BACnet *Errata*
ANSI/ASHRAE STANDARD 135-2001

1/24/04

This document lists all known *errata* to ANSI/ASHRAE 135-2001 as of the above date. Each entry is cited first by clause, then page number. Entries 1 – 37 were incorporated into a reprinted version of the standard in May 2002. The outside back cover marking identifying the reprint is "86447 PC 5/02 Includes all errata known up to 4/12/2002."

1) 5.4.5.3, p.34-35: In the SendAbort transition description the 'server' parameter should be set to TRUE.

   "…transmit a BACnet-Abort-PDU with 'server' = ~~FALSE~~ *TRUE*…"

2) 10.4.11.2, p. 120: In the DataNak transition description the list of Data Nak message types is incorrect.

   "…and FrameType is equal to Data Nak 0 XOFF, ~~Data Nak 1~~ *Data Nak 0* XON, ~~Data Nak 0~~ *Data Nak 1* XOFF, or Data Nak 1 XON…"

3) 10.4.11.3, p.122: In the Data1_FullBuffers transition the receive sequence number is incorrect.

   "…and RxSequenceNumber is equal to ~~0~~ *1*…"

4) 12.6.19 and 12.6.20, p.156: The clause reference for details about minimum on and minimum off time is incorrect. The correct reference is Clause 19.2.3

   The mechanism by which this is accomplished is described in ~~19.3~~ *19.2.3*.

5) 12.7.17 and 12.7.18, p.161: The clause reference for details about minimum on and minimum off time is incorrect. The correct reference is Clause 19.2.3

   "…the mechanism by which this is accomplished is described in ~~19.3~~ *19.2.3*."

6) 13 paragraph 5, p.233: In the fourth sentence "event notification messages" should be "event-notification messages."

7) 13.8.1, p.253: In table 13-8 the symbol "(=)" is incorrectly applied in the Rsp and Cnf columns.

Table 13-8. Structure of ConfirmedEventNotification Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
…				
Result(+)			S~~(=)~~	S(=)
Result(-)			S~~(=)~~	S(=)
Error Type			M	M*(=)*

8) Clause 23, p. 383, BACnetAbortReason incorrectly states that maximum enumeration size to be 65535.

-- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values 64-~~65535~~*255*
-- may be used by others subject to the procedures and constraints described in Clause 23.

9) Clause 23, p.412-414: Misspelled word in the header.

23. EXTENDING BACnet TO ACCOMMODATE VENDOR PROPRIE~~TA~~TARY INFORMATION

10) Annex A, p 425, Missed word change made in Addendum 135d-2:

~~Additional~~ *List all* BACnet Interoperability Building Blocks Supported (Annex K):

11) Annex A, p. 425, Missed phrases inserted in Addendum 135d-2:

1) Whether objects of this type are dynamically creatable *using the CreateObject service*
2) Whether objects of this type are dynamically deletable *using the DeleteObject service*

12) Annex F, p. 476- 509, Missing space in the header. "ANNEXF" should be "ANNEX F"

13) Clause 21, p. 399, Missing comma after trend-log enumeration and extra comma after life-safety-zone enumeration:

```
…
trend-log           (20),
life-safety-point   (21),
life-safety-zone    (22)
}
```

14) Clause K.5.18, p.547, References to DM-BR-A should be DM-BR-B.
Devices claiming conformance to ~~DM-BR-A~~ *DM-BR-B* must support the device ~~A~~ *B* capabilities as described in Clause 19.1.

15) Annex A Device Address Binding, p.426, "two-way" is misspelled and there is a space missing between the two words "certain" and "other."

Is static device binding supported? (This is currently necessary for ~~tw-way~~ *two-way* communication with MS/TP slaves and ~~certainother~~ *certain other* devices.) ☐Yes ☐ No

16) History of Revisions for Addendum 135*d* to ANSI/ASHRAE 135-1995, p.556, in item one, "Specification" should be "Interoperability."

    1. Replace Clause 22 with a new clause entitled "Conformance and ~~Specification~~ *Interoperability*."

17) Clause 24.2.1 (f), p. 416, "Request" should not be capitalized.

    (f) Once the key server has been authenticated, device B shall decipher SK_{AB} and place it in its Device object. A 'Result(+)' is then returned to the key server for the AddListElement ~~Request~~ *request*.

18) Clause 24.2.1 (i), p. 416, "Request" should not be capitalized.

    (i) Once the key server has been authenticated, client A shall decipher SK_{AB} and place it in its Device object. A 'Result(+)' is then returned to the key server for the AddListElement ~~Request~~ *request*.

19) Clause 24.2.3 (b), p. 416, "InvokeID" should not be capitalized.

    (b) Client A generates an Authenticate service request and sends it to Server B. The 'Expected Invoke ID' parameter of this message shall contain the ~~InvokeID~~ *invokeID* of the Confirmed Request APDU being authenticated. The Service Request portion of the BACnet-Confirmed-Request-PDU, i.e., the fully formed Authenticate-Request production, is enciphered with SK_{AB}. The APDU header and all lower layer PCI are left unenciphered.

20) Clause 24.5.2.1.1, p. 421, is incorrectly numbered.

    ~~24.5.2.1.1~~ *24.5.2.1* Message Execution Authentication

21) History of Revisions for Addendum 135*e* to ANSI/ASHRAE 135-1995, p.557, in item one, connection and cannot are misspelled.

    1. Define the PTP ~~conection~~ *connection* status when the half-router can and ~~canot~~ *cannot* re-establish the connection.

22) Clause 16.8.1, p. 315, Clause 16.9.1, p.316, Clause 16.9.3, p. 317, Clause 16.10.1, p. 318, Clause 16.10.3, p. 319, Tables 16-8 through 16-11 are incorrectly numbered. When the UTCTimeSynchronization service was added the table was labeled with as 16-X. This was a placeholder to allow insertion at the proper place and renumbering subsequent tables. The correction is to identify this table as 16-8 and to renumber subsequent tables.

16.8.1 Structure

The structure of the UTCTimeSynchronization service primitive is shown in Table ~~16-X~~ *Table 16-8*. The terminology and symbology used in this table are explained in 5.6.

Table ~~16-X~~ *16-8*. Structure of UTCTimeSynchronization Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Time	M	M(=)

16.9.1 Who-Has Service Structure

The structure of the Who-Has service primitive is shown in Table ~~16-8~~ *16.9*. The terminology and symbology used in this table are explained in 5.6.

Table ~~16-8~~ *16.9*. Structure of Who-Has Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Device Instance Range Low Limit	U	U(=)
Device Instance Range High Limit	U	U(=)
Object Identifier	S	S(=)
Object Name	S	S(=)

16.9.3 I-Have Service Structure

The structure of the I-Have service primitive is shown in Table ~~16-9~~ *16.10*. The terminology and symbology used in this table are explained in 5.6.

Table ~~16-9~~ *16.10*. Structure of I-Have Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Device Identifier	M	M(=)
Object Identifier	M	M(=)
Object Name	M	M(=)

16.10.1 Who-Is Service Structure

The structure of the Who-Is service primitive is shown in Table ~~16-10~~ *16.11*. The terminology and symbology used in this table are explained in 5.6.

Table ~~16-10~~ *16.11*. Structure of Who-Is Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Device Instance Range Low Limit	U	U(=)
Device Instance Range High Limit	U	U(=)

16.10.3 I-Am Service Structure

The structure of the I-Am service primitive is shown in Table ~~16-11~~ *16-12*. The terminology and symbology used in this table are explained in 5.6.

Table ~~16-11~~ *16-12*. Structure of I-Am Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
I-Am Device Identifier	M	M(=)
Max APDU Length Accepted	M	M(=)
Segmentation Supported	M	M(=)
Vendor Identifier	M	M(=)

23) Clause 5.2.1.3, p. 19, has a misspelled word in the heading.

5.2.1.3 ~~Maximun~~ *Maximum* Segments Accepted

24) Clause 5.3.5.3 (c), p. 23, should have a period at the end of the paragraph.

(c) The client receives a duplicate ComplexACK segment, that is, one that has already been acknowledged with a SegmentACK. In this case, the client shall discard the duplicate segment but shall return an appropriate SegmentACK APDU. A segment can be identified uniquely by the peer address, Invoke Id, and Sequence Number of the segment~~;~~.

25) Clause 10.4.7.3, p. 110, FrameType X'3' should be X'13'.

FrameType

If ReceiveError is FALSE and DataAvailable is TRUE and Index is 0 and the content of DataRegister is not equal to X'10', X'11', or ~~X'3'~~ *X'13'*,

26) Clause 17.5.3, p. 332, Figure 17-5 is missing two ellipses that indicate the pattern of tab

```
         1         2         3         4         5
12345678901234567890123456789012345678901234567890 ...
       T       T       T       T       T       T ...
```

Figure 17-5. VT tab positions.

stops continues.

27) Clause 6.6.4, p. 63, the clause number is mistakenly repeated in the heading.

6.6.4 ~~6.6.4~~ Router Congestion Control

28) Clause 9.5.4.2, p. 82, the Preamble2 transition should refer to the "content" of the DataRegister.

Preamble2

If ReceiveError is FALSE and DataAvailable is TRUE and the ~~contents~~ *content* of DataRegister is X 'FF',

29) Clause 9.5.6.8, p. 90-91, the ReceiveReplyToPFM transition incorrectly capitalized "Then."

~~Then~~ *then* set SoleMaster to FALSE; set NS equal to SourceAddress; set EventCount to zero; call SendFrame to transmit a Token frame to NS; set PS to the value of TS; set TokenCount and RetryCount to zero; set ReceivedValidFrame to FALSE; and enter the PASS_TOKEN state.

30) Clause 9.9.1, p. 96, missing comma after "… except the one on which the zero was detected."

When a zero is detected, the repeater disables all receivers, except the one on which the zero was ~~detected~~ *detected,* and enables all transmitters, except the one associated with the port on which the zero was detected. The repeater then enters the state associated with the active receiver port. If a zero is detected simultaneously on more than one port, the method used to arbitrate between them is a local matter.

31) Clause 22.1.1.1, p. 409, the introductory paragraph should read "At a minimum" for the PICS information requirements.

~~As~~ *At* a minimum, a BACnet PICS shall convey the following information.

32) Clauses 16.9.1.1.1, 16.9.1.1.2, and 16.9.1.1.3, p. 316 and 16.9.1.1.4, p. 317, all contain clause references of the form16.8.xxx which should be 16.9.xxx.

33) Clauses 16.10.3.1.2, 16.10.3.1.3, and 16.10.3.1.4, p. 319, all contain references of the form 12.9.x that should be 12.10.x.

34) Clauses 16.7.2, p. 314 and D.10, p. 445, Time_Synchronization_Recipients is misspelled in several places.

16.7.2 Service Procedure

…

No restrictions on the use of this service exist when it is invoked at the request of an operator. Otherwise, the use of this service is controlled by the value of the ~~Time_Synchronization_Recipient~~ *Time_Synchronization_Recipients* property in the

Device object. When the ~~Time_Synchronization_Recipient~~ *Time_Synchronization_Recipients* list is of length zero, a device may not automatically send a TimeSynchronization request. When ~~Time_Synchronization_Recipient~~ *Time_Synchronization_Recipients* list is of length one or more, a device may automatically send a TimeSynchronization request but only to the devices or addresses contained in the ~~Time_Synchronization_Recipient~~ *Time_Synchronization_Recipients* list.

D.10 Examples of a Device Object

…

Property: ~~Time_Synchronization_Recipient~~ *Time_Synchronization_Recipients* = (Device, Instance 18)

35) Clause 16.8.2, p. 315, Time_Synchronization_Recipients is misspelled in several places and references to TimeSynchronization request should be UTCTimeSynchronization request.

16.8.2 Service Procedure

…

No restrictions on the use of this service exist when it is invoked at the request of an operator. Otherwise, the use of this service is controlled by the value of the ~~Time_Synchronization_Recipient~~ *Time_Synchronization_Recipients* property in the Device object. When the ~~Time_Synchronization_Recipient~~ *Time_Synchronization_Recipients* list is of length zero, a device may not automatically send a *UTC*TimeSynchronization request. When ~~Time_Synchronization_Recipient~~ *Time_Synchronization_Recipients* list is of length one or more, a device may automatically send a *UTC*TimeSynchronization request but only to the devices or addresses contained in the ~~Time_Synchronization_Recipient~~ *Time_Synchronization_Recipients* list.

36) Revision History, p. 557, A new entry is added to clarify that ANSI/ASHRAE 135-2001 is a consolidated version containing all addenda through Addendum e to ANSI/ASHRAE 135-1995.

1	2	**ANSI/ASHRAE 135-2001** A consolidated version of the standard that incorporated all of the known errata and revisions up through Addendum e to ANSI/ASHRAE 135-1995.

37) Clause 25, p. 424, Periods are missing at the end of two references.

Echelon: Echelon Corporation, 4015 Miranda Ave., Palo Alto, CA ~~94304~~ 94304.

…

IEEE: The Institute of Electrical and Electronics Engineers, Inc., 345 E. 47th St., New York, NY ~~10017~~ 10017.

38) Annex C, p. 428-438, The ASN.1 tag value for Profile_Name property in the examples is incorrect. The correct value is 168.

39) Cause 12, p. 129, PROCESS_ERROR was omitted from the list of values for Reliability.

PROCESS_ERROR	*A processing error was encountered.*

40) 5.4.4.4, p. 30 LastSegmentOfGroupReceived
The InitialSequenceNumber needs to be set to LastSequenceNumber, similar to the LastSegmentOfGroupReceived case on the server side (5.4.5.2, p. 33).

Change
"then save the BACnet-ComplexACK-PDU segment; increment LastSequenceNumber, modulo 256; issue an N-UNITDATA.request..."
to
"then save the BACnet-ComplexACK-PDU segment; increment LastSequenceNumber, modulo 256; set InitialSequenceNumber to LastSequenceNumber; issue an N-UNITDATA.request..."

41) Annex D p.439 Clause 11 should be Clause 12.
This annex provides examples of the BACnet standard object types defined in Clause ~~11~~*12*.

42) Clause 12.4.13, p.147, improperly capitalized word and extra comma in the last sentence.

If an implementation supports writing to 'Object_Property_Reference', then ~~If~~ *if* 'Object_Property_Reference' is written to using BACnet services, then all of the buffer samples shall become invalid, 'Attempted_Samples' shall become zero, 'Valid_Samples' shall become zero,~~,~~ 'Minimum_Value' shall become INF, 'Average_Value' shall become NaN and 'Maximum_Value' shall become -INF.

43) Foreword, p. vii, The date in the heading should be 2001.

Foreword to ANSI/ASHRAE 135-~~1995~~ *2001*

44) Annex A, p.425, clarification of segmentation capability language.

Segmentation Capability:

☐ ~~Segmented requests supported~~*Able to transmit segmented messages*
☐ ~~Segmented responses supported~~*Able to receive segmented messages*

45) Clause 6.7.1.3, p.64, Addendum *e* to BACnet 1995 added clarifying language for half routers. One of the places this language should have appeared is 6.7.1.3.

6.7.1.3 Answering Half-Router Procedure
Upon connection establishment from a PTP half-router, the answering half-router shall set an activity timer (T_{active}) based upon a received Establish-Connection-To-Network message from the initiating half-router, synchronize its routing table with the routing table of the

initiating half-router partner using the procedures in 6.7.3, and broadcast an I-Am-Router-To-Network message containing all of the DNETs accessible through the initiating half-router to all directly connected networks. If the connection is terminated, the answering half-router's routing table shall be adjusted to indicate that any DNET accessible from the initiating half-router has a "reachability status" that is "temporarily ~~unreachable."~~ *unreachable," if the answering half-router is able to re-establish the connection or "permanently unreachable" if the answering half-router is unable to re-establish the connection.* If the connection is established, the answering half-router's routing table shall be adjusted to indicate that any DNET accessible from the initiating half-router has a "reachability status" that is "reachable."

46) Clause 20.2.16, p.365, Error in tags for encoding example.

Example: SEQUENCE value

ASN.1 =	BACnetDateTime
Value =	January 24, 1991, 5:35:45.17 P.M.
Application Tag =	Date (Tag Number = ~~9)~~ *10)*
Encoded Tag =	~~X'94'~~ *X'A4'*
Encoded Data =	X'5B011805'
Application Tag =	Time (Tag Number = ~~10)~~ *11)*
Encoded Tag =	~~X'A4'~~ *X'B4'*
Encoded Data =	X'11232D11'

Example: Context-tagged SEQUENCE value

ASN.1 =	[0] BACnetDateTime	
Value =	January 24, 1991, 5:35:45.17 P.M.	
Context Tag =	0	
Encoded Tag =	X'0E' (opening tag)	
	Application Tag =	Date (Tag Number = ~~9)~~ *10)*
	Encoded Tag =	~~X'94'~~ *X'A4'*
	Encoded Data =	X'5B011805'
	Application Tag =	Time (Tag Number = ~~10)~~ *11)*
	Encoded Tag =	~~X'A4'~~ *X'B4'*
	Encoded Data =	X'11232D11'
Encoded Tag =	X'0F' (closing tag)	

47) Clause 5.4.5.1, p.31, Clarification of how a server responds to a confirmed request with a multicast or broadcast address. Clause 6.3 prohibits the transmission of such messages.

5.4.5 State Machine for Responding BACnet User (server)

5.4.5.1 IDLE

In the IDLE state, the device waits for a PDU from the network layer.

UnconfirmedReceived
If a BACnet-Unconfirmed-Request-PDU is received from the network layer,

then send an UNCONF_SERV.indication to the local application program, and enter the IDLE state.

ConfirmedBroadcastReceived
If a BACnet-Confirmed-Request-PDU whose destination address is a multicast or broadcast address is received from the network layer,

then enter the IDLE state.

…

48) Clause 13.15.2, p.270 paragraph 2, Correction to make this clause consistent with 13.15.1.4 and 13.15.1.5.

[Change clause 13.15.2]

… ~~If the 'Lifetime' parameter is not present but the 'Issue Confirmed Notifications' parameter is present, then a value of zero (indefinite lifetime) shall be assumed for the lifetime.~~ …

49) Clause F.1.8, p.482, Incorrect encoding for PDU type, SD Tag 0, and Priority. Invoke ID=01 should be Invoke ID=1.

F.1.8 Encoding for Example E.1.8 - GetEventInformation Service

X'~~00~~*02*' PDU Type = 0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=1)
X'02' Maximum APDU Size Accepted = 206 octets
X'01' Invoke ID = 1
X'1D' Service Choice = 29 (GetEventInformation)

Assuming the service procedure executes correctly, a complex acknowledgment is returned:

X'30' PDU Type = 3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'01' Invoke ID=~~0~~1
X'1D' Service ACK Choice = 29, (GetEventInformation-ACK)

X'0E' PD opening Tag 0
X'0C' SD context Tag 0 (ObjectIdentifier, L=4)
X'00000002' Analog Input, Instance Number = 2
X'19' SD context Tag 1 (Enumerated, L=1)
X'03' 3 (HIGH_LIMIT)
X'2A' SD context Tag 2 (Bit String, L=2)
X'0560' 0,1,1 (FALSE, TRUE, TRUE)
X'3E' PD opening Tag 3
X'~~0D~~0C' SD context Tag 0 (Time L=4)

X'0F230014'	Time 15:35:00.20
X'~~0D~~0C'	SD context Tag 0 (Time L=4)
X'FFFFFFFF'	Time unspecified
X'~~0D~~0C'	SD context Tag 0 (Time L=4)
X'FFFFFFFF'	Time unspecified
X'3F'	PD closing Tag 3
X'49'	SD context Tag 4 (Enumerated, L=1)
X'00'	0 (ALARM)
X'5A'	SD context Tag 5 (Bit String, L=2)
X'05E0'	1,1,1 (TRUE, TRUE, TRUE)
X'6E'	PD opening Tag 6
X'21'	Application Tag 2 (Unsigned Integer, L=1)
X'~~0E~~0F'	15 (Priority)
X'21'	Application Tag 2 (Unsigned Integer, L=1)
X'~~0E~~0F'	15 (Priority)
X'21'	Application Tag 2 (Unsigned Integer, L=1)
X'14'	20 (Priority)
X'6F'	PD closing Tag 6
X'0C'	SD context Tag 0 (ObjectIdentifier, L=4)
X'00000003'	Analog Input, Instance Number = 3
X'19'	SD context Tag 1 (Enumerated, L=1)
X'00'	0 (NORMAL)
X'2A'	SD context Tag 2 (Bit String, L=2)
X'05C0'	1,1,0 (TRUE, TRUE, FALSE)
X'3E'	PD opening Tag 3
X'~~0D~~0C'	SD context Tag 0 (Time L=4)
X'0F280000'	Time 15:40:00.00
X'~~0D~~0C'	SD context Tag 0 (Time L=4)
X'FFFFFFFF'	Time unspecified
X'~~0D~~0C'	SD context Tag 0 (Time L=4)
X'0F2D1E1E'	~~Time~~ 15:45:30.30
X'3F'	PD closing Tag 3
X'49'	SD context Tag 4 (Enumerated, L=1)
X'00'	0 (ALARM)
X'5A'	SD context Tag 5 (Bit String, L=2)
X'05E0'	1,1,1 (TRUE, TRUE, TRUE)
X'6E'	PD opening Tag 6
X'21'	Application Tag 2 (Unsigned Integer, L=1)
X'~~0E~~0F'	15 (Priority)
X'21'	Application Tag 2 (Unsigned Integer, L=1)
X'~~0E~~0F'	15 (Priority)
X'21'	Application Tag 2 (Unsigned Integer, L=1)
X'14'	20 (Priority)
X'6F'	PD closing Tag 6
X'0F'	PD closing Tag 0
X'19'	SD context Tag 1 (Boolean, L=1)
X'00'	FALSE (More Events)

50) Clause F.1.9, p.484, "Monitored Object Identifier" should be "Requesting Source."

F.1.9 Encoding for Example E.1.9 - LifeSafetyOperation

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'	Maximum APDU Size Accepted=206 octets
X'0F'	Invoke ID=15
X'1B'	Service Choice=27 (LifeSafetyOperation-Request)
X'09'	SD Context Tag 0 (Requesting Process Identifier, L=1)
X'12'	18
X'1C'	SD Context Tag 1 (~~Monitored Object Identifier~~ *Requesting Source*, L=4)

…

51) Clause F.1.11, p.485, Incorrect encoding for 'COV Increment' tag.

…

X'4F'	PD Closing Tag 4 (Property Identifier)
X'~~59~~*5C*'	SD Context Tag 5 (COV Increment, L=4)
X"3F800000'	1.0

…

52) Clause F.3.6, p.495 Example 4, Incorrect application tags for character strings (two occurrences).

Example 4: Finding the identifiers of all objects with Object_Names that match the comparison values "C* Pressure" and "AC? Supply Temp".

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'04'	Maximum APDU Size Accepted=1024 octets
X'54'	Invoke ID=84
X'0D'	Service Choice=13 (ReadPropertyConditional Request)
X'0E'	PD Opening Tag 0 (Object Selection Criteria)
X'09'	SD Context Tag 0 (Selection Logic, L=1)
X'01'	1 (OR)
X'1E'	PD Opening Tag 1 (List Of Selection Criteria)
X'09'	SD Context Tag 0 (Property Identifier, L=1)
X'4D'	77 (OBJECT_NAME)
X'29'	SD Context Tag 2 (Relation Specifier)
X'00'	0 (EQUAL)
X'3E'	PD Opening Tag 3 (Comparison Value)
X'~~7D~~*75*'	Application Tag 7 (Character String, L>4)
X'0C'	Extended Length=12
X'00'	ANSI X3.4 Encoding
X'432A205072657373757265'	"C* Pressure"
X'3F'	PD Closing Tag 3 (Comparison Value)

X'09'	SD Context Tag 0 (Property Identifier, L=1)
X'4D'	77 (OBJECT_NAME)
X'29'	SD Context Tag 2 (Relation Specifier)
X'00'	0 (EQUAL)
X'3E'	PD Opening Tag 3 (Comparison Value)
X'~~7D~~*75*'	Application Tag 7 (Character String, L>4)
…	

53) Clause F.3.8, p.499, Trend Log incorrectly encoded, Invoke ID=01 should be Invoke ID=1, and two times are incorrectly encoded.

Example 1: Reading records from a Trend Log object.

X'02'	PDU Type = 0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=1)
X'02'	Maximum APDU Size Accepted = 206 octets
X'01'	Invoke ID = 1
X'1A'	Service Choice = (n), (ReadRange-Request)
X'0C'	SD Context Tag 0 (Object Identifier, L=4)
X'0~~48~~*50*00001'	Trend Log, Instance Number = 1
X'19'	SD Context Tag 1 (Property Identifier, L=1)
X'83'	131 (LOG_BUFFER)
X'5E'	PD Opening Tag 5 (Time Range)
X'A4'	Application Tag 10 (Date, L=4)
X'620317FF'	March 23, 1998 (Day Of Week Unspecified)
X'B4'	Application Tag 11, (Time, L=4)
X'13342200'	19:52:34.0
X'A4'	Application Tag 10 (Date, L=4)
X'620317FF'	March 23, 1998 (Day Of Week Unspecified)
X'B4'	Application Tag 11, (Time, L=4)
X'13392200'	19:57:34.0
X'5F'	PD Closing Tag 5 (Time Range)

Assuming the service procedure executes correctly, a complex acknowledgment is returned containing the requested data:

X'30'	PDU Type = 3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'01'	Invoke ID=~~0~~1
X'1A'	Service ACK Choice = (n), (ReadRange-ACK)
X'0C'	SD Context Tag 0 (Object Identifier, L=4)
X'0~~48~~*50*00001'	Trend Log, Instance Number = 1
X'19'	SD Context Tag 1 (Property Identifier, L=1)
X'83'	131 (LOG_BUFFER)
X'3A'	SD Context Tag 3 (Result Flags, L=2)
X'05C0'	1,1,0 (TRUE, TRUE, FALSE)

X'49'	SD Context Tag 4 (Item Count, L=1)
X'02'	2
X'5E'	PD Opening Tag 5 (Item Data)
X'0E'	PD Opening Tag 0 (Timestamp)
X'A4'	Application Tag 10 (Date, L=4)
X'62031701'	Monday, March 23, 1998
X'B4'	Application Tag 11, (Time, L=4)
X'1336~~2~~1B00'	19:54:27.0
X'0F'	PD Closing Tag 0 (Timestamp)
X'1E'	PD Opening Tag 1 (Log Datum)
X'2C'	SD Context Tag 2 (REAL, L=4)
X'41900000'	18.0
X'1F'	PD Closing Tag 1 (Log Datum)
X'2A'	SD Context Tag 2 (Status Flags, L=2)
X'0400'	0,0,0,0 (FALSE, FALSE, FALSE, FALSE)
X'0E'	PD Opening Tag 0 (Timestamp)
X'A4'	Application Tag 10 (Date, L=4)
X'62031701'	Monday, March 23, 1998
X'B4'	Application Tag 11, (Time, L=4)
X'1338~~2~~*1*B00'	19:56:27.0
…	

54) Clause F.6, p.508, Network numbers should be encoded in the minimum number of octets.

--- Note that all octets from here to the end of the APDU are to be enciphered ---

X'C4'	*Application Tag 12 (Object Identifier, L=4) (Requesting Device Identifier)*
X'02000001'	*Device, Instance Number=1*
X'~~22~~21'	*Application Tag 2 (Unsigned Integer, L=~~2~~1) (Network Number)*
X'~~00~~02'	*2*
X'61'	*Application Tag 6 (Octet String, L=1) (MAC Address)*
X'11'	*17*
X'C4'	*Application Tag 12 (Object Identifier, L=4) (Remote Device Identifier)*
X'02000002'	*Device, Instance Number=2*
X'~~22~~21'	*Application Tag 2 (Unsigned Integer, L=~~2~~1) (Network Number)*
X'~~00~~02'	*2*
X'61'	*Application Tag 6 (Octet String, L=1) (MAC Address)*
X'22'	*34*

55) Clause 21, p.404, Syntax error in BACnetReliability production.

BACnetReliability ::= ENUMERATED {
…
~~multi-state fault (9),~~
multi-state-fault (9),
...
}

56) Clause D.8, p.443, The Calendar Object example was changed to make it consistent with the reference to in D.22.

D.8 Example of a Calendar Object

The following is an example of a CALENDAR object that specifies the holidays for a school district.

Property:	Object_Identifier =	(Calendar, Instance 1)
Property:	Object_Name =	"HOLIDAYS"
Property:	Object_Type =	CALENDAR
Property:	Description =	"1995-*1996* School District Holidays"
Property:	Present_Value =	TRUE
Property:	Date_List =	(*((23-DEC-1995)-(3-JAN-1996)),*
		(19-FEB-1~~995~~*1996*),
		(~~28~~*27*-MAY-1~~995~~*1996*)~~,~~*)*
		~~((24-DEC-1995)-(4-JAN-1996)))~~

This is a calendar called "HOLIDAYS". Holidays are defined *for Christmas Vacation between December 23 and January 3,* for Presidents' Day on February 19, *and* for Memorial Day on May *27.* ~~28, and for~~ ~~Christmas Vacation between December 24 and January 4 of the next year.~~ On these dates the Present_Value of the calendar will be TRUE. On all other dates, the Present_Value will be FALSE. A real school calendar would likely have more members in the Date_List; only three are shown here for simplicity.

57) Clause 13.12, p.263, Missing open parenthesis.

13.12 GetEventInformation Service

The GetEventInformation service is used by a client BACnet-user to obtain a summary of all "active event states". The term "active event states" refers to all event-initiating objects that

((~~a)~~ have an Event_State property whose value is not equal to NORMAL, or
(b)~~b)~~ have an Acknowledged_Transitions property, which has at least one of the bits (TO_OFFNORMAL, TO_FAULT, TO_NORMAL) set to FALSE.

58) Clause J.5.2.1, p. 532, Duplicate "the" and missing period in the last sentence.

… This value will be initialized to the ~~the~~ 2-octet Time-to-Live value supplied at the time of registration.

59) Clause 13.12 (b), p. 263, Acknowledged_Transitions should be Acked_Transitions.

(b) have an ~~Acknowledged~~Acked_Transitions property, which has at least one of the bits (TO_OFFNORMAL, TO_FAULT, TO_NORMAL) set to FALSE.

60) Clause 13.12.1.2.1.1, p. 264, Acknowledged_Transitions should be Acked_Transitions.

13.12.1.2.1.1 Object Identifier
This parameter, of type BACnetObjectIdentifier, shall identify the event-initiating object that has an Event_State property whose value is not equal to NORMAL or has an ~~Acknowledged~~Acked_Transitions property that has at least one of the following bits (TO_OFFNORMAL, TO_FAULT, TO_NORMAL) set to FALSE.

61) Clause 13.12.2 (b), p. 264, Acknowledged_Transitions should be Acked_Transitions.

(b) have an ~~Acknowledged~~Acked_Transitions property that has at least one of the following bits (TO_OFFNORMAL, TO_FAULT, TO_NORMAL) set to FALSE.

62) Clause 21 BACnetEventParameter production, p. 393, Missing comma at the end of the buffer-ready choice.

BACnetEventParameter ::= CHOICE {

-- These choices have a one-to-one correspondence with Event_Type enumeration
…
  buffer-ready   [7] SEQUENCE {
                notification-threshold   [0] Unsigned,
                previous-notification-count [1] Unsigned32
                ~~}~~ *},*
…
  }

63) Clause 12.10.34, p. 173, One possible case for updating the database revision was omitted.

12.10.34 Database_Revision

This property, of type Unsigned, is a logical revision number for the device's database. It is incremented when an object is created, an object is deleted, an object's name is changed, *an object's Object_Identifier property is changed,* or a restore is performed.

64) Clauses 16.1 and 16.1.1.1.1, p. 304, Clarification that supporting ReinitializeDevice execution is not required if DeviceCommunicationControl execution is supported.

16.1 DeviceCommunicationControl Service

The DeviceCommunicationControl service is used by a client BACnet-user to instruct a remote device to stop initiating and responding to all APDUs (except DeviceCommunicationControl ~~or~~ *or, if supported,* ReinitializeDevice) on the communication network or internetwork for a specified duration of time. This service is primarily used by a human operator for diagnostic purposes. A password may be required from the client BACnet-user prior to executing the service. The time duration may be set to "indefinite," meaning communication must be re-enabled by a DeviceCommunicationControl ~~or~~ *or, if supported, a* ReinitializeDevice service, not by time.

16.1.1.1.1 Time Duration

This optional parameter, of type Unsigned16, indicates the number of minutes that the remote device shall ignore all APDUs except DeviceCommunicationControl ~~and~~ *and, if supported,* ReinitializeDevice APDUs. If the 'Time Duration' parameter is not present, then the time duration shall be considered indefinite, meaning that only an explicit DeviceCommunicationControl or ReinitializeDevice APDU shall enable communications. The 'Time Duration' parameter shall be ignored and the time period considered to be indefinite if the 'Enable/Disable' parameter has a value of ENABLE.

65) Clauses 16.1.2, p. 305, Clarification that supporting ReinitializeDevice execution is not required if DeviceCommunicationControl execution is supported.

16.1.2 Service Procedure

After verifying the validity of the request, including the password, the responding BACnet-user shall respond with a 'Result(+)' service primitive and, if the 'Enable/Disable' parameter is DISABLE, discontinue responding to any subsequent messages except DeviceCommunicationControl ~~and~~ *and, if supported,* ReinitializeDevice messages and discontinue initiating messages. Communication shall be disabled until either the 'Time Duration' has expired or a valid DeviceCommunicationControl (with 'Enable/Disable' = ENABLE) ~~or~~ *or, if supported,* a *valid* ReinitializeDevice message is received. If the responding BACnet-user does not have a clock and the 'Time Duration' parameter is not set to "indefinite," the APDU shall be ignored and a 'Result(-)' service primitive shall be issued. If the password is invalid or absent when one is required, the APDU shall be ignored and a 'Result(-)' response primitive shall be issued.

66) Clause 12.1.4, p. 130, Remove unnecessary cross references.

12.1.4 Present_Value

This property, of type REAL, indicates the current value, in engineering units, of the input being measured ~~(see 12.1.12)~~. The Present_Value property shall be writable when Out_Of_Service is TRUE ~~(see 12.1.10)~~.

67) Clause 12.1.7, p. 131, Remove unnecessary cross references.

12.1.7 Status_Flags

…

where:

IN_ALARM	Logical FALSE (0) if the Event_State property ~~(see 12.1.8)~~ has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property ~~(see 12.1.9)~~ is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the Present_Value and Reliability properties are no longer tracking changes to the physical input.
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property ~~(see 12.1.10)~~ has a value of TRUE, otherwise logical FALSE (0).

68) Clause 12.2.4, p. 135, Remove unnecessary cross references.

12.2.4 Present_Value (Commandable)

This property, of type REAL, indicates the current value, in engineering units, of the output. ~~See 12.2.11.~~

69) Clause 12.2.7, p. 136, Remove unnecessary cross references.

12.2.7 Status_Flags

…

where:

IN_ALARM	Logical FALSE (0) if the Event_State property ~~(12.2.8)~~ has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property ~~(12.2.9)~~ is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the physical output is no longer tracking changes to the Present_Value property and the Reliability property is no longer a reflection of the physical output.
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property ~~(12.2.10)~~ has a value of TRUE, otherwise logical FALSE (0)**.**

70) Clause 12.3.4, p. 140, Remove unnecessary cross references.

12.3.4 Present_Value

This property, of type REAL, indicates the current value, in engineering units, of the analog value ~~(see 12.3.10).~~ Present_Value shall be optionally commandable. If Present_Value is commandable for a given object instance, then the Priority_Array and Relinquish_Default properties shall also be present for that instance. The Present_Value property shall be writable when Out_Of_Service is TRUE. ~~(see 12.3.9)~~

71) Clause 12.3.6, p. 141, Remove unnecessary cross references.

12.3.6 Status_Flags

…

where:

IN_ALARM	Logical FALSE (0) if the Event_State property ~~(12.3.7)~~ has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property ~~(12.3.8)~~ is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services.
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property ~~(12.3.9)~~ has a value of TRUE, otherwise logical FALSE (0).

72) Clause 12.5.4, p. 149, Remove unnecessary cross references.

…

The Present_Value property shall be writable when Out_Of_Service is TRUE ~~(see 12.5.10)~~.

73) Clause 12.5.7, p. 149, Remove unnecessary cross references.

12.5.7 Status_Flags

…

where:

IN_ALARM	Logical FALSE (0) if the Event_State property ~~(12.4.8)~~ has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property ~~(12.4.9)~~ is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the Present_Value and Reliability properties are no longer tracking changes to the physical input.
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property ~~(12.4.10)~~ has a value of TRUE, otherwise logical FALSE (0).

74) Clause 12.6.7, p. 154, Remove unnecessary cross references.

12.6.7 Status_Flags

…

where:

IN_ALARM	Logical FALSE (0) if the Event_State property ~~(12.5.8)~~ has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property ~~(12.5.9)~~ is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the physical output is no longer tracking changes to the Present_Value property and the Reliability property is no longer a reflection of the physical output.
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property ~~(12.5.10)~~ has a value of TRUE, otherwise logical FALSE(0).

75) Clause 12.7.4, p. 160, Remove unnecessary cross references.

12.7.4 Present_Value

This property, of type BACnetBinaryPV, reflects the logical state of the Binary Value. The logical state shall be either INACTIVE or ACTIVE. Present_Value shall be optionally commandable. If Present_Value is commandable for a given instance, then the Priority_Array and Relinquish_Default properties shall also be present for that instance. The Present_Value property shall be writable when Out_Of_Service is TRUE ~~(see 12.7.9)~~.

76) Clause 12.7.6, p. 160, Remove unnecessary cross references.

12.7.6 Status_Flags

…

where:

IN_ALARM	Logical FALSE (0) if the Event_State property ~~(12.6.7)~~ has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property ~~(12.6.8)~~ is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services.
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property ~~(12.6.9)~~ has a value of TRUE, otherwise logical FALSE (0).

77) Clause 12.14.4, p. 185, Remove unnecessary cross references.

12.14.4 Present_Value

This property, of type BACnetLifeSafetyState, reflects the state of the Life Safety Point object. The means of deriving the Present_Value shall be a local matter. Present_Value may latch non-NORMAL state values until reset. The Present_Value property shall be writable when Out_Of_Service is TRUE ~~(see 12.14.11)~~.

78) Clause 12.14.8, p. 185, Remove unnecessary cross references.

12.14.8 Status_Flags
…
where:

IN_ALARM	Logical FALSE (0) if the Event_State property ~~(12.14.9)~~ has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property ~~(12.14.10)~~ does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the physical input(s) are no longer tracking changes to the Present_Value property and the Reliability property is no longer a reflection of the physical input(s).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property ~~(12.14.11)~~ has a value of TRUE, otherwise logical FALSE (0).

79) Clause 12.15.4, p. 191, Remove unnecessary cross references.

12.15.4 Present_Value

This property, of type BACnetLifeSafetyState, reflects the state of the Life Safety Zone object. The means of deriving the Present_Value shall be a local matter. Present_Value may latch non-NORMAL state values until reset. The Present_Value property shall be writable when Out_Of_Service is TRUE ~~(see 12.15.11)~~.

80) Clause 12.15.8, p. 191, Remove unnecessary cross references.

12.15.8 Status_Flags
…
where:

IN_ALARM	Logical FALSE (0) if the Event_State property ~~(12.15.9)~~ has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property ~~(12.15.10)~~ does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the physical input(s) are no longer tracking changes to the Present_Value property and the Reliability property is no longer a reflection of the physical input(s).

OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property ~~(12.15.11)~~ has a value of TRUE, otherwise logical FALSE (0).

81) Clause 12.16.6, p. 197, Remove unnecessary cross references.

12.16.6 Status_Flags

…
where:

IN_ALARM	Logical FALSE (0) if the Event_State property ~~(12.13.7)~~ has a value of NORMAL, otherwise Logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property ~~(12.13.8)~~ is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services.
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property ~~(12.13.9)~~ has a value of TRUE, otherwise logical FALSE(0).

82) Clause 12.17.4, p. 203, Remove unnecessary cross references.

12.17.4 Present_Value

This property, of type Unsigned, reflects the logical state of the input. The logical state of the input shall be one of 'n' states, where 'n' is the number of states defined in the Number_Of_States property. The means used to determine the current state is a local matter. The Present_Value property shall always have a value greater than zero. The Present_Value property shall be writable when Out_Of_Service is TRUE ~~(see 12.15.10)~~.

83) Clause 12.17.7, p. 204, Remove unnecessary cross references.

12.17.7 Status_Flags
…
where:

IN_ALARM	Logical FALSE (0) if the Event_State property ~~(12.14.8)~~ has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property ~~(12.14.9)~~ is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the Present_Value and Reliability properties are no longer tracking changes to the physical input.
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property ~~(12.14.10)~~ has a value of TRUE, otherwise logical FALSE (0).

84) Clause 12.18.7, p. 208, Remove unnecessary cross references.

12.18.7 Status_Flags
…
where:

IN_ALARM	Logical FALSE (0) if the Event_State property ~~(12.15.8)~~ has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property ~~(12.15.9)~~ is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the physical output is no longer tracking changes to the Present_Value property and the Reliability property is no longer a reflection of the physical output.
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property ~~(12.15.10)~~ has a value of TRUE, otherwise logical FALSE (0).

85) Clause 12.19.4, p. 212, Remove unnecessary cross references.

12.19.4 Present_Value

This property, of type Unsigned, reflects the logical state of the multi-state value. The logical state of the multi-state value shall be one of 'n' states, where 'n' is the number of states defined in the Number_Of_States property. How the Present_Value is used is a local matter. The Present_Value property shall always have a value greater than zero. Present_Value shall be optionally commandable. If Present_Value is commandable for a given object instance, then the Priority_Array and Relinquish_Default properties shall also be present for that instance. The Present_Value property shall be writable when Out_Of_Service is TRUE ~~(see 12.17.9)~~.

86) Clause 12.19.6, p. 212, Remove unnecessary cross references.

12.19.6 Status_Flags
…
where:

IN_ALARM	Logical FALSE (0) if the Event_State property ~~(12.17.7)~~ has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property ~~(12.17.8)~~ is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services.
	OUT_OF_SERVICE Logical TRUE (1) if the Out_Of_Service property ~~(12.17.9)~~ has a value of TRUE, otherwise logical FALSE (0).

87) Clause 12.21.11, p. 222, Remove unnecessary cross references.

12.21.11 Status_Flags
…

where:

IN_ALARM	The value of this flag shall be logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property ~~(12.21.12)~~ is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the program has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that neither the Program_Change, Program_State nor any other program-specific property may be changed through BACnet services.
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property ~~(12.21.13)~~ has a value of TRUE, otherwise logical FALSE (0).

88) Clause 12.22.8, p. 225, Remove unnecessary cross references.

12.22.8 Exception_Schedule

…

The Period may be a BACnetCalendarEntry or it may refer to a Calendar object ~~as described in 12.8~~. A BACnetCalendarEntry would be used if the Exception_Schedule is specific to this Schedule object, while calendars might be defined for common holidays to be referenced by multiple Schedule objects. Each BACnetCalendarEntry is either an individual date (Date), range of dates (BACnetDateRange), or month/week-of-month/day-of-week specification (BACnetWeekNDay). If the current date matches any of the calendar entry criteria, the Exception Schedule would be activated and the list of BACnetTimeValues would be enabled for use.

89) Clause 12.23.23, p. 231, "NORMAL-NORMAL transition" should be "TO_NORMAL transition."

In the context of Trend Log objects, the value of the Records_Since_Notification property becoming equal to or greater than the value of the Notification_Threshold property shall cause a ~~NORMAL-NORMAL~~ *TO_NORMAL* transition.

90) Clause 16.10.2, p. 318, "Identifier" is misspelled four lines up from the bottom of the page.

"…BACnet-users shall return their Device Object_~~Iidentifier~~*Identifier*…"

91) Clause D.10, p. 445, Active_COV_Subscriptions has two object identifiers that are missing the word "instance."

Property: Active_COV_Subscriptions = ((((0, (Device, Instance 12)), 300),
(~~(Analog Input, 1)~~*(Analog Input, Instance 1),*
Present_Value),TRUE,100,1.0),
(((0, (Device, Instance 40)), 600),
(~~(Analog Input, 1)~~ *(Analog Input, Instance 1),*
Present_Value),TRUE,3,1.5))

92) Clause 12.12, p. 169, Table 12-2 has the wrong datatype for Last_Restore_Time.

Table 12-12. Properties of the Device Object Type

Property Identifier	Property Datatype	Conformance Code
…		
Last_Restore_Time	~~BACnetDateTime~~ *BACnetTimeStamp*	O[7]
…		

93) Clause 21, p. 369, There is a duplicate context tag number and an error in a comment.

BACnet-Confirmed-Request-PDU ::= SEQUENCE {
…
  reserved [4] Unsigned (0..3), -- must be set to zero
  max-segments-accepted [5] Unsigned (0..7), -- as per 20.1.2.4
  max-APDU-length-accepted ~~[5]~~ *[6]* Unsigned (0..15), -- as per 20.1.2.5
  invokeID ~~[6]~~ *[7]* Unsigned (0..255),
  sequence-number ~~[7]~~ *[8]* Unsigned (0..255) OPTIONAL, -- only if segmented msg
  proposed-window-size ~~[8]~~ *[9]* Unsigned (1..127) OPTIONAL, -- only if segmented msg
  service-choice ~~[9]~~ *[10]* BACnetConfirmedServiceChoice,
  service-request ~~[10]~~ *[11]*BACnet-Confirmed-Service-Request OPTIONAL
-- Context-specific tags ~~0..1~~ *0..11* are NOT used in header encoding
  }