Request-APDU (clause 20.1.2.4 of BACnet-2001).

The value of the Max\_Segments\_Accepted property, if present, of a peer device should be considered when initiating a service request to that peer. Devices responding to a service request shall use the 'max-segments-accepted' parameter of a confirmed service request to determine if the response is transmittable (See BACnet-2001, clause 5.4.5.3, transition CannotSend SegmentedComplex-ACK.)

### 5.3 Indicate the current max-segments-accepted limitation in the request APDU

If the number of the segments that can be received in the response for a particular request is less than that displayed in the Max\_Segments\_Accepted property of the Device object for some reason, such as other concurrent requests using up available buffers, a smaller value shall be indicated in the 'max-segments-accepted' parameter of the BACnet-Confirmed-Request-APDU.

The value of zero for this parameter, indicating "unspecified number of segments accepted," is present for backwards compatibility with older devices, prior to Addendum *e* of ANSI/ASHRAE 135-1995, and should not be used (see 5.5).

### 5.4 If Max\_Segments\_Accepted is not present, the maximum number of segments might be two

If a peer device does not support the Max\_Segments\_Accepted property of its Device object but indicates that it supports segmentation, the maximum number of segments of a message that the peer device will accept might be as low as two.

### 5.5 If 'max-segmented-accepted' is '000', the maximum number of segments might be two

If the 'max-segments-accepted' parameter of a confirmed service request is '000', indicating "unspecified number of segments accepted") and the 'segmented-response-accepted' parameter is TRUE, the maximum number of segments of a message that the peer device will accept might be as low as two.

### 5.6 Devices supporting segmentation shall be able to use non-segmented methods to retrieve large amounts of data

Client devices that support segmentation and retrieve large amounts of data shall be prepared to use non-segmented services when server devices indicate they do not support segmentation.

For example, the Object\_List for a device may not fit into a single segment. In this case, the individual array items shall be read. One method is to read the size of the Object\_List property and then read each

item individually using the ReadProperty service or a group of items could be read using ReadPropertyMultiple service provided that each request and answer response can be placed in a single segment.

**5.7 Devices shall be prepared to interoperate with implementations that claim segmentation in one direction only**

Devices shall be able to interoperate with implementations that support segmentation on transmit only.  
Devices shall be able to interoperate with implementations that support segmentation on receive only.

## **6 Device Binding**

The following guidelines cover the implementation of the Who-Is, I-Am, Who-Has and I-Have service requests, which are used to locate ("bind to") specific devices.

### **6.1 Implement I-Am and I-Have**

The Who-Is service is a convenient mechanism used by most devices to bind to peer devices. A few devices use the Who-Has request instead. For this reason all devices should execute the Who-Is and Who-Has requests and initiate, in response, the IAm or IHave request. MS/TP slave devices are exempt because they cannot initiate requests, however there are devices that cannot communicate with the slave devices. See 6.8.

It is preferable, but not required, that I-Am and I-Have requests generated as a result of a Who-Is or Who-Has request be broadcast only on the network where the Who-Is originated. This reduces the amount of unnecessary network traffic elsewhere in the system.

### **6.2 Broadcast an I-Am on start-up**

It is a common practice to broadcast an I-Am globally (to the entire internetwork) on start-up. Though this practice can slow startup of large systems after a power failure, it allows client devices that have already started to bind to this device sooner, and may restore normal operation more quickly when a device has been moved or replaced, perhaps with a device with a different MAC address.

### **6.3 Use Who-Is and I-Am for device binding**

Use Who-Is and I-Am for locating devices on the internetwork ("device binding").

When a device with a worldwide unique MAC address (such as an Ethernet MAC address) is replaced, static binding (entering a device's network number and MAC address) can create a lot of error-prone work, because every single static reference to the device that was replaced, throughout the entire system, needs to be reconfigured.

### **6.4 Do not periodically broadcast I-Am and I-Have**

Although the BACnet standard allows I-Am and I-Have requests to be initiated at any time by a BACnet device, these requests should not be periodically broadcast. This creates unnecessary traffic that can impede PTP connections and slower LANs. In addition, some devices undertake additional actions (in the form of confirmed service requests to the device that sent the IAm) in response to an I-Am because the device may have been replaced, the device may have been replaced with a different type of device, or its location may have changed.

### **6.5 Restrict the Who-Is device instance range**

Do not issue globally broadcast Who-Is messages (i.e., without 'Device Instance Range Low/High Limit' parameters), except as the first step of mapping a system such as by a workstation. In large systems these cause massive numbers of IAm broadcasts, causing IAm responses and other messages to be lost. It is better to issue a single Who-Is request for each specific peer device that is to be located.

### **6.6 Space out Who-Is broadcasts**

Do not issue a number of Who-Is requests, one immediately after the next. The resulting flood of Who-Is and IAm messages can overwhelm routers and slower connections and LANs. It is better to space the requests out, say, one per second or one per 100 mS, rather than issue them all at once.

### **6.7 Reduce the rate of repeated Who-Is broadcasts**

If it is not absolutely essential that a device's function requires the earliest possible establishment of communications with a peer (for example, if it is not involved in life- or equipment-safety operations), the

rate at which an unanswered Who-Is for that peer is repeated should be reduced. If part of a system becomes unreachable, say by a router going offline, the resulting flood of rapidly-repeated Who-Is requests for devices on the other side of that router can impede PTP communication and slower LANs.

Where the peer-to-peer communications are less than critical, it is better to initially repeat the unanswered Who-Is every 10 seconds, or the time specified by the Device object's APDU\_Timeout property, increasing over time to some maximum interval, say, 5 minutes or the time calculated by multiplying the value of the Device object's APDU\_Timeout property by (1 plus the value of APDU\_Retries).

#### **6.8 Client devices should support static binding**

Client devices should be able to be configured with the network number and MAC address of a device with which they will be communicating, as an alternative to the device instance. This will allow them to communicate with MS/TP slaves or other devices that do not support Who-Is and Who-Has execution.

#### **6.9 Support for proxy I-Am responses for MS/TP devices does not require Protocol Revision 4**

The BTL allows a device to support proxy I-Am responses for MS/TP devices regardless of the Protocol\_Revision claimed by the device. Support for the feature is indicated by the presence of the Slave\_Proxy\_Enable property.

## **7 Data Sharing**

The following guidelines apply to all devices with an application layer, i.e., they acquire data from or provide data to other devices in BACnet APDUs.

### **7.1 Support ReadPropertyMultiple**

Support ReadPropertyMultiple execution (and, if a client device, ReadPropertyMultiple initiation) even in small devices. The improvement in network bandwidth when data is being polled is significant.

### **7.2 Clients should be able to fall back to smaller ReadPropertyMultiple requests**

Devices that initiate ReadPropertyMultiple requests should be able to retry with, or be reconfigured to use, ReadPropertyMultiple requests for fewer property values if there is some indication that the data cannot be conveyed to the client. This will help ensure interoperation between the two devices.

The indicators received from the peer device typically are an Abort (BUFFER\_OVERFLOW) APDU or an Abort (SEGMENTATION\_NOT\_SUPPORTED) APDU (see clause 5.4.5.3 in the BACnet standard, transition 'CannotSendSegmented-ComplexACK'). Another indicator could be that a reply is never received, if there is an intervening connection (LonTalk or MS/TP LAN, or PTP connection) unable to convey the reply. (See item 2.5.)

### **7.3 Clients should be able to fall back to ReadProperty requests**

Devices that initiate ReadPropertyMultiple requests should be able to revert to ReadProperty requests if the server device does not indicate (on the server's Device object's Protocol\_Services\_Supported property) support ReadPropertyMultiple execution, or (as in 7.2) if there is some indication that the data cannot be conveyed to the client, or if an Error PDU conveying the code SERVICE\_NOT\_SUPPORTED or a Reject PDU conveying the code UNRECOGNIZED\_SERVICE is received. This will help ensure interoperation between the two devices.

### **7.4 Do not poll as fast as the network allows, and polling intervals should be configurable**

A device that polls other devices for data should not poll as fast as the network will allow; this can impede PTP communication and slower LANs. The interval between polls should also be configurable.

### **7.5 If multiple devices are being polled, alternate the polls**

A device that is polling multiple peer devices for data at the same rates should, if possible, rotate the requests among the peer devices "round-robin" rather than send a number of ReadProperty (or ReadPropertyMultiple) requests to one device, then a number of requests to the next device, and so on.

### **7.6 Do not subscribe for COV notifications with 'Lifetime' set to zero (indefinite lifetime)**

Devices should not issue SubscribeCOV or SubscribeCOVProperty requests with the 'Lifetime' parameter set to zero. (The value zero is prohibited for SubscribeCOVProperty requests.) The COV subscription list is not guaranteed to remain through a device reset or power loss, or the device holding the subscription could fail and be replaced. Worse, if the subscription list is permanently held and the subscribing device is removed, the storage for the removed device's subscription remains allocated forever.

SubscribeCOV requests should be issued with a non-zero 'Lifetime' parameter for some period (such as the amount of time it is acceptable for the subscribing device to operate with out-of-date data), and the subscription should be refreshed periodically.

### **7.7 If a COV subscription fails, poll for data and/or inform the operator**

COV subscriptions might not always succeed, even with devices that execute SubscribeCOV or SubscribeCOVProperty requests. Devices are permitted to support a limited number of subscriptions, for example (clauses K.1.12 and K.1.13 in the BACnet standard require only five concurrent subscriptions). If possible, some action to correct the situation should be initiated.

One possible course of action is to fall back to polling for the data. Another (not necessarily exclusive) is to notify the operator of the failure to subscribe so that a change can be made in the system configuration.

#### **7.8 COV notifications from proprietary objects should include Present\_Value and Status\_Flags**

COV notifications from proprietary objects should include the values of the Present\_Value and Status\_Flags properties in addition to any other property values included in the notifications. This is for consistency with most other COV notifications.

#### **7.9 List properties should be modifiable using WriteProperty**

List properties that may be modified using BACnet services should also be writable using the WriteProperty service request (and WritePropertyMultiple, if supported). Modifications of such properties should not be restricted (by the server device) to the AddListElement and RemoveListElement service requests, for better interoperability.

#### **7.10 List properties should be modified using AddListElement and RemoveListElement**

In order to reduce the potential for conflict when two or more clients are modifying a list property concurrently, the list should be modified using AddListElement and RemoveListElement service requests. (However, if an element of a list is removed by one RemoveListElement request and a subsequent RemoveListElement request specifies that element, the latter request will fail in its entirety.)

#### **7.11 ReadPropertyMultiple execution shall support ALL, REQUIRED, and OPTIONAL**

The BACnet standard requires that ReadPropertyMultiple service execution supports the special property identifiers ALL, REQUIRED, and OPTIONAL. (See clause 15.7.2 in the BACnet standard.) Some workstation devices rely on these special properties for accessing other devices' objects.

#### **7.12 ReadRange execution shall support all list and array of lists properties**

The BACnet standard requires that ReadRange service execution support any property that is a list or an array of lists. (See clause 15.8 in the BACnet standard; note, though, there is a proposal to allow ReadRange to read arrays.) ReadRange service execution is not allowed to be restricted to the Trend Log object's Log\_Buffer property.

#### **7.13 Inter-related properties should be read and written together for consistency in their values**

Inter-related properties, such as High\_Limit and Low\_Limit should be read and written together using ReadPropertyMultiple and WritePropertyMultiple. Although this does not guarantee consistency between the values, because the operations are not required to be atomic, it may improve the likelihood of consistency over access using ReadProperty and WriteProperty services.

#### **7.14 Writable Character String properties should support strings of useful length**

Writable Character String properties should support strings of useful length. Although "useful" is a vague term, an Object\_Name property that can contain at most a four octet Character String is not likely to be useful to a user, whereas an Object\_Name of length 512 is likely to be far more than adequate.

#### **7.15 Client devices should be able to handle Signed and INTEGER values up to 32 bits**

Client devices should be able to handle Signed and INTEGER values at least up to 32 bits in length, unless the particular value is specifically restricted by the BACnet standard (such as Unsigned8). This is the minimum recommended by the BTL for interoperability with other devices.

#### **7.16 Client devices should be able to handle Enumerated values up to 16 bits**

Client devices should be able to handle Enumerated values at least up to 16 bits in length, unless the particular value is specifically restricted by the BACnet standard. This is the minimum recommended by the BTL for interoperability with other devices.

### **7.17 Time and Date wildcard values should only be used in service requests**

Time and Date wildcard values (X'FF') should only be used in service requests, or when all elements of the Time or Date value are unknown. Time and Date values conveying actual time information should not use wildcard values. There is a proposal currently before the BACnet committee to add this requirement to the BACnet standard.

When specifying date or time ranges, the date or time should have all or no wildcards, except when used in a consistent fashion, for example, "from 1<sup>st</sup> week in October to 3<sup>rd</sup> week in October." Inconsistent usages such as "from Thursday until December," should be avoided.

### **7.18 Workstation devices should be able to read all basic datatypes**

Operator workstations should be able to read all basic ("application," or "primitive") datatypes (as enumerated in clause 20.2.1.4 of the BACnet standard), and be able to read basic datatypes from proprietary properties. This is recommended for improved interoperability with other devices.

### **7.19 Support as-yet-undefined and proprietary object properties with primitive datatypes**

Devices that can read and write properties in other devices should be configurable to read and write properties in as-yet-undefined standard objects and proprietary objects, as well as as-yet-undefined standard properties and proprietary properties in any objects, at least where those properties have primitive datatypes.

This improves the ability of such devices to access new objects as they are defined, as well as access to other manufacturers' proprietary objects.

### **7.20 Use detailed error reporting when all ReadPropertyMultiple accesses fail**

The BTL recommends that if all of a ReadPropertyMultiple request's accesses fail, the response should be a 'Result(+)' primitive returning an error class and code for each access rather than a 'Result(-)' primitive conveying a single error code and class, particularly where the error classes and codes differ for different accesses.

### **7.21 ReadPropertyMultiple data values are returned in the order requested**

The data values conveyed in a ReadPropertyMultiple-ACK shall be returned in the order in which they are requested in the ReadPropertyMultiple-Request. This is required by Clause 15.7.2 of the BACnet standard.

### **7.22 Support NaN and +/-INF values for REAL and Double datatypes**

IEEE REAL and Double numbers have encodings for NaN (Not a Number) as well as plus and minus infinity. If such a value is written to the device, an error reply may be appropriate. Be prepared to process such a value "appropriately" if it is returned in a response to a request such as ReadProperty.

### **7.23 The absence of a Priority\_Array property does not indicate the Present\_Value property is read only**

The Present\_Value property may be writable even if the value object does not support the Priority\_Array property. If the property is writable the device shall ignore the priority sent on a WriteProperty command.

### **7.24 Server devices are required to support all COV lifetime values up to 24 hours (86400 seconds)**

The BTL requires that devices that execute SubscribeCOV or SubscribeCOVProperty (COV servers) accept all lifetime values in the range 1 through 86400 seconds (24 hours). COV server devices may support values outside the range as well.

The BTL requires support for this range of lifetime values in order to ensure that COV clients are able to select lifetime values that will not be rejected by COV server devices.

**7.25 Client devices are required to support a COV lifetime value within 1 to 86400 seconds (up to 24 hours)**

The BTL requires that COV clients be able to subscribe with a lifetime in the range 1 through 86400. Client devices need not support all values in the range, nor are they restricted to supporting values only in that range.

The BTL requires that COV clients support a value within this range to ensure that COV clients can subscribe with a lifetime value that will be accepted by all COV servers.

## 8 Arrays

The following guidelines cover the implementation of arrays, including the priority arrays found in "Output" and "Value" object types (such as Analog Output or Multi-state Value), and described in clause 19.2 of the BACnet standard.

### 8.1 Support for array element 0 is required

Support for array element 0, which reports the size of the array, is required by the BACnet standard for all arrays, including the Device object's Object\_List property. (See clause 12 of the BACnet standard.) In the latter case, devices sometime read element 0 to determine how they will proceed to read the Object\_List property (see 3.4).

### 8.2 Arrays shall be readable in their entirety (except...)

The BTL requires that array properties shall be readable in their entirety, such as with the ReadProperty service, except when the response is too large to fit in a single APDU and either the client or the server device does not support segmentation. The BTL does not allow devices to arbitrarily restrict read access methods, in order to improve interoperability.

### 8.3 Resizable arrays should be resizable per Addendum 135a-8 to BACnet-2001

Resizable arrays should be resizable by the methods specified in Addendum 135a-8 to BACnet-2001 for improved interoperability (in BACnet-2004 the method is described in the initial paragraphs of Clause 12).

### 8.4 Client devices should be prepared for array indexes of at least 16 bits

Client devices should be prepared to handle array indexes of at least 16 bits; this includes the size of array element 0, which reports the size of the array. This is recommended for interoperability with other devices.

### 8.5 All 16 priority array levels shall be present

When implementing priority arrays, all 16 levels shall be present. The BACnet standard does not allow an implementation to have fewer levels.

### 8.6 Priority array level 6 is not writable

The BACnet standard reserves Level 6 of the priority array for the Minimum On/Off Time algorithm and prohibits its use for any other purpose in any object. (Clause 19.2.3 of the BACnet standard: "Command priority 6 is reserved for use by this algorithm and may not be used for other purposes in any object.") As such, writes using BACnet services should result in 'Result(-)' responses.

### 8.7 If the device cannot afford a priority array, in out "Output" object, use a "Value" object

All output objects shall have priority arrays, but value objects are not required to have them. If the device needs to have an object that represents a physical output but cannot afford a priority array, use the corresponding "Value" object type instead (for example, a Binary Value object instead of a Binary Output).

### 8.8 Re-evaluate priority array as soon as possible upon write

Priority arrays should be evaluated to update the Present\_Value property as soon as possible when written, preferably before the write is ACKed, but in particular before a read request for the Present\_Value or Priority\_Array properties is serviced. Extensively delayed updates can confuse operators and can cause problems in testing.

## **9 Alarms**

The following guidelines apply to devices that initiate or receive alarms, i.e. ConfirmedEventNotification or UnconfirmedEventNotification messages.

### **9.1 Use the standard event algorithms if at all possible**

When implementing event initiation, use the standard event algorithms if at all possible. There are at present devices that are not able to parse proprietary ‘Event Values’ (the event notification parameters). (See 9.2.)

### **9.2 Parse all event notifications**

Devices that receive and process event notifications should be able to receive and process proprietary event types, where the construction of the ‘Event Values’ (the notification parameters) is not known. The parser should be able to parse through this field, even though the values will be discarded, in order that the notification can be received and processed. A device should not drop (ignore) event notifications simply because it does not know the event type of the notification; this is especially important for devices that support the AE-N-A BIBB.

Devices that at present are unable to parse unknown event type parameters should be upgraded to do so.

### **9.3 Do not implement the Event Enrollment object’s Recipient property**

Do not implement the Recipient property of Event Enrollment objects. This property was removed in Addendum 135a-4 to BACnet-2001. Use a Notification\_Class object instead.

### **9.4 Support the ‘device’ form of BACnetRecipient and the DM-DDB-A BIBB**

The BTL requires that for devices that generate alarms or events (i.e. support the AE-NI-B, AE-NE-B, or T-ATR-B BIBBs) to receive a BTL Listing, the ‘device’ option (the BACnetObjectIdentifier form) of the ‘Recipient’ parameter of BACnetDestination elements of the Recipient\_List property of the Notification Class object shall be supported. In addition to this the BTL requires support for the DM-DDB-A BIBB (initiate the Who-Is service in order to locate alarm recipients).

The BTL also recommends support for the ‘address’ option (the BACnetAddress form), including the broadcast form of the ‘address’ option.

### **9.5 Do not rely on the GetAlarmSummary service for acknowledging unseen alarms**

GetAlarmSummary does not provide sufficient information to acknowledge alarms not seen by the acknowledging device. Devices that need to acknowledge outstanding alarms should be able to initiate GetEventInformation.

### **9.6 Support the GetEventInformation service**

The BTL requires that for devices that generate alarms or events (i.e. support the AE-NI-B or AE-NE-B BIBBs) to receive a BTL Listing, the devices shall be able to execute the GetEventInformation service.

### **9.7 GetEventInformation requires APDU sizes greater than 50 or segmentation**

Devices that initiate the GetEventInformation service (AE-INFO-A) need to be able to receive APDUs larger than 50 octets, or be able to receive segmented messages, because the GetEventInformation-ACK (response) will not fit in a 50-octet APDU.

Likewise, devices that execute the GetEventInformation service (AE-INFO-B) need to be able to send APDUs larger than 50 octets, or be able to transmit segmented messages.

**9.8 The Notification Class object’s Recipient\_List property should be persistent**

The contents of the Notification Class object’s Recipient\_List property should not be lost when the device is powered down or reset. There is no standard mechanism for restoring this information from some other source when the device starts up.

**9.9 B-OWS devices should be able to modify standard alarm properties of standard objects**

Operator workstations claiming the BOWS device profile should be able to modify standard alarm properties of standard objects in order to change alarm configurations. These properties are:

Alarm_Value	Event_Enable	High_Limit	Notification_Class
Alarm_Values	Event_Parameters	Life_Safety_Alarm_Values	Notify_Type
Deadband	Fault_Values	Limit_Enable	Time_Delay
Error_Limit	Feedback_Value	Low_Limit	

Workstations should also be able to modify the Event\_Type property of devices implemented prior to Addendum 135c-3 to BACnet-2001.

**9.10 Unless fixed by the application, standard alarm parameters should be writable**

Unless fixed by the device’s application, the standard properties presenting alarm parameters should be writable to allow modification by workstations. These properties are:

Alarm_Value	Event_Enable	High_Limit	Notification_Class
Alarm_Values	Event_Parameters	Life_Safety_Alarm_Values	Notify_Type
Deadband	Fault_Values	Limit_Enable	Time_Delay
Error_Limit	Feedback_Value	Low_Limit	

**9.11 Recipient\_List shall be writable**

The BTL requires that for devices that generate alarms or events (i.e. support the AE-NI-B, AE-NE-B or T-ATR-B BIBBs) to receive a BTL Listing, the Notification Class object’s Recipient\_List property shall be writable.

**9.12 Support both ConfirmedEventNotification and UnconfirmedEventNotification**

The BTL requires that for a device that generates alarms or events (i.e. supports the AE-NI-B or AE-NE-B BIBBs) to receive a BTL Listing, the device shall be able to initiate both ConfirmedEventNotification and UnconfirmedEventNotification service requests.

Likewise, the BTL requires that for a device receives alarms or events (i.e. supports the AE-N-A BIBB) to receive a BTL Listing, the device shall be able to execute both ConfirmedEventNotification and UnconfirmedEventNotification service requests.

This reflects the requirements of the AE-NI-B and AE-N-A BIBBs.

**9.13 UnconfirmedEventNotification Execution requires APDU sizes greater than 50**

Devices that execute the UnconfirmedEventNotification service (AE-N-A) shall be able to receive APDUs larger than or equal to 128 octets, because many of the event types conveyed by an UnconfirmedEventNotification request will not fit in a 50-octet APDU.

**9.14 Accept any Acknowledging Process Identifier in the AcknowledgeAlarm service request**

The BTL requires that a device accept any Acknowledging Process Identifier parameter in the AcknowledgeAlarm service.

Some implementations have required that the Acknowledging Process Identifier match a configured recipient of an alarm for the AcknowledgeAlarm service request to succeed. This prevents operator workstations that learn of an alarm by means other than receiving an alarm from acknowledging the alarm.

### **9.15 Life Safety objects may advertise supported modes despite Protocol\_Revision**

The BTL permits devices claiming a Protocol\_Revision less than 4 to advertise supported life safety modes in the Life Safety Point and Life Safety Zone object's Accepted\_Modes property (introduced in Addendum 135-2001c-1).

### **9.16 LifeSafetyOperation execution Unsilence operations require Protocol\_Revision 4**

The BTL requires that devices in which the Unsilence operations (unsilence, unsilence-audible, unsilence-visual) support execution of the 'request' parameter of the LifeSafetyOperation service request (introduced in Addendum 135-2001c-2) to claim Protocol\_Revision 4 or higher.

### **9.17 The Event\_Parameters property sets the event type, the Event\_Type property indicates it**

Addendum 135-2001c-3 (Protocol\_Revision 4) made it clear that the event type of the Event Enrollment object is set by writing its Event\_Parameters property and that the Event\_Type property indicates the event type. Devices that claim a Protocol\_Revision less than 4 should implement this relationship between the two properties.

### **9.18 Use the event priorities found in Annex M**

The BTL suggests that the event priorities defined in Annex M (introduced in Addendum 135-2001c-6) be implemented, independent of the Protocol\_Revision claimed by the device.

### **9.19 The Life Safety object behaviour in Addendum 135-2004a-1 requires Protocol\_Revision 5**

The BTL requires that devices supporting Life Safety Point and Life Safety Zone objects implementing the behaviours described in Addendum 135-2004a-1 claim Protocol\_Revision 5.

### **9.20 To indicate “all day” in BACnetDestination use 00:00:00.00 to 23:59:59.99**

For client devices configuring a BACnetDestination, indicate that a particular destination is viable all day by using 00:00:00.00 for 'From Time' and 23:59:59.99 for 'To Time'. The intent here is unambiguous, unlike the use of 00:00:00.00 for 'To Time' which can be interpreted as "active only at 00:00:00.00."

### **9.21 If the alarm support properties are present in an object, it shall support intrinsic alarming**

The BTL requires that if the properties which support intrinsic alarming are present in an object, it shall support intrinsic alarming.

## **10 Scheduling**

The following guidelines apply to devices that contain Calendar and Schedule objects, and to workstations that can modify those objects.

### **10.1 Schedules should be modifiable**

Unless the application dictates otherwise the Weekly\_Schedule and Exception\_Schedule properties of the Schedule object and the Date\_List property of the Calendar object should be modifiable, preferably using standard BACnet services. This allows operator workstations from multiple manufacturers to modify schedules.

### **10.2 Schedules should be capable of numerous weekday entries and exception entries**

The Scheduling BIBBs (K.3.2 in the BACnet standard) require only six entries per day in the Weekly\_Schedule and one entry in the Exception\_Schedule. For improved usefulness, unless the application dictates otherwise, devices should support more entries than that, especially in the Exception\_Schedule.

### **10.3 Schedule objects should not be used to schedule complex or proprietary datatypes**

Schedule objects should not be configured to schedule complex (“constructed”) or proprietary datatypes, for better interoperability with operator workstations.

### **10.4 Schedule objects should schedule certain basic datatypes**

The BTL has determined that Schedule objects whose List\_Of\_Object\_Property\_References property can be changed, especially if by standard BACnet services, should be able to schedule at least a minimum set of datatypes, for improved interoperability. These datatypes are:

NULL        Unsigned    REAL  
BOOLEAN    INTEGER    Enumerated

If a Schedule object supports only SCHED-I-B (that is, it only writes to properties within the same device as the Schedule object), if its List\_Of\_Object\_Property\_Reference can be changed and there are any commandable objects in the device, then the Schedule object should be capable of scheduling NULL values.

### **10.5 Workstations shall be able to configure schedules with NULL datatype**

Workstations shall be able to configure Schedule object schedules to write the NULL datatype. This allows the Schedule object to relinquish a value previously written to a priority array.

### **10.6 Schedule objects should at a minimum be able to schedule BACnetBinaryPV**

Unless fixed by the application, Schedule objects should at a minimum be able to schedule the BACnetBinaryPV enumeration, for interoperability.

### **10.7 Workstations should, at a minimum, support Schedules that schedule BACnetBinaryPV**

Operator workstations should, at a minimum, be able to view and modify Schedule objects that schedule the BACnetBinaryPV enumeration.

### **10.8 Schedule objects scheduling Unsigned and INTEGER values should support 32 bit values**

Schedule objects that are able to schedule Unsigned and INTEGER values should be able to schedule, at a minimum, 32 bit values for purposes of interoperability.

#### **10.9 Schedule objects scheduling Enumerated values should support 16 bit values**

Schedule objects that are able to schedule Enumerated values should be able to schedule, at a minimum, 16 bit values for purposes of interoperability.

#### **10.10 Support for Protocol Revision 4 Schedule objects requires at least Protocol Revision 4**

The BTL requires that a device that supports (that is, can contain) Schedule objects conforming to Protocol\_Revision 4 (implemented with the changes appearing in Addendum 135-2001a-1, including the Schedule\_Default property) shall claim protocol revision 4 or higher in its Device object's property Protocol\_Revision. This is required as an unambiguous declaration of the Schedule object's expected operation.

The BTL requires workstations to support (that is, to access Schedule objects in other devices) both Schedule objects conforming to Protocol\_Revision 4 as well as objects conforming to earlier versions of the standard. [Note: the old version is published in electronic format only in the ASHRAE Bookstore's superseded standards section]

#### **10.11 NULL is a valid value for the Schedule\_Default property**

Scheduling prior to revision 4 allowed the use of the NULL value in the TimeValue pair of the Weekly or Exception Schedule in order to indicate that the schedule has completed and control should return to the object. This allows the schedule object to relinquish control of the controlled objects.

In a revision 4 schedule, the only way for the schedule to relinquish control is to place a NULL in the Schedule\_Default property.

## **11 Trending**

The following guidelines apply to devices that contain Trend Log objects or that obtain trending data using the ReadRange service.

### **11.1 Implement the most recent Trend Log and ReadRange**

The Trend Log object and ReadRange service shall be implemented as revised in Addendum *b* to BACnet-2001 (and as found in BACnet-2004). These are more robust in guaranteeing properly ordered data.

### **11.2 Support for Protocol Revision 4 Trend Log objects requires at least Protocol Revision 3**

The BTL requires that a device that supports Trend Log objects conforming to Protocol\_Revision 3 (implemented with the changes appearing in Addendum 135-2001*b*-1) shall claim protocol revision 3 or higher in its Device object's property Protocol\_Revision. This is required as an unambiguous declaration of the Trend Log object's expected operation.

### **11.3 ReadRange clients should keep the 'Count' parameter in the range of Signed16**

The BACnet standard does not restrict the range of values used for the 'Count' parameter of the ReadRange service, though it is unlikely that server devices will be able to handle the full BACnet-encodable range of the INTEGER datatype. Client devices should restrict the range they use to INTEGER16 to ensure interoperation. (A proposal to this effect is being submitted to the BACnet committee.)

## **12 Routing**

The following guidelines apply to BACnet routers.

### **12.1 Drop messages with a Hop Count of zero**

The BTL requires that if a message with a Hop Count with a value of zero is received, it shall be discarded. (A message with a Hop Count of zero should never be seen on a LAN; this is a defence against a non-conformant router.)

### **12.2 Drop messages if Hop Count is decremented past zero**

To accommodate the intent of clause 6.3.2 of the BACnet standard, the BTL requires that if the Hop Count of a received message has a value less than or equal to the amount by which it is about to be decremented (which means that after being decremented it would have a negative or zero value), the message shall be discarded.

### **12.3 Initialize-Routing-Table might not operate as expected**

The Initialize-Routing-Table message can convey device-specific (proprietary) information. Therefore this message might not operate as expected when conveyed between equipment from different manufacturers.

### **12.4 Add known networks to empty Router-Busy-To-Network messages**

The BACnet standard permits a router to initiate Router-Busy-To-Network messages with an empty list of networks. A router that receives such a message (with an empty list) shall add a list of all networks it knows to be reachable through the initiating router before forwarding the Router-Busy-To-Network to its other ports.

### **12.5 Routers should redirect unicast messages**

The BACnet standard does not discuss the issue of redirecting unicast messages incorrectly sent to the wrong router on a network (that is, the message is sent to a router that is not the router to the destination network). The BTL-WG has determined that if a router receives a message from a device on a network that should have gone to a different device on that network, the router should decrement the Hop Count in the message (if present), add SNET information to the NPCI (if not present), and forward the message to the correct router.

### **12.6 Routers should not redirect broadcast messages**

The BACnet standard allows a device to locate the router to a remote network by broadcasting a confirmed request on the local network and observing the SA conveyed in the response (Clause 6.5.3, paragraph 2). In order to avoid unnecessary duplicate messages, a router should drop confirmed-request messages sent with a broadcast MAC address where the destination network is not reachable through the router.

### **12.7 Routers should forward unknown network message types**

The BACnet standard provides for a Reject-Message-To-Network with a reject reason octet of X'03', specifying "It is an unknown network layer message type." This response should only be returned by the device to which the message is addressed, not by an intervening router. (Routers implemented to Protocol\_Revision 4 or higher shall convey unknown network layer message types not addressed to them.) This allows new message types to be conveyed across networks using older routers.

### **12.8 Routers should support the maximum frame size for each supported medium**

Routers should always support the maximum frame size for each supported LAN type, as defined in **Table 6-1** of BACnet-2004.

### **13 Backup and Restore**

The following guidelines apply to devices that can have their configurations backed up, or are capable of backing up and restoring other devices' configurations, per clause 19.1 of the BACnet standard.

#### **13.1 Size the configuration File objects before restoring**

Do not use the "append" version of the AtomicWriteFile service (the 'File Start Position' or 'File Start Record' parameter has the value -1) when restoring a device's configuration in Backup and Restore. Instead, set the file size by writing to the File\_Size or Record\_Count property of the File object, and then write the file data to the appropriate locations or records.

The reason for this is that devices may have fixed file sizes for the configuration File object, preventing the "append" version from restoring the configuration.

#### **13.2 A device should be able to handle an interrupted Restore operation**

In the case where a Restore operation fails, Clause 19.1.3.4 of BACnet-2004 states: "Every attempt shall be made to leave device B in a state that will accept additional restore procedures."

#### **13.3 Empty files shall be backed up and restored if listed in the Configuration\_Files property**

If a file is listed in the Configuration\_Files property it shall be backed up and restored even if its File\_Size property indicates a size of zero.

## **14 Annex J BACnet/IP**

The following guidelines apply to devices that implement Annex J of BACnet-2004, BACnet/IP.

### **14.1 A non-BBMD BACnet/IP device shall be able to register as a Foreign Device**

The BTL requires that a BACnet/IP device that does not contain a BBMD shall be able to register as a Foreign Device with a BBMD. This prevents situations where a BACnet/IP device cannot communicate due to an inability to receive broadcasts.

This capability should be configurable to be enabled or disabled by some means, as the device could reside on the same subnet as a BBMD.

### **14.2 A Broadcast Distribution Table should be able to contain more than four entries**

The BTL requires that a BBMD's Broadcast Distribution Table be able to hold at least four entries. If at all possible the BDT should be much larger, able to contain at least 32 entries. This allows the BBMD to fit better in larger installations.

### **14.3 Two-hop BBMD entries are required – one-hop support is not required**

The BTL is only requiring that a BBMD's Broadcast Distribution Table be able to support a netmask of 255.255.255.255. This is in accordance with observed practices and policies for Annex J installations, where one-hop support (from IP routers) is generally disallowed.

### **14.4 BBMDs shall support Foreign Device registrations**

The BTL requires BBMDs to support Foreign Device registrations which imply a connection across a larger internet. This is to ensure that any time a BBMD is present in an Annex J B/IP network foreign devices can connect to the B/IP network.

### **14.5 A Foreign Device Table should be able to contain at least two entries**

The BTL requires that a BBMD's Foreign Device Table be able to hold at least two entries. If at all possible the BTL recommends support for more entries in an FDT.

### **14.6 A BBMD shall be capable of forwarding Forwarded-NPDU broadcasts to its local subnet**

Annex J.4.5, paragraph four, allows a BBMD that receives a Forwarded-NPDU to omit the broadcast on its local subnet (using the B/IP broadcast address) if no other BACnet devices are present on its local subnet. The BTL requires that a BBMD be capable of forwarding the broadcasts in order to support the presence of other BACnet devices on the subnet.

### **14.7 BACnet/IP devices should support classless addressing**

BACnet/IP devices should support classless IP addressing, because this is frequently encountered in installations.

### **14.8 Routers supporting BACnet/IP shall support fragmentation**

Routers which support Annex J BACnet/IP routers shall support fragmentation (and re-assembly) of IP frames. This permits full-sized Ethernet frames (APDU size 1470 octets) to be conveyed across IP networks.

## **15 Miscellaneous**

This section lists items that don't fit anywhere else.

### **15.1 Context tags greater than 14 shall be properly handled by a parser**

Although early versions of the BACnet standard contained no standard context tags greater than 13, it is not unreasonable to expect context tag numbers across the complete range, including extended tags in the range 15 and higher. Parsers shall handle all tag numbers, including those 15 and higher without crashing. The method for encoding tag numbers of 15 and higher is described in clause 20.2.1.2 of the BACnet standard.

### **15.2 Implement the Abort reasons of Clause 5.4.5.3 (BACnet-2004)**

The BTL suggests that the Abort reasons of Clause 5.4.5.3 (in BACnet-2004, introduced in Addendum 135-2001c-7) be implemented, regardless of the Protocol\_Revision claimed by the device.

### **15.3 Use error codes from higher Protocol\_Revisions levels may be supported**

The BTL suggests that error codes and usages introduced in a later Protocol\_Revision may be implemented regardless of the Protocol\_Revision claimed by the device.

### **15.4 BACnet-EIB/KNX mapping may be done independent of Protocol\_Revision**

The BTL allows a device to implement the BACnet to EIB/KNX mapping defined in Annex H.5 and introduced in Addendum 135-2001d-1 without regard to the Protocol\_Revision claimed by the device.

## **16 Known Issues**

This section lists known issues of interpreting the standard.

### **16.1 Rapid transitions in the OUT\_OF\_RANGE and FLOATING\_LIMIT algorithms**

The exact behavior of the OUT\_OF\_RANGE and FLOATING\_LIMIT algorithms is not explicitly clear in the case when the value of the property transitions "instantaneously" (in a time much shorter than Time\_Delay) between the ranges associated with the High\_Limit and Low\_Limit states.<sup>2</sup> This was considered to be indicative of a fault condition, not normal behavior.

The BACnet committee is evaluating this ambiguity; until it is resolved this situation will be outside the scope of BTL testing. This ambiguity does not impair interoperability.

### **16.2 Interoperation of the Limit\_Enable, Event\_Enable and Event\_State properties**

The interoperation of the Limit\_Enable, Event\_Enable and Event\_State properties is not as clear as it could be. The prevailing interpretation appears to be that the Limit\_Enable and Event\_Enable properties control the reporting of events at the time when Event\_State transitions from one state to another, which Event\_State does regardless of the settings of Limit\_Enable and Event\_Enable.

The BACnet committee is evaluating this ambiguity; until it is resolved this situation will be outside the scope of BTL testing. This ambiguity does not impair interoperability.

---

<sup>2</sup> A frequent point of confusion when considering these algorithms can be avoided by recognizing that the transitions between the states taken on by an algorithm's states is driven by the transitions of a property's value across various limits.

## 17 Resources

Resources for implementers of BACnet devices are listed here.

### 17.1 BACnet standards:

**ANSI/ASHRAE 135-2004** (the BACnet standard) and **ASHRAE 135.1-2003** (the testing standard) are available in print, CD-ROM and downloadable PDF from: <http://resourcecenter.ashrae.org/store/ashrae>. (The CD-ROM is published several months after the print version.)

**Addenda** (additions) and **errata** (corrections) to the BACnet standards (ANSI/ASHRAE 135-1995, 2001 and 2004) can be downloaded from <http://www.ashrae.org>: select "Technology & Standards," then select "Standards Addenda/Errata/Interpretations."

**Interpretation Requests** (clarification of particular issues in the standard) may be downloaded from ASHRAE's website at:

<http://www.ashrae.org/template/AssetDetail/assetid/35572> (BACnet-2001)

<http://www.ashrae.org/template/AssetDetail/assetid/35575> (BACnet-2004)

### ISO Standard 16484-5:

<http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=37298>

### 17.2 The BACnet website

The BACnet committee (ASHRAE/SSPC 135) maintains a website at <http://www.bacnet.org>. This site presents the latest news from the BACnet world, links to various BACnet organizations, a bibliography of BACnet articles, a list of BACnet Vendor IDs plus vendor contact information, and much more.

### 17.3 The BACnet Testing Labs website

The BACnet Testing Labs (BTL) has posted a number of Internet links of use to BACnet implementers on its website at: <http://bacnetinternational.org/btl>. This site also contains information about the BTL's conformance testing and listing program for BACnet devices, as well as the application package for submitting devices for testing.

This site also includes a link for downloading the latest published version of this "BTL Implementers Guide".

### 17.4 Public e-mail lists:

There are several organizational (open to members) and regionally-oriented BACnet e-mail lists maintained by various organizations. There are also a couple of general lists, as follows

**BACnet-L** <http://www.bacnet.org/Contact/BACnet-L.htm>

BACnet-L is a list for general discussions of all things BACnet.

**BTL Announcements** <http://groups.yahoo.com/group/BTL-announcements/join>

The BTL maintains a mailing list for announcing upcoming BTL-related events, such as its Interoperability Testing Workshops (a.k.a. "plugfests"), the release of new versions of its BACnet testing tool, Visual Test Shell (VTS), and upcoming meetings of the BTL working group.