

ASHRAE Standing Standard Project Committee 135
Cognizant TC: TC 1.4, Control Theory and Application
SPLS Liaison: Frank E. Jakob

William O. Swan, III, *Chair**
David Robin, *Vice-Chair*
Carl Neilson, *Secretary**
Donald P. Alexander*
Ron E. Anderson
Beauford W. Atwater
David J. Branson
Barry B. Bridges*
Coleman L. Brumley, Jr.*
Ernest C. Bryant*
Steven T. Bushby
James F. Butler
A. J. Capowski
Keith A. Corbett
Jeffrey Cosiol
Troy Cowan

Harsha M. Dabholkar
Sharon E. Dinges*
Dana R. Epperson
Thomas Ertsgaard
Craig P. Gemmill
Daniel P. Giorgis
Ira G. Goldschmidt
Winston I. Hetherington
David G. Holmberg*
Anthony J. Icenhour
Robert L. Johnson*
Stephen Karg*
J. Damian Ljungquist*
James G. Luth*
John J. Lynch

Jerald P. Martocci
H. Michael Newman
Cherisse M. Nicastro
Robert L. Old
Mark A. Railsback
Carl J. Ruther*
Ernest Senior
Patrick F. Sheridan
David G. Shike*
Kevin Sweeney
David B. Thompson*
Daniel A. Traill
J. Michael Whitcomb
David F. White
Grant N. Wichenko
Robert J. Zamojcin

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

ASHRAE STANDARDS COMMITTEE 2008–2009

Hugh F. Crowther, *Chair*
Steven T. Bushby, *Vice-Chair*
Robert G. Baker
Michael F. Beda
Donald L. Brandt
Paul W. Cabot
Kenneth W. Cooper
Samuel D. Cummings, Jr
K. William Dean
Robert G. Doerr
Allan B. Fraser
Nadar R. Jayaraman
Byron W. Jones
Jay A. Kohler

Carol E. Marriott
R. Michael Martin
Merle F. McBride
Frank Myers
H. Michael Newman
Janice C. Peterson
Douglas T. Reindl
Lawrence J. Schoen
Boggarm S. Setty
Bodh R. Subherwal
William F. Walter
Michael W. Woodford
David E. Knebel, *BOD ExO*
Andrew K. Persily, *CO*

Claire B. Ramspeck, *Director of Technology*

SPECIAL NOTE

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

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

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

The Assistant Director of Technology for Standards and Special Projects of ASHRAE should be contacted for:

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

DISCLAIMER

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

ASHRAE INDUSTRIAL ADVERTISING POLICY ON STANDARDS

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

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

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

FOREWORD

Addendum 135*b* to ANSI/ASHRAE Standard 135-2004 contains a number of changes to the current standard. These modifications are the result of change proposals made pursuant to the ASHRAE continuous maintenance procedures and of deliberations within Standing Standard Project Committee 135. The changes are summarized below.

- 135-2004*b*-1. Add a new Event Log object type, p. 1.
- 135-2004*b*-2. Add a new Global Group object type, p. 12. (Removed after third public review.)
- 135-2004*b*-3. Add a new Trend Log Multiple object type, p. 13.
- 135-2004*b*-4. Harmonize the Trend Log object with the new Event Log and Trend Log Multiple objects, p. 26.
- 135-2004*b*-5. Define a means for a device to provide a notification that it has restarted, p. 36.
- 135-2004*b*-6. Define a means to configure a device to periodically send time synchronization messages, p. 39.
- 135-2004*b*-7. Extend the number of character sets supported, p. 41. (Removed after first public review.)
- 135-2004*b*-8. Enable devices other than alarm recipients to acknowledge alarms, p. 42.
- 135-2004*b*-9. Allow MS/TP BACnet Data Expecting Reply frames to be broadcast, p. 43.
- 135-2004*b*-10. Revise the Clause 5 state machines to handle slow servers, p. 45. (Removed after second public review.)
- 135-2004*b*-11. Add new Error Codes and specify usage, p. 46.
- 135-2004*b*-12. Add new Reliability enumeration to objects with a Reliability property, p. 52.

In the following document, language to be added to existing clauses of ANSI/ASHRAE 135-2004 and Addenda is indicated through the use of *italics*, while deletions are indicated by ~~strike through~~. Where entirely new subclauses are added, plain type is used throughout.

SSPC 135 wishes to recognize the efforts of the following people in developing this addendum: Howard Coleman, Wilson Fowle, John L. Hartman, Yoshiyuki Honda, Bernhard Isler, René Kälin, René Quirighetti, and Takeji Toyoda. The committee also appreciates the significant contributions of Steven Bushby, Clifford H. Copass, David M. Fisher, and Carl Neilson.

135-2004b-1. Add a new Event Log object type.

Rationale

There is need for a standard object type to log events.

Addendum 135-2004b-1

[Change **Clause 12** introduction, p. 128, as follows]

...

Several object types defined in this clause have a property called "Reliability." This property is an enumerated datatype that may have different possible enumerations for different object types. The values defined below are a superset of all possible values of the Reliability property for all object types. The range of possible values returned for each specific object is defined in the appropriate object-type definition.

...

CONFIGURATION_ERROR

The object's properties are not in a consistent state.

COMMUNICATION_FAILURE

Proper operation of the object is dependent on communication with a remote sensor or device and communication with the remote sensor or device has been lost.

UNRELIABLE_OTHER

The controller has detected that the present value is unreliable, but none of the other conditions describe the nature of the problem. A generic fault other than those listed above has been detected, e.g., a Binary Input is not cycling as expected.

[Add entirely new **Clause 12.X**. The exact numbering of this clause will be determined when the next edition of the standard is published.]

12.X Event Log Object Type

An Event Log object records event notifications with timestamps and other pertinent data in an internal buffer for subsequent retrieval. Each timestamped buffer entry is called an event log "record."

Each Event Log object maintains an internal, optionally fixed-size buffer. This buffer fills or grows as event log records are added. If the buffer becomes full, the least recent records are overwritten when new records are added, or collection may be set to stop. Event log records are transferred as BACnetEventLogRecords using the ReadRange service. The buffer may be cleared by writing a zero to the Record_Count property. The determination of which notifications are placed into the log is a local matter. Each record in the buffer has an implied SequenceNumber that is equal to the value of the Total_Record_Count property immediately after the record is added.

Logging may be enabled and disabled through the Enable property and at dates and times specified by the Start_Time and Stop_Time properties. Event Log enabling and disabling is recorded in the event log buffer.

Event reporting (notification) may be provided to facilitate automatic fetching of event log records by processes on other devices such as file servers. Support is provided for algorithmic reporting; optionally, intrinsic reporting may be provided.

In intrinsic reporting, when the number of records specified by the Notification_Threshold property has been collected since the previous notification (or startup), a new notification is sent to all subscribed devices.

In response to a notification, subscribers may fetch all of the new records. If a subscriber needs to fetch all of the new records, it should use the 'By Sequence Number' form of the ReadRange service request.

A missed notification may be detected by a subscriber if the 'Current Notification' parameter received in the previous BUFFER_READY notification is different than the 'Previous Notification' parameter of the current BUFFER_READY notification. If the ReadRange-ACK response to the ReadRange request issued under these conditions has the FIRST_ITEM bit of the 'Result Flags' parameter set to TRUE, event log records have probably been missed by this subscriber.

The acquisition of log records by remote devices has no effect upon the state of the Event Log object itself. This allows completely independent, but properly sequential, access to its log records by all remote devices. Any remote device can independently update its records at any time.

Table 12-X1. Properties of the Event Log Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Enable	BOOLEAN	W
Start_Time	BACnetDateTime	O ^{1,2}
Stop_Time	BACnetDateTime	O ^{1,2}
Stop_When_Full	BOOLEAN	R
Buffer_Size	Unsigned32	R
Log_Buffer	List of BACnetEventLogRecord	R
Record_Count	Unsigned32	W
Total_Record_Count	Unsigned32	R
Notification_Threshold	Unsigned32	O ³
Records_Since_Notification	Unsigned32	O ³
Last_Notify_Record	Unsigned32	O ³
Notification_Class	Unsigned	O ³
Event_Enable	BACnetEventTransitionBits	O ³
Acked_Transitions	BACnetEventTransitionBits	O ³
Notify_Type	BACnetNotifyType	O ³
Event_Time_Stamps	BACnetARRAY[3] of if BACnetTimeStamp	O ³
Profile_Name	CharacterString	O

¹ If present, these properties are required to be writable.

² If one of these properties is present, then all shall be present.

³ These properties are required to be present if the object supports intrinsic reporting.

12.X.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet Device that maintains it.

12.X.2 Object_Name

This property, of type CharacterString, shall represent a name for the Object that is unique within the BACnet Device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.X.3. Object_Type

This property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be EVENT_LOG.

12.X.4 Description

This optional property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.X.5 Status_Flags

This property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of an Event Log object. The IN_ALARM and FAULT flags are associated with the values of other properties of this object. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).

FAULT Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

OVERRIDDEN The value of this flag shall be Logical FALSE (0).

OUT_OF_SERVICE The value of this flag shall be Logical FALSE (0).

12.X.6 Event_State

The Event_State property, of type BACnetEventState, is included in order to provide a way to determine if this object has an active event state associated with it. If the object supports intrinsic reporting, then the Event_State property shall indicate the event state of the object. If the object does not support intrinsic reporting, then the value of this property shall be NORMAL. The Event_State property for this object may have either of the following values:

{NORMAL, FAULT}

12.X.7 Reliability

This optional property, of type BACnetReliability, provides an indication of whether the application-specific properties of the object or the process executing the application program are "reliable" as far as the BACnet Device can determine. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, CONFIGURATION_ERROR, COMMUNICATION_FAILURE, UNRELIABLE_OTHER}.

12.X.8 Enable

This property, of type BOOLEAN, indicates and controls whether (TRUE) or not (FALSE) logging of events is enabled. Logging occurs if and only if Enable is TRUE, Local_Time is on or after Start_Time, and Local_Time is before Stop_Time. If Start_Time contains any wildcards, then it shall be considered equal to 'the start of time'. If Stop_Time contains any wildcards, then it shall be considered equal to 'the end of time'. Log_Buffer records of type log-status are recorded without regard to the value of the Enable property.

Attempts to write the value TRUE to the Enable property while Stop_When_Full is TRUE and Record_Count is equal to Buffer_Size shall cause a Result(-) response to be issued, specifying an 'Error Class' of OBJECT and an 'Error Code' of LOG_BUFFER_FULL.

12.X.9 Start_Time

This optional property, of type BACnetDateTime, specifies the date and time at or after which logging shall be enabled by this property. If any of the fields of the BACnetDateTime contain "wildcard" values, then the conditions for logging to be enabled by Start_Time shall be ignored. If Start_Time specifies a date and time after Stop_Time, then logging shall be disabled. If either of the optional properties Start_Time or Stop_Time is present, then both of these properties shall be present. This property shall be writable if present.

12.X.10 Stop_Time

This optional property, of type BACnetDateTime, specifies the date and time at or after which logging shall be disabled by this property. If any of the fields of the BACnetDateTime contain "wildcard" values, then the conditions for logging to be disabled by Stop_Time shall be ignored. If Stop_Time specifies a date and time earlier than Start_Time, then logging shall be disabled. If either of the optional properties Start_Time or Stop_Time is present, then both of these properties shall be present. This property shall be writable if present.

12.X.11 Stop_When_Full

This property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) logging should cease when the buffer is full. When logging ceases because the addition of one more record would cause the buffer to be full, Enable shall be set to FALSE and the event recorded.

If Stop_When_Full is writable, attempts to write the value TRUE to the Stop_When_Full property while Record_Count is equal to Buffer_Size shall result in the oldest Log_Buffer record being discarded, and shall cause the Enable property to be set to FALSE and the event to be recorded.

12.X.12 Buffer_Size

This property, of type Unsigned32, shall specify the maximum number of records the buffer may hold. If writable, it may not be written when Enable is TRUE. The disposition of existing records when Buffer_Size is written is a local matter.

12.X.13 Log_Buffer

This property is a list of up to Buffer_Size timestamped records of datatype BACnetEventLogRecord, each of which conveys the event notification parameters or status changes in the Event Log object. Each record has data fields as follows:

Timestamp The local date and time that the entry was placed into the event log.

LogDatum The notification information, or a change in status or operation of the Event Log object itself.

The choices available for the LogDatum are listed below:

log-status This choice represents a change in the status or operation of the Event Log object. Whenever one of the events represented by the flags listed below occurs, a record shall be appended to the buffer.

LOG_DISABLED	This flag is changed whenever collection of records by the Event Log object is enabled or disabled. It shall be TRUE if Enable is FALSE, or the local time is outside the range defined by Start_Time and Stop_Time, or the addition of this record will cause the buffer to be full and Stop_When_Full is TRUE; otherwise it shall be FALSE.
BUFFER_PURGED	This flag shall be set to TRUE whenever the buffer is deleted by a write of the value zero to the Record_Count property. After this value is recorded in the buffer, the subsequent immediate change to FALSE shall not be recorded.
LOG_INTERRUPTED	This flag indicates that the collection of records by the Event Log object was interrupted by a power failure, device reset, object reconfiguration or other such disruption, such that samples prior to this record might have been missed.
notification	This choice represents an event notification that was received. It consists of the body of the ConfirmedEventNotification or UnconfirmedEventNotification. If the event was generated locally, this shall hold what would be received if the Event Log object existed on a remote device. In such a case the value of the Process Identifier parameter is a local matter.
time-change	This choice, which represents a change in the clock setting in the device, records the number of seconds by which the clock changed. If the number is not known, such as when the clock is initialized for the first time, the value recorded shall be zero.

Also associated with each record is an implied record number, the value of which is equal to Total_Record_Count at the point where the record has been added into the Log_Buffer and Total_Record_Count has been adjusted accordingly. All clients shall be able to correctly handle the case where the event log is reset such that its Total_Record_Count is returned to zero and also the case where Total_Record_Count has wrapped back to one.

The buffer is not network accessible except through the use of the ReadRange service in order to avoid problems with record sequencing when segmentation is required. Attempts to read this property with the ReadProperty-Request or ReadPropertyMultiple-Request shall result in an error specifying an error class of PROPERTY and an error code of READ_ACCESS_DENIED.

12.X.14 Record_Count

This property, of type Unsigned32, shall represent the number of records currently resident in the log buffer. A write of the value zero to this property shall cause all records in the log buffer to be deleted and Records_Since_Notification to be reset to zero. Upon completion, this event shall be reported in the log as the initial entry.

12.X.15 Total_Record_Count

This property, of type Unsigned32, shall represent the total number of records collected by the Event Log object since creation. When the value of Total_Record_Count reaches its maximum possible value of $2^{32} - 1$, the next value it takes shall be one. Once this value has wrapped to one, its semantic value (the total number of records collected) has been lost but its use in generating notifications remains.

12.X.16 Notification_Threshold

This optional property, of type Unsigned32, shall specify the value of Records_Since_Notification at which notification occurs. This property is required if intrinsic reporting is supported by this object.

12.X.17 Records_Since_Notification

This optional property, of type Unsigned32, represents the number of records collected since the previous notification, or since the beginning of logging if no previous notification has occurred. This property is required if intrinsic reporting is supported by this object.

12.X.18 Last_Notify_Record

This property, of type Unsigned32, represents the SequenceNumber associated with the most recently collected record whose collection caused the value of the Records_Since_Notification property to become equal to or greater than the value of the Notification_Threshold property which triggered a notification. If no notification has occurred since logging began, the value of this property shall be zero. This property is required if intrinsic reporting is supported by this object.

12.X.19 Notification_Class

This optional property, of type Unsigned, shall specify the notification class to be used when handling and generating event notifications for this object. The Notification_Class property implicitly refers to a Notification Class object that has a Notification_Class property with the same value. This property is required if intrinsic reporting is supported by this object.

12.X.20 Event_Enable

This optional property, of type BACnetEventTransitionBits, shall convey flags that separately enable and disable reporting of TO-NORMAL and TO-FAULT events. This property is required if intrinsic reporting is supported by this object.

12.X.20.1 Conditions for Generating a TO-NORMAL Event

A TO-NORMAL event is generated if the TO-NORMAL flag is enabled in the Event_Enable property, and either:

- (a) the value of the Records_Since_Notification property changes from being less than the value of Notification_Threshold to being equal to or greater than the value of Notification_Threshold, or
- (b) the value of the Notification_Threshold property changes from being greater than the value of Records_Since_Notification to being equal to or less than the value of Records_Since_Notification, or
- (c) the Reliability property changes to the value NO_FAULT_DETECTED.

12.X.20.2 Conditions for Generating a TO-FAULT Event

A TO-FAULT event is generated if the TO-FAULT flag is enabled in the Event_Enable property and the Reliability property changes to a value other than NO_FAULT_DETECTED.

12.X.21 Acked_Transitions

This optional property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the receipt of acknowledgements for TO-OFFNORMAL, TO-FAULT and TO-NORMAL events. These flags shall be cleared upon the occurrence of the corresponding event and set under any of these conditions:

- (a) upon receipt of the corresponding acknowledgement;
- (b) upon the occurrence of the event if the corresponding flag is not set in the Event_Enable property (meaning event notifications will not be generated for this condition and thus no acknowledgement is expected);
- (c) upon the occurrence of the event if the corresponding flag is set in the Event_Enable property and the corresponding flag in the Ack_Required property of the Notification Class object implicitly referenced by the Notification_Class property of this object is not set (meaning no acknowledgement is expected).

12.X.22 Notify_Type

This optional property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. This property is required if intrinsic reporting is supported by this object.

12.X.23 Event_Time_Stamps

This optional property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last event notifications for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively. Time stamps of type Time or Date shall have 'FF' in each octet and Sequence number time stamps shall have the value 0 if no event notification of that type has been generated since the object was created. This property is required if intrinsic reporting is supported by this object.

12.X.24 Profile_Name

This optional property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name must begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile document named by the remainder of the profile name. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. The definition of the profiles themselves is outside the scope of this standard.

[Change entry in **Table 13-2**, p. 256]

Object Type	Criteria	Event Type
...
Trend Log <i>Event Log,</i> <i>Trend Log,</i> <i>Trend Log Multiple</i>	If Event_State is NORMAL and Records_Since_Notification is <i>greater than or equal</i> to Notification_Threshold	BUFFER_READY
...

[Change entry in **Table 13-3**, p. 257]

Object	Event Type	Notification Parameters	Referenced Object's Properties
...
Trend Log <i>Event Log,</i> <i>Trend Log,</i> <i>Trend Log Multiple</i>	BUFFER_READY	Buffer_Property Previous_Notification Current_Notification	BACnetDeviceObjectPropertyReference ¹ Last_Notify_Record Total_Record_Count
...

¹This parameter conveys a reference to the Log_Buffer property of the ~~Trend Log~~ object.

[Add new **BACnetEventLogRecord** production in **Clause 21**, p.415]

```
BACnetEventLogRecord ::= SEQUENCE {
    timestamp    [0] BACnetDateTime,
    logDatum     [1] CHOICE {
        log-status    [0] BACnetLogStatus,
        notification  [1] ConfirmedEventNotification-Request,
        time-change   [2] REAL
    }
}
```

[Change **BACnetReliability** production in **Clause 21**, p. 429]

```
BACnetReliability ::= ENUMERATED {
    ...
    configuration-error    (10),
    -- enumeration value 11 is reserved for a future addendum
    communication-failure (12),
    ...
}
```

[Change **BACnetObjectType** production in **Clause 21**, p. 421]

```
BACnetObjectType ::= ENUMERATED {
    ...
    event-enrollment      (9),
    event-log              (25),
    file                   (10),
    ...
    trend-log              (20),
    trend-log-multiple     (27),
    -- see life-safety-point (21),
    ...
    -- see pulse-converter (24),
    -- see event-log        (25),
    -- enumeration value 26 is reserved for a future addendum
    -- see trend-log-multiple (27),
    ...
}
-- Enumerated values 0-127 are reserved for definition by ASHRAE. Enumerated values
-- 128-1023 may be used by others subject to the procedures and constraints described
-- in Clause 23.
```

[Change **BACnetObjectTypesSupported** production in **Clause 21**, p. 422]

```
BACnetObjectTypesSupported ::= BIT STRING {
    ...
    event-enrollment      (9),
    -- event-log           (25),
    file                   (10),
    ...
    -- trend-log          (20),
    -- trend-log-multiple (27),
    -- Objects added after 1995
    ...
}
```

```

    pulse-converter      (24) (24),
-- Objects added after 2004
    event-log            (25),
    -- enumeration value 26 is reserved for a future addendum
    trend-log-multiple   (27)
}

```

[Change **BACnetPropertyIdentifier** production in **Clause 21**, pp. 423-428]

```

BACnetPropertyIdentifier ::= ENUMERATED {
    ...
    alarm-values          (7),
    align-intervals       (193),
    all                   (8),
    ...
    elapsed-active-time   (33),
    enable                 (133),  --renamed from previous version
    error-limit           (34),
    ...
    integral-constant-units (50),
    interval-offset       (195),
    -- issue-confirmed-notifications (51),  This property was deleted in version 1 revision 4.
    last-notify-record    (173),
    last-restart-reason   (196),
    life-safety-alarm-values (166),
    ...

    log-device-object-property (132),
    -- see enable            (133),
    log-interval             (134),
    logging-object           (183),
    logging-record          (184),
    logging-type            (197),
    low-limit               (59),
    ...
    present-value           (85),
    -- previous-notify-time (138),  This property was deleted in version 1 revision 3.
    priority                (86),
    ...
    resolution              (106),
    restart-notification-recipients (202),
    scale                   (187),
    ...
    time-of-active-time-reset (114),
    time-of-device-restart   (203),
    time-of-state-count-reset (115),
    time-synchronization-interval (204),
    time-synchronization-recipients (116),
    ...
    tracking-value          (164),
    trigger                 (205),
    units                   (117),
    ...
    utc-offset              (119),
    utc-time-synchronization-recipients (206),
    valid-samples           (146),
}

```

```

...
-- see log-device-object-property          (132),
-- see enable                             (133),
-- see log-interval                       (134),
...
-- see value-change-time                  (192),
-- see align-intervals                    (193),
-- enumeration value 194 is reserved for a future addendum
-- see interval-offset                    (195),
-- see last-restart reason                 (196),
-- see logging-type                       (197),
-- enumeration value 198 is reserved for a future addendum
-- enumeration value 199 is reserved for a future addendum
-- enumeration value 201 is reserved for a future addendum
-- see restart-notification-recipients    (202),
-- see time-of-device-restart              (203),
-- see time-synchronization-interval      (204),
-- see trigger                             (205),
-- see utc-time-synchronization-recipients (206),
-- property values 194, 198, 199, and 201 are reserved for a future addendum.
...
}

```

-- The special property identifiers all, optional, and required are reserved for use in the ReadPropertyConditional and ReadPropertyMultiple services or services not defined in this standard.

-- Enumerated values 0-511 are reserved for definition by ASHRAE. Enumerated values 512-4194303 may be used by others subject to the procedures and constraints described in Clause 23.

[Add to **Annex C**, p.458]

```

EVENT-LOG ::= SEQUENCE {
    object-identifier    [75]    BACnetObjectIdentifier,
    object-name          [77]    CharacterString,
    object-type          [79]    BACnetObjectType,
    description          [28]    CharacterString OPTIONAL,
    status-flags        [111]   BACnetStatusFlags,
    event-state          [36]    BACnetEventState,
    reliability          [103]   BACnetReliability OPTIONAL,
    enable               [133]   BOOLEAN,
    start-time           [142]   BACnetDateTime OPTIONAL,
    stop-time            [143]   BACnetDateTime OPTIONAL,
    stop-when-full      [144]   BOOLEAN,
    buffer-size          [126]   Unsigned32,
    log-buffer           [131]   SEQUENCE OF BACnetEventLogRecord,
    record-count        [141]   Unsigned32,
    total-record-count  [145]   Unsigned32,
    notification-threshold [137] Unsigned32 OPTIONAL,
    records-since-notification [140] Unsigned32 OPTIONAL,
    last-notify-record  [173]   Unsigned32 OPTIONAL,
    notification-class  [17]    Unsigned OPTIONAL,
    event-enable        [35]    BACnetEventTransitionBits OPTIONAL,
    acked-transitions   [0]     BACnetEventTransitionBits OPTIONAL,
    notify-type         [72]    BACnetNotifyType OPTIONAL,
    event-time-stamps   [130]   SEQUENCE OF BACnetTimeStamp OPTIONAL,

```

```

        profile-name          [168]  --accessed as a BACnetARRAY
        }                      CharacterString OPTIONAL

```

[Add a new **Annex D.13**, p.475, and renumber the existing **Annex D.13** and subsequent clauses]

D.13 Example of an Event Log Object

The following is an example of an Event Log object that logs event notifications and which performs buffer-ready notification via intrinsic reporting.

```

Property:  Object_Identifier =      (Event Log, Instance 1)
Property:  Object_Name =           "Event Log"
Property:  Object_Type =           EVENT_LOG
Property:  Description =           "All event notifications"
Property:  Status_Flags =          {FALSE, FALSE, FALSE, FALSE}
Property:  Event_State =           NORMAL
Property:  Reliability =           NO_FAULT_DETECTED
Property:  Enable =                TRUE
Property:  Stop_When_Full =        FALSE
Property:  Buffer_Size =            250
Property:  Log_Buffer =            (((23-MAR-2000,12:32:33.0), (0, (Device, Instance 20), (Analog Input,
                                     Instance 1), (23-MAR-2000,12:32:25.0), 1, 1, OUT_OF_RANGE, "Too
                                     Hot", ALARM, TRUE, NORMAL, HIGH_LIMIT, (105.1, (TRUE, FALSE,
                                     FALSE, FALSE), 2.0, 100.0)), ...))
Property:  Record_Count =          953
Property:  Total_Record_Count =    1000
Property:  Notification_Threshold = 950
Property:  Records_Since_Notification = 3
Property:  Last_Notify_Record =    950
Property:  Notification_Class =    1
Property:  Event_Enable =          {FALSE, TRUE, TRUE}
Property:  Acked_Transitions =     {TRUE, TRUE, TRUE}
Property:  Notify_Type =           EVENT
Property:  Event_Time_Stamps =     (((23-MAR-2004, 6:50:21.2),(*-*.*,*:*:*.*), (23-MAR-2004,
                                     6:01:34.0))

```

135-2004b-2. Add a new Global Group object type.

Note

This section was withdrawn after third public review.

135-2004b-3. Add a new Trend Log Multiple object type.

Rationale

There is need for a standard object similar to the Trend Log object type but which can record multiple data items in a single record, align its recording intervals to the clock, and to be able to collect the data items upon command (i.e., when a certain property is written).

Addendum 135-2004b-3

[Add entirely new **Clause 12.Z**. The exact numbering of this clause will be determined when the next edition of the standard is published.]

12.Z Trend Log Multiple Object Type

A Trend Log Multiple object monitors one or more properties of one or more referenced objects, either in the same device as the Trend Log Multiple object or in an external device. When predefined conditions are met, the object saves ("logs") the value of the properties and a timestamp into an internal buffer for subsequent retrieval. The data may be logged periodically or when "triggered" by a write to the Trigger property. Errors that prevent the acquisition of the data, as well as changes in the status or operation of the logging process itself, are also recorded. Each timestamped buffer entry is called a "log record".

The Log_DeviceObjectProperty array holds the list of properties to be monitored and logged. If an element of the Log_DeviceObjectProperty array has an object or device instance number equal to 4194303, this indicates that the element is 'empty' or 'uninitialized'. For empty or uninitialized elements, an indication that no property was specified shall be written to the corresponding entry in each log record.

Each Trend Log Multiple object maintains an internal, optionally fixed-size, buffer in its Log_Buffer property. This buffer fills or grows as log records are added. If the buffer becomes full, the least recent log record is overwritten when a new log record is added, or collection may be set to stop. Trend Log Multiple buffers are transferred as a list of BACnetLogMultipleRecord using the ReadRange service. The buffer may be cleared by writing a zero to the Record_Count property. Each log record in the buffer has an implied SequenceNumber that is equal to the value of the Total_Record_Count property immediately after the log record is added.

Logging may be enabled and disabled through the Enable property and at dates and times specified by the Start_Time and Stop_Time properties. The enabling and disabling of record collection is recorded in the Log_Buffer.

Event reporting (notification) may be provided to facilitate automatic fetching of log records by processes on other devices such as file servers. Mechanisms for both algorithmic and intrinsic reporting are provided.

In intrinsic reporting, when the number of records specified by the Notification_Threshold property has been collected since the previous notification (or startup), a new notification is sent to all subscribed devices.

In response to a notification, subscribers may fetch all of the new log records. If a subscriber needs to fetch all of the new log records, it should use the 'By Sequence Number' form of the ReadRange service request.

A missed notification may be detected by a subscriber if the 'Current Notification' parameter received in the previous BUFFER_READY notification is different than the 'Previous Notification' parameter of the current BUFFER_READY notification. If the ReadRange-ACK response to the ReadRange request issued under these conditions has the FIRST_ITEM bit of the 'Result Flags' parameter set to TRUE, Trend Log Multiple log records have probably been missed by this subscriber.

The acquisition of log records by remote devices has no effect upon the state of the Trend Log Multiple object itself. This allows completely independent, but properly sequential, access to its log records by all remote devices. Any remote device can independently update its records at any time.

Table 12-Z1. Properties of the Trend Log Multiple Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Enable	BOOLEAN	W
Start_Time	BACnetDateTime	O ¹
Stop_Time	BACnetDateTime	O ¹
Log_DeviceObjectProperty	BACnetARRAY[N] of BACnetDeviceObjectPropertyReference	R
Logging_Type	BACnetLoggingType	R
Log_Interval	Unsigned	R ⁴
Align_Intervals	BOOLEAN	O ²
Interval_Offset	Unsigned	O ²
Trigger	BOOLEAN	O
Stop_When_Full	BOOLEAN	R
Buffer_Size	Unsigned32	R
Log_Buffer	List of BACnetLogMultipleRecord	R
Record_Count	Unsigned32	W
Total_Record_Count	Unsigned32	R
Notification_Threshold	Unsigned32	O ³
Records_Since_Notification	Unsigned32	O ³
Last_Notify_Record	Unsigned32	O ³
Notification_Class	Unsigned	O ³
Event_Enable	BACnetEventTransitionBits	O ³
Acked_Transitions	BACnetEventTransitionBits	O ³
Notify_Type	BACnetNotifyType	O ³
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O ³
Profile_Name	CharacterString	O

¹ If present, these properties are required to be writable.

² These properties are required to be present if the object supports clock-aligned logging.

³ These properties are required to be present if the object supports intrinsic reporting.

⁴ This property is required to be writable when Logging_Type has the value POLLED.

12.Z.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet Device that maintains it.

12.Z.2 Object_Name

This property, of type CharacterString, shall represent a name for the Object that is unique within the BACnet Device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.Z.3 Object_Type

This property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be TREND LOG MULTIPLE.

12.Z.4 Description

This optional property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.Z.5 Status_Flags

This property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Trend Log Multiple object. The IN_ALARM and FAULT flags are associated with the values of other properties of this object. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).

FAULT Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

OVERRIDDEN The value of this flag shall be Logical FALSE (0).

OUT_OF_SERVICE The value of this flag shall be Logical FALSE (0).

12.Z.6 Event_State

The Event_State property, of type BACnetEventState, is included in order to provide a way to determine if this object has an active event state associated with it. If the object supports intrinsic reporting, then the Event_State property shall indicate the event state of the object. If the object does not support intrinsic reporting, then the value of this property shall be NORMAL. The Event_State property of this object may have either of the following values:

{NORMAL, FAULT}

12.Z.7 Reliability

This optional property, of type BACnetReliability, provides an indication of whether the application-specific properties of the object or the process executing the application program are "reliable" as far as the BACnet Device can determine. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, CONFIGURATION_ERROR, COMMUNICATION_FAILURE, UNRELIABLE_OTHER}

12.Z.8 Enable

This property, of type BOOLEAN, indicates and controls whether (TRUE) or not (FALSE) logging of events and collected data is enabled. Logging occurs if and only if Enable is TRUE, Local_Time is on or after Start_Time, and Local_Time is before Stop_Time. If Start_Time contains any wildcards, then it shall be considered equal to 'the start of time'. If Stop_Time contains any wildcards, then it shall be considered equal to 'the end of time'. Log records of type log-status or no-data are recorded without regard to the value of the Enable property.

Attempts to write the value TRUE to the Enable property while Stop_When_Full is TRUE and Record_Count is equal to Buffer_Size shall cause a Result(-) response to be issued, specifying an 'Error Class' of OBJECT and an 'Error Code' of LOG_BUFFER_FULL.

12.Z.9 Start_Time

This optional property, of type BACnetDateTime, specifies the date and time at or after which logging shall be enabled by this property. If any of the fields of the BACnetDateTime contain "wildcard" values, then the conditions for logging to be enabled by Start_Time shall be ignored. If Start_Time specifies a date and time after Stop_Time, then logging shall be disabled. This property shall be writable if present.

When Start_Time is reached, the value of the Enable property is not changed.

12.Z.10 Stop_Time

This optional property, of type BACnetDateTime, specifies the date and time at or after which logging shall be disabled by this property. If any of the fields of the BACnetDateTime contain "wildcard" values, then the conditions for logging to be disabled by Stop_Time shall be ignored. If Stop_Time specifies a date and time earlier than Start_Time, then logging shall be disabled. This property shall be writable if present.

When Stop_Time is reached, the value of the Enable property is not changed.

12.Z.11 Log_DeviceObjectProperty

This property, of type BACnetARRAY of BACnetDeviceObjectPropertyReference, specifies the properties to be logged.

If this property is writable, it may be restricted to reference-only objects inside the device containing the Trend Log Multiple object. If the property is restricted to referencing objects within the containing device, an attempt to write a reference to an object outside the containing device into this property shall cause a Result(-) response to be issued, specifying an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

Elements of the Log_DeviceObjectProperty array containing object or device instance numbers equal to 4194303 are considered to be 'empty' or 'uninitialized'. An error shall be written to log record entries corresponding to these empty or uninitialized array elements, specifying an 'Error Class' of PROPERTY and an 'Error Code' of NO_PROPERTY_SPECIFIED.

If this property is changed, one of the following actions shall be taken:

Either the Log_Buffer shall be purged and a BUFFER_PURGED log-status record shall be recorded, or

Log data values corresponding to changed elements of the Log_DeviceObjectProperty array shall be purged by replacing the relevant log data values in each log record with errors, specifying an 'Error Class' of PROPERTY and an 'Error Code' of LOGGED_VALUE_PURGED.

The selection of which action to take is a local matter.

12.Z.12 Logging_Type

This property, of type BACnetLoggingType, specifies whether the Trend Log Multiple collects records using polling or triggered acquisition.

COV Logging is not allowed for a Trend Log Multiple object. If this property is writable, an attempt to write the value COV into this property shall cause a Result(-) response to be issued, specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

If the value POLLED is written to this property when the value of Log_Interval is zero, the Log_Interval property shall be updated with an appropriate default polling interval. Determination of the appropriate default polling interval is a local matter.

If the value TRIGGERED is written to this property when Log_Interval has a non-zero value, the Log_Interval property shall be updated with the value zero.

12.Z.13 Log_Interval

This property, of type Unsigned, specifies the periodic interval in hundredths of seconds for which the referenced properties are to be logged when Logging_Type has the value POLLED. If Logging_Type has the value TRIGGERED, then the value of this property shall be zero and ignored.

This property shall be writable if Logging_Type has the value POLLED, and shall be read-only if Logging_Type has the value TRIGGERED.

12.Z.14 Align_Intervals

This optional property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) clock-aligned periodic logging is enabled. If periodic logging is enabled and the value of Log_Interval is a factor of (that is, it divides without remainder) a second, minute, hour or day, then the beginning of the period specified for logging shall be aligned to the second, minute, hour or day, respectively.

This property is required if the object supports clock-aligned logging. This property has no effect on the behavior of the Trend Log Multiple object if the Logging_Type property has a value other than POLLED.

12.Z.15 Interval_Offset

This optional property, of type Unsigned, specifies the offset in hundredths of seconds from the beginning of the period specified for logging until the actual acquisition of a log record begins. The offset used shall be the value of Interval_Offset modulo the value of Log_Interval; i.e., if Interval_Offset has the value 31 and Log_Interval is 30, the offset used shall be 1. Interval_Offset shall have no effect if Align_Intervals = FALSE.

This property is required if the object supports clock-aligned logging.

12.Z.16 Trigger

This optional property, of type BOOLEAN, shall cause the Trend Log Multiple object to acquire a log record whenever the value of this property is changed from FALSE to TRUE. It shall remain TRUE while the Trend Log Multiple object is acquiring the data items for a log record. When all log data items have been collected or it has been determined that all outstanding data requests will not be fulfilled, the Trend Log Multiple object shall reset the value to FALSE.

If the value of the Logging_Type property is not TRIGGERED and an attempt is made to write the value TRUE to the Trigger property, it is left as a local matter whether to execute the logging operation or not. For devices that choose not to execute triggered logging when Logging_Type is not equal to TRIGGERED, attempts to write the value TRUE to this property when Logging_Type has a value other than TRIGGERED shall cause a Result(-) response to be issued, specifying an 'Error Class' of PROPERTY and an 'Error Code' of NOT_CONFIGURED_FOR_TRIGGERED_LOGGING.

Writing to the Trigger property is not restricted to network visible write operations; internal processes may control the acquisition of samples by writing to this property.

12.Z.17 Stop_When_Full

This property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) logging should cease when the Log_Buffer is full. When logging ceases because the addition of one more log record would cause the buffer to be full, Enable shall be set to FALSE and the event recorded.

If Stop_When_Full is writable, attempts to write the value TRUE to the Stop_When_Full property while Record_Count is equal to Buffer_Size shall result in the oldest log record in the Log_Buffer being discarded, and shall cause the Enable property to be set to FALSE and the event to be recorded.

12.Z.18 Buffer_Size

This property, of type Unsigned32, shall specify the maximum number of records the Log_Buffer can hold. If writable, it may not be written when Enable is TRUE. The disposition of existing log records when Buffer_Size is written is a local matter.

12.Z.19 Log_Buffer

This property is a list of BACnetLogMultipleRecord records. Each log record conveys either a set of recorded data values or errors related to data-collection, a status change in the Trend Log Multiple object, or an indication that the time and/or date was changed in the device hosting the Trend Log Multiple object.

Each log record has data fields as follows:

Timestamp	The local date and time when the record was stored.
LogData	A set of recorded data values or errors related to data-collection, a status change in the Trend Log Multiple object itself, or an indication that the time and/or date was changed in the device hosting the Trend Log Multiple object.

The choices available for LogData are listed below:

log- status	This choice represents a change in the status or operation of the Trend Log Multiple object. Whenever one of the events represented by the flags listed below occurs, a log record shall be appended to the Log_Buffer.
LOG_DISABLED	This flag is changed whenever collection of log records by the Trend Log Multiple object is enabled or disabled. It shall be TRUE if Enable is FALSE, or the local time is outside the range defined by Start_Time and Stop_Time, or the addition of this log record will cause the Log_Buffer to be full and Stop_When_Full is TRUE; otherwise it shall be FALSE.
BUFFER_PURGED	This flag shall be set to TRUE whenever the Log_Buffer is cleared by writing zero to the Record_Count property, or due to a change to the Log_DeviceObjectProperty property. After this value is recorded in the Log_Buffer, the subsequent immediate change to FALSE shall not be recorded. A log record indicating the purging of the Log_Buffer shall be placed into the buffer even if logging is disabled or outside of the time range defined by the Start_Time and Stop_Time properties.
LOG_INTERRUPTED	This flag indicates that the collection of log records by the Trend Log Multiple object was interrupted by a power failure, device reset, object reconfiguration or other such disruption, such that samples prior to this record might have been missed.

log-data	The set of logged values. The order of logged values shall correspond to the order of the Log_DeviceObjectProperty array.	
	boolean-value real-value enum-value unsigned-value signed-value bitstring-value null-value	These choices represent data values read from the monitored object and property.
	any-value	This choice represents data values read from the monitored object and property.
	failure	This choice indicates either that an entry in the Log_DeviceObjectProperty array contains an object or device instance equal to 4194303, that a previously logged value was purged, or that an error was encountered in an attempt to read a data value from the monitored object. If the error is conveyed by an error response from a remote device, the Error Class and Error Code in the response shall be recorded.
time-change	This choice represents a change in the clock setting in the device; it records the number of seconds by which the clock changed. If the number is not known, such as when the clock is initialized for the first time, the value recorded shall be zero.	

Also associated with each log record is an implied record number, the value of which is equal to Total_Record_Count at the point where the log record has been added into the Log_Buffer and Total_Record_Count has been adjusted accordingly. All clients must be able to correctly handle the case where the Log_Buffer is reset such that its Total_Record_Count is returned to zero and also the case where Total_Record_Count has wrapped back to zero.

The Log_Buffer is not network accessible except through the use of the ReadRange service, in order to avoid problems with record sequencing when segmentation is required. Attempts to read this property with the ReadProperty-Request or ReadPropertyMultiple-Request shall cause a Result(-) response to be issued, specifying an 'Error Class' of PROPERTY and an 'Error Code' of READ_ACCESS_DENIED.

12.Z.20 Record_Count

This property, of type Unsigned32, shall represent the number of log records currently resident in the Log_Buffer. A write of the value zero to this property shall cause all log records in the Log_Buffer to be deleted and Records_Since_Notification to be reset to zero. Upon completion, this event shall be reported in the log as the initial entry.

12.Z.21 Total_Record_Count

This property, of type Unsigned32, shall represent the total number of log records collected by the Trend Log Multiple object since creation. When the value of Total_Record_Count reaches its maximum possible value of $2^{32} - 1$, the next value it takes shall be one. Once this value has wrapped to one, its semantic value (the total number of log records collected) has been lost but its use in generating notifications remains.

12.Z.22 Notification_Threshold

This optional property, of type Unsigned32, shall specify the value of Records_Since_Notification at which notification occurs. This property is required if intrinsic reporting is supported by this object.

12.Z.23 Records_Since_Notification

This optional property, of type Unsigned32, represents the number of log records collected since the previous notification, or since the beginning of logging if no previous notification has occurred. This property is required if intrinsic reporting is supported by this object.

12.Z.24 Last_Notify_Record

This property, of type Unsigned32, represents the SequenceNumber associated with the most recently collected log record whose collection caused the value of the Records_Since_Notification property to become equal to or greater than the value of the Notification_Threshold property which triggered a notification. If no notification has occurred since logging began the value of this property shall be zero. This property is required if intrinsic reporting is supported by this object.

12.Z.25 Notification_Class

This optional property, of type Unsigned, shall specify the notification class to be used when handling and generating event notifications for this object. The Notification_Class property implicitly refers to a Notification Class object that has a Notification_Class property with the same value. This property is required if intrinsic reporting is supported by this object.

12.Z.26 Event_Enable

This optional property, of type BACnetEventTransitionBits, shall convey flags that separately enable and disable reporting of TO-FAULT and TO-NORMAL events.. This property is required if intrinsic reporting is supported by this object.

12.Z.26.1 Conditions for Generating a TO-NORMAL Event

A TO-NORMAL event is generated if the TO-NORMAL flag is enabled in the Event_Enable property, and either:

- (a) the value of the Records_Since_Notification property changes from being less than the value of Notification_Threshold to being equal to or greater than the value of Notification_Threshold, or
- (b) the value of the Notification_Threshold property changes from being greater than the value of Records_Since_Notification to being equal to or less than the value of Records_Since_Notification, or
- (c) the Reliability property changes to the value NO_FAULT_DETECTED.

12.Z.26.2 Conditions for Generating a TO-FAULT Event

A TO-FAULT event is generated if the TO-FAULT flag is enabled in the Event_Enable property, and the Reliability property changes to a value other than NO_FAULT_DETECTED.

12.Z.27 Acked_Transitions

This optional property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the receipt of acknowledgments for TO-OFFNORMAL, TO-FAULT and TO-NORMAL events. These flags shall be cleared upon the occurrence of the corresponding event and set under any of these conditions:

- (a) upon receipt of the corresponding acknowledgment;
- (b) upon the occurrence of the event if the corresponding flag is not set in the Event_Enable property (meaning event notifications will not be generated for this condition and thus no acknowledgment is expected);

- (c) upon the occurrence of the event if the corresponding flag is set in the Event_Enable property and the corresponding flag in the Ack_Required property of the Notification Class object implicitly referenced by the Notification_Class property of this object is not set (meaning no acknowledgment is expected).

This property is required if intrinsic reporting is supported by this object.

12.Z.28 Notify_Type

This optional property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. This property is required if intrinsic reporting is supported by this object.

12.Z.29 Event_Time_Stamps

This optional property, of type BACnetARRAY [3] of BACnetTimeStamp, shall convey the times of the last event notifications for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively. Time stamps of type Time or Date shall have 'FF' in each octet and Sequence number time stamps shall have the value 0 if no event notification of that type has been generated since the object was created. This property is required if intrinsic reporting is supported by this object.

12.Z.30 Profile_Name

This optional property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name must begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile document named by the remainder of the profile name. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. The definition of the profiles themselves is outside the scope of this standard.

[Note: Change to **Table 13-2**, p. 256 appears in Addendum 135-2004b-1.]

[Note: Change to **Table 13-3**, p. 257 appears in Addendum 135-2004b-1.]

[Add **BACnetLogData** production in **Clause 21**, p. 419]

```

BACnetLogData ::= CHOICE {
  log-status    [0] BACnetLogStatus,
  log-data      [1] SEQUENCE of CHOICE {
    boolean-value [0] BOOLEAN,
    real-value     [1] REAL,
    enum-value    [2] ENUMERATED, -- Optionally limited to 32 bits
    unsigned-value [3] Unsigned,   -- Optionally limited to 32 bits
    signed-value   [4] INTEGER,    -- Optionally limited to 32 bits
    bitstring-value [5] BIT STRING, -- Optionally limited to 32 bits
    null-value     [6] NULL,
    failure        [7] Error,
    any-value      [8] ABSTRACT-SYNTAX.&Type -- Optional
  }
  time-change [2] REAL
}

```

[Add **BACnetLogMultipleRecord** production in **Clause 21**, p. 419]

```
BACnetLogMultipleRecord ::= SEQUENCE {  
    timestamp    [0] BACnetDateTime,  
    logData      [1] BACnetLogData,  
}
```

[Change **BACnetLogStatus** production, **Clause 21**, p. 419]

```
BACnetLogStatus ::= BITSTRING {  
    log-disabled    (0),  
    buffer-purged  (1) (1),  
    log-interrupted (2)  
}
```

[Add **BACnetLoggingType** production, **Clause 21**, p. 419]

```
BACnetLoggingType ::= ENUMERATED {  
    polled    (0),  
    cov       (1),  
    triggered (2),  
    ...  
}
```

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

[Note: Change to **BACnetObjectType** production in **Clause 21** appears in Addendum 135-2004b-1.]

[Note: Change to **BACnetObjectTypesSupported** production in **Clause 21** appears in Addendum 135-2004b-1.]

[Note: Change to **BACnetPropertyIdentifier** production in **Clause 21** appears in Addendum 135-2004b-1.]

[Change **Clause 22.2.1.4**, Trending, p. 436]

22.2.1.4 Trending

~~"Trending" is the accumulation of (time, value) pairs at specified rates for a specified duration. "Trending" is the accumulation of records consisting of a timestamp and a set of one or more logged data values. These records are collected at specified rates for a specified duration. The values are those of a specific property properties of a specific object. objects. "Trending" is distinguished from the real-time plotting of data in that the data are usually destined for long-term storage and the sampling intervals are usually longer. Interoperability in this area permits the establishment of trending logging parameters and the subsequent retrieval and storage of trend logged data.~~

[Change **Table 23-1**, p, 437]

Table 23-1. Extensible Enumerations

Enumeration Name	Reserved Range	Maximum Value
error-class	0-63	65535
error-code	0-255	65535
BACnetAbortReason	0-63	255
BACnetDeviceStatus	0-63	65535
BACnetEngineeringUnits	0-255	65535
BACnetEventState	0-63	65535
BACnetEventType	0-63	65535
BACnetLifeSafetyMode	0-255	65535
BACnetLifeSafetyState	0-255	65535
BACnetLifeSafetyOperation	0-63	65535
<i>BACnetLoggingType</i>	<i>0-63</i>	<i>255</i>
BACnetMaintenance	0-255	65535
BACnetObjectType	0-127	1023
BACnetProgramError	0-63	65535
BACnetPropertyIdentifier	0-511	4194303
BACnetPropertyStates	0-63	254
BACnetReliability	0-63	65535
BACnetRejectReason	0-63	255
BACnetSilencedState	0-63	65535
BACnetVTClass	0-63	65535

[Add new production to **Annex C**, p. 464]

```

TREND-LOG-MULTIPLE ::= SEQUENCE {
    object-identifier      [75]  BACnetObjectIdentifier,
    object-name            [77]  CharacterString,
    object-type            [79]  BACnetObjectType,
    description            [28]  CharacterString OPTIONAL,
    status-flags           [111] BACnetStatusFlags,
    event-state            [36]  BACnetEventState,
    reliability             [103] BACnetReliability OPTIONAL,
    enable                 [133] BOOLEAN,
    start-time             [142] BACnetDateTime OPTIONAL,
    stop-time              [143] BACnetDateTime OPTIONAL,
    log-device-object-property [132] SEQUENCE OF BACnetDeviceObjectProperty,
                                --accessed as a BACnetARRAY
    logging-type           [197] BACnetLoggingType,
    log-interval           [134] Unsigned,
    align-intervals        [193] BOOLEAN OPTIONAL,
    interval-offset        [195] Unsigned OPTIONAL,
    trigger                [205] BOOLEAN OPTIONAL,
    stop-when-full         [144] BOOLEAN,
    buffer-size            [126] Unsigned32,
    log-buffer             [131] SEQUENCE OF BACnetLogMultipleRecord,
    record-count           [141] Unsigned32,
    total-record-count     [145] Unsigned32,
    notification-threshold [137] Unsigned32 OPTIONAL,
    records-since-notification [140] Unsigned32 OPTIONAL,
    last-notify-record     [173] Unsigned32 OPTIONAL,
    notification-class     [17]  Unsigned OPTIONAL,

```

```

event-enable          [35]  BACnetEventTransitionBits OPTIONAL,
acked-transitions    [0]   BACnetEventTransitionBits OPTIONAL,
notify-type          [72]  BACnetNotifyType OPTIONAL,
event-time-stamps   [130] SEQUENCE OF BACnetTimeStamp OPTIONAL
--accessed as a BACnetARRAY

profile-name         [168] CharacterString OPTIONAL
}

```

[Add new Clause **D.X**, p. 484]

D.X Example of a Trend Log Multiple Object

The following is an example of a Trend Log Multiple object that logs data every 5 minutes from objects in remote device 100 and which performs buffer-ready notification via intrinsic reporting.

```

Property: Object_Identifier =      (Trend Log Multiple, Instance 1)
Property: Object_Name =           "Area 47 Log"
Property: Object_Type =           TREND_LOG_MULTIPLE
Property: Description =           "Area 47 Records"
Property: Status_Flags =         {FALSE, FALSE, FALSE, FALSE}
Property: Event_State =          NORMAL
Property: Reliability =          NO_FAULT_DETECTED
Property: Enable =               TRUE
Property: Log_DeviceObjectProperty = (((Analog Input, Instance 3), Present_Value, (Device, Instance 100)),
                                     ((Analog Input, Instance 3), Status_Flags, (Device, Instance 100)),
                                     ((Binary Output, Instance 5), Present_Value, (Device, Instance 100)))

Property: Logging_Type =         POLLED
Property: Log_Interval =         30000
Property: Align_Intervals =      TRUE
Property: Interval_Offset =      15000
Property: Stop_When_Full =       FALSE
Property: Buffer_Size =           250
Property: Log_Buffer =           (((23-MAR-2004,12:32:33.0),72.0,(FALSE,FALSE,FALSE,FALSE),ON),
                                   ((23-MAR-2004,12:34:32.0),72.1, (FALSE,FALSE,FALSE,FALSE),ON),
                                   ...)

Property: Record_Count =         250
Property: Total_Record_Count =   131040
Property: Notification_Threshold = 83
Property: Records_Since_Notification = 30
Property: Last_Notify_Record =   131010
Property: Notification_Class =   1
Property: Event_Enable =         {FALSE, TRUE, TRUE}
Property: Acked_Transitions =    {TRUE, TRUE, TRUE}
Property: Notify_Type =          EVENT
Property: Event_Time_Stamps =    ((23-MAR-2004, 18:50:21.2),(*-*-*,*:*:*.*), (23-MAR-2004,
                                   18:01:34.0))

```

[Add new Annex K.4.6 through K.4.10, p. 583]

K.4.6 BIBB - Trending-Viewing and Modifying Multiple Values-A (T-VMMV-A)

The A device displays data from a Trend Log Multiple object in the B device and manipulates Trend Log Multiple object collection parameters in the B device.

BACnet Service	Initiate	Execute
ReadRange	x	

K.4.7 BIBB - Trending-Viewing and Modifying Multiple Values Internal-B (T-VMMV-I-B)

The B device collects the multiple-data log records in an internal buffer. Each device claiming conformance to T-VMMV-I-B shall be able to support at least one Trend Log Multiple object.

BACnet Service	Initiate	Execute
ReadRange		x

K.4.8 BIBB - Trending-Viewing and Modifying Multiple Values External-B (T-VMMV-E-B)

The B device is capable of logging multiple properties of multiple objects contained in other devices. The B device shall support T-VMMV-I-B and DS-RPM-A. The Log_Interval and Log_DeviceObjectProperty properties shall be writable.

K.4.9 BIBB - Trending-Automated Multiple Value Retrieval-A (T-AMVR-A)

The A device responds to a notification that a Trend Log Multiple object is ready with new data and acquires the new data from the log.

BACnet Service	Initiate	Execute
ConfirmedEventNotification		x
ReadRange	x	

Devices claiming conformance to T-AMVR-A must be able to process BUFFER_READY event notifications generated by Trend Log Multiple objects and Event Enrollment objects.

K.4.10 BIBB - Trending-Automated Multiple Value Retrieval-B (T-AMVR-B)

The B device notifies the A device that a Trend Log Multiple object's buffer has acquired a predetermined number of data samples using the BUFFER_READY event algorithm either intrinsically in the Trend Log Multiple object or algorithmically using an Event Enrollment object.

BACnet Service	Initiate	Execute
ConfirmedEventNotification	x	
ReadRange		x

Devices claiming conformance to T-AMVR-B shall support the Trend Log Multiple object.

135-2004b-4. Harmonize the Trend Log object with the new Event Log and Trend Log Multiple objects.

Rationale

Several features were added in the Event Log and Trend Log Multiple object types, along with some changes in the language. These features and language are added to the Trend Log object to make it consistent with the other object types.

Addendum 135-2004b-4

[Change **Clause 12.25**, p. 246]

A Trend Log object monitors a property of a referenced object and, when predefined conditions are met, saves ("logs") the value of the property and a timestamp in an internal buffer for subsequent retrieval. The data may be logged periodically ~~or~~, upon a change of value *or when "triggered" by a write to the Trigger property. The Trigger property allows the acquisition of samples to be controlled by network write operations or internal processes.* Errors that prevent the acquisition of the data, as well as changes in the status or operation of the logging process itself, are also recorded. Each timestamped buffer entry is called a trend log "record."

...

Each Trend Log object maintains an internal, optionally fixed-size buffer. This buffer fills or grows as log records are added. If the buffer becomes full, the least recent record is overwritten when a new record is added, or collection may be set to stop. Trend Log records are transferred as BACnetLogRecords using the ReadRange service. The buffer may be cleared by writing a zero to the Record_Count property. Each record in the buffer has an implied SequenceNumber which is equal to the value *of the Total_Record_Count property has immediately after the record is added. If the Total_Record_Count is incremented past $2^{32}-1$, then it shall reset to 1.*

...

Logging may be enabled and disabled through the ~~Log_Enable~~ *Enable* property and at dates and times specified by the Start_Time and Stop_Time properties. Trend Log enabling and disabling is recorded in the log buffer.

...

In intrinsic reporting, when the number of records specified by the Notification_Threshold property has been collected since the previous notification (or startup), a new notification is sent to all subscribed devices. ~~BUFFER_READY algorithmic reporting is described in Clause 13.3.7.~~

...

A missed notification may be detected by a subscriber if the ~~Current_Notify_Record~~ *'Current Notification' parameter* received in the previous *BUFFER_READY* notification is different than the ~~Previous_Notify_Record~~ *'Previous Notification' parameter* of the current *BUFFER_READY* notification. If the ReadRange-ACK response to the ReadRange request issued under these conditions has ~~its~~ *the FIRST_ITEM flag bit of the 'Result Flags' parameter* set to TRUE, Trend Log records have probably been missed by this subscriber.

[Change **Table 12-29**, p.247]

Table 12-29. Properties of the Trend Log Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Log_Enable <i>Enable</i>	BOOLEAN	W
...
Log_Interval	Unsigned	$\Theta^{1-2} O^{1.5}$
...
Profile_Name	CharacterString	O
Logging_Type	BACnetLoggingType	R
Align_Intervals	BOOLEAN	O^4
Interval_Offset	Unsigned	O^4
Trigger	BOOLEAN	O
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	O

¹ These properties are required to be present if the monitored property is a BACnet property.

² If present, these properties are required to be writable.

³ These properties are required to be present if the object supports intrinsic reporting.

⁴ These properties are required to be present if the object supports clock-aligned logging.

⁵ If present, this property is required to be writable when Logging_Type has the value POLLED or the value COV. If present, this property is required to be read-only if Logging_Type has the value TRIGGERED.

[Change **Clause 12.25.4**, p.247]

12.25.4 Description

This *optional* property, of type CharacterString, is a string of printable characters whose content is not restricted.

[Change existing **Clause 12.25.5**, p.248]

12.25.5 ~~Log_Enable~~ *Enable*

This property, of type BOOLEAN, indicates and controls whether (TRUE) or not (FALSE) logging of events and collected data is enabled. ~~A value of FALSE overrides the time interval defined by Start_Time and Stop_Time. If logging is otherwise enabled by the Start_Time and Stop_Time properties, changes to the value of the Log_Enable property shall be recorded in the log. When the device begins operation the value TRUE shall be recorded in the log. Logging occurs if and only if Enable is TRUE, Local_Time is on or after Start_Time, and Local_Time is before Stop_Time. If Start_Time contains any wildcards, then it shall be considered equal to 'the start of time'. If Stop_Time contains any wildcards, then it shall be considered equal to 'the end of time'. Log_Buffer records of type log-status are recorded without regard to the value of the Enable property.~~

If Enable is writable, attempts to write the value TRUE to the Enable property while Stop_When_Full is TRUE and Record_Count is equal to Buffer_Size shall cause a Result(-) response to be issued, specifying an 'Error Class' of OBJECT and an 'Error Code' of LOG_BUFFER_FULL.

[Change existing **Clause 12.25.6**, p.248]

12.25.6 Start_Time

This *optional* property, of type BACnetDateTime, specifies the date and time at or after which logging shall be enabled by this property. If any of the fields of the BACnetDateTime contain "wildcard" values, then the conditions for logging to be enabled by Start_Time shall be ignored. If Start_Time specifies a date and time after Stop_Time, then logging shall be disabled. This property ~~must~~ *shall* be writable if present.

When Start_Time is reached, the value of the Enable property is not changed.

[Change existing **Clause 12.25.7**, p.248]

12.25.7 Stop_Time

This *optional* property, of type BACnetDateTime, specifies the date and time at or after which logging shall be disabled by this property. If any of the fields of the BACnetDateTime contain "wildcard" values, then the conditions for logging to be ~~enabled~~ *disabled* by Stop_Time shall be ignored. If Stop_Time specifies a date and time earlier than Start_Time, then logging shall be disabled. This property ~~must~~ *shall* be writable if present.

When Stop_Time is reached, the value of the Enable property is not changed.

[Change existing **Clause 12.25.8**, p.248]

12.25.8 Log_DeviceObjectProperty

This *optional* property, of type BACnetDeviceObjectPropertyReference, specifies the Device Identifier, Object Identifier and Property Identifier of the property to be trend logged.

...

[Change existing **Clause 12.25.9**, p.248]

12.25.9 Log_Interval

~~This property, of type Unsigned, specifies the periodic interval in hundredths of seconds for which the referenced property is to be logged. If this property has the value zero then the Trend Log shall issue COV subscriptions for the referenced property. The value of this property must be non zero if COV_Resubscription_Interval is not present. This property must be writable if present.~~

This optional property, of type Unsigned, specifies the periodic interval in hundredths of seconds for which the referenced property is to be logged when Logging_Type has the value POLLED. If the Logging_Type property has either of the values COV or TRIGGERED, then the value of the Log_Interval property shall be zero and ignored.

To maintain compatibility with previous versions of the standard, devices supporting COV data collection shall support switching between COV and polled modes in response to writes to Log_Interval. If the Logging_Type property has the value POLLED, changing the Log_Interval property from a non-zero value to the value zero shall change the value of Logging_Type to COV, and shall cause the Trend Log object to issue COV subscriptions for the referenced property. If the Logging_Type property has the value COV, writing a non-zero value to the Log_Interval property shall change the value of Logging_Type to POLLED, and shall cause the Trend Log object to periodically poll the monitored property.

If present, this property shall be writable if Logging_Type has either the value POLLED or the value COV. This property shall be read-only if Logging_Type has the value TRIGGERED.

[Change existing **Clause 12.25.10**, p.248]

12.25.10 COV_Resubscription_Interval

If the Trend Log is acquiring data from a remote device by COV subscription, this *optional* property, of type Unsigned, specifies the number of seconds between COV resubscriptions, provided that COV subscription is in effect. SubscribeCOV requests shall specify twice this lifetime for the subscription and shall specify the issuance of confirmed notifications. If COV subscriptions are in effect, the first COV subscription is issued when the Trend Log object begins operation or when ~~Log_Enable~~ *Enable* becomes TRUE. If present, the value of this property ~~must~~ *shall* be non-zero. If this property is not present, then COV subscription shall not be attempted

[Change existing **Clause 12.25.11**, p.248]

12.25.11 Client_COV_Increment

If the Trend Log is acquiring COV data, this *optional* property, of type BACnetClientCOV, specifies the increment to be used in determining that a change of value has occurred. If the referenced object and property supports COV reporting according to 13.1, this property may have the value NULL; in this case change of value is determined by the criteria of 13.1.

[Change existing **Clause 12.25.12**, p.248]

12.25.12 Stop_When_Full

This property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) logging should cease when the buffer is full. When logging ceases *because the addition of one more record would cause the buffer to be full*, ~~Log_Enable~~ *Enable* shall be set ~~FALSE~~ *to FALSE and the event recorded*.

If Stop_When_Full is writable, attempts to write the value TRUE to the Stop_When_Full property while Record_Count is equal to Buffer_Size shall result in the oldest Log_Buffer record being discarded, and shall cause the Enable property to be set to FALSE and the event to be recorded.

[Change existing **Clause 12.25.13**, p.248]

12.25.13 Buffer_Size

This property, of type Unsigned32, shall specify the maximum number of records the buffer may hold. If writable, it may not be written when ~~Log_Enable~~ *Enable* is TRUE. The disposition of existing records when Buffer_Size is written is a local matter.

[Change existing **Clause 12.25.14**, p.249]

12.25.14 Log_Buffer

...

The choices available for the LogDatum are listed below:

log-status This choice represents a change in the status or operation of the Trend Log object. Whenever one of the events represented by the flags listed below occurs, ~~except as noted,~~ a record shall be appended to the buffer.

~~log-disabled~~
~~LOG_DISABLED~~ ~~This flag is set whenever the Trend Log object is disabled, such as when Log_Enable is set to FALSE. Whenever the Trend Log object begins operation, this flag shall be presumed to have changed from TRUE to FALSE and a log entry shall be made.~~
This flag is changed whenever collection of records by the Trend Log object is enabled or disabled. It shall be TRUE if Enable is FALSE, or the local time is outside the range defined by Start_Time and Stop_Time, or the addition of this record will cause the buffer to be full and Stop_When_Full is TRUE; otherwise it shall be FALSE.

~~buffer purged~~
~~BUFFER_PURGED~~ ~~This flag shall be set to TRUE whenever the buffer is deleted by a write of the value zero to the Record_Count property. This flag shall be set to TRUE whenever the buffer is cleared by writing zero to the Record_Count property or by a change to the Log_DeviceObjectProperty property. After this value is recorded in the buffer, the subsequent immediate change to FALSE shall not be recorded. A record indicating the purging of the buffer shall be placed into the buffer even if the Trend Log is disabled or outside of the time range defined by the Start_Time and Stop_Time properties.~~

LOG_INTERRUPTED *This flag indicates that the collection of records by the Trend Log object was interrupted by a power failure, device reset, object reconfiguration or other such disruption, such that samples prior to this record might have been missed.*

...

Also associated with each record is an implied record number, the value of which is equal to Total_Record_Count at the point where the record has been added into the Log Buffer and Total_Record_Count has been adjusted accordingly. All clients ~~must~~ *shall* be able to correctly handle the case where the Trend Log is reset such that its Total_Record_Count is returned to zero and also the case where Total_Record_Count has wrapped back to ~~1~~ *one*.

The buffer is not network accessible except through the use of the ReadRange service, in order to avoid problems with record sequencing when segmentation is required. *Attempts to read this property with the ReadProperty-Request or ReadPropertyMultiple-Request shall cause a Result(-) response to be issued, specifying an 'Error Class' of PROPERTY and an 'Error Code' of READ_ACCESS_DENIED.*

[Change existing **Clause 12.25.19**]

12.25.19 Last_Notify_Record

This property, of type Unsigned32, represents the SequenceNumber associated with the most recently collected record whose collection *caused the value of the Records_Since_Notification property to become equal to or greater than the value of the Notification_Threshold property that triggered a notification.* If no notification has occurred since logging began the value of this property shall be zero. This property is required if intrinsic reporting is supported by this object.

[Change existing **Clause 12.25.22**, p. 253]

12.25.22 Event Enable

~~This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable reporting of TO FAULT and TO NORMAL events. 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 TO-NORMAL transition. The failure of an attempted COV subscription shall cause a TO FAULT state transition. The TO-NORMAL transition must be enabled when intrinsic reporting is to be used; this shall be set by default. This property is required if intrinsic reporting is supported by this object.~~

This optional property, of type BACnetEventTransitionBits, shall convey flags that separately enable and disable reporting of TO-FAULT and TO-NORMAL events. This property is required if intrinsic reporting is supported by this object.

12.25.22.1 Conditions for Generating a TO-NORMAL Event

A TO-NORMAL event is generated if the TO-NORMAL flag is enabled in the Event_Enable property, and either:

- (a) the value of the Records_Since_Notification property changes from being less than the value of Notification_Threshold to being equal to or greater than the value of Notification_Threshold, or*
- (b) the value of the Notification_Threshold property changes from being greater than the value of Records_Since_Notification to being equal to or less than the value of Records_Since_Notification, or*
- (c) the Reliability property changes to the value NO_FAULT_DETECTED.*

12.25.22.2 Conditions for Generating a TO-FAULT Event

A TO-FAULT event is generated if the TO-FAULT flag is enabled in the Event_Enable property and the Reliability property changes to a value other than NO_FAULT_DETECTED.

[Add new **Clause 12.25.A1**, p. 251]

12.25.A1 Logging_Type

This property, of type BACnetLoggingType, specifies whether the Trend Log collects records using polling, COV or triggered acquisition.

If this property is writable, an attempt to write a value not supported by the object into this property shall cause a Result(-) response to be issued, specifying an 'Error Class' of PROPERTY and 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

If this property has the value COV, then the Trend Log shall issue COV subscriptions for the referenced property, and shall log the COV notifications if they indicate a changed value.

If this property has the value POLLED, then the Trend Log shall periodically poll the monitored property on the interval defined by the Log_Interval, Align_Intervals, and Interval_Offset properties.

If the value POLLED is written to this property when the value of Log_Interval is zero, then the Log_Interval property shall be updated with an appropriate default polling interval. Determination of the appropriate default polling interval is a local matter.

If either of the values COV or TRIGGERED is written to this property when Log_Interval has a non-zero value, then the Log_Interval property shall be updated with the value zero.

[Add new **Clause 12.25.A2**, p. 251]

12.25.A2 Align_Intervals

This optional property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) clock-aligned periodic logging is enabled. If clock-aligned periodic logging is enabled and the value of Log_Interval is a factor of (that is, divides without remainder) a second, minute, hour or day, then the beginning of the period specified for logging shall be aligned to the second, minute, hour or day, respectively.

This property is required if the object supports clock-aligned logging. This property has no effect on the behavior of the Trend Log object if the Logging_Type property has a value other than POLLED.

[Add new **Clause 12.25.A3**, p. 251]

12.25.A3 Interval_Offset

This optional property, of type Unsigned, specifies the offset in hundredths of seconds from the beginning of the period specified for logging until the actual acquisition of a log record begins. The offset used shall be the value of Interval_Offset modulo the value of Log_Interval; i.e., if Interval_Offset has the value 31 and Log_Interval is 30, the offset used shall be 1. Interval_Offset shall have no effect if Align_Intervals = FALSE.

This property is required if the object supports clock-aligned logging.

[Add new **Clause 12.25.A4**, p.254]

12.25.A4 Trigger

This optional property, of type BOOLEAN, shall cause the Trend Log object to acquire a log record whenever the value of this property is changed from FALSE to TRUE. It shall remain TRUE while the Trend Log object is acquiring the data items for a record. When all data items have been collected or it has been determined that all outstanding data requests will not be fulfilled, the Trend Log object shall reset the value to FALSE.

If the value of the Logging_Type property is not TRIGGERED and an attempt is made to write the value TRUE to the Trigger property, it is left as a local matter whether to execute the logging operation or not. For devices that choose not to execute triggered logging when Logging_Type is not equal to TRIGGERED, attempts to write the value TRUE to this property when Logging_Type has a value other than TRIGGERED shall cause a Result(-) response to be issued, specifying an 'Error Class' of PROPERTY and an 'Error Code' of NOT_CONFIGURED_FOR_TRIGGERED_LOGGING.

Writing to the Trigger property is not restricted to network-visible write operations; internal processes may control the acquisition of samples by writing to this property.

[Add new **Clause 12.25.A5**, p. 251]

12.25.A5 Status_Flags

This property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Trend Log object. The IN_ALARM and FAULT flags are associated with the values of other properties of this object. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).

FAULT Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

OVERRIDDEN The value of this flag shall be Logical FALSE (0).

OUT_OF_SERVICE The value of this flag shall be Logical FALSE (0).

[Add new **Clause 12.25.A6**, p. 251]

12.25.A6 Reliability

This optional property, of type BACnetReliability, provides an indication of whether the application-specific properties of the object or the process executing the application program are "reliable" as far as the BACnet Device can determine. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, CONFIGURATION_ERROR, COMMUNICATION_FAILURE, UNRELIABLE_OTHER}.

[Change **Clause 13.3.7**, p.264]

13.3.7 BUFFER_READY Algorithm

A BUFFER_READY occurs when the number of records specified by Notification_Threshold has been entered into the log since the start of operation or the previous notification, whichever is most recent. The number of records collected is determined by the formula $Total_Record_Count - Previous_Notification_Count$ if $Total_Record_Count$ is greater than or equal to $Previous_Notification_Count$; otherwise it is determined by the formula $Total_Record_Count - Previous_Notification_Count + 2^{32} - 2^{32} - 1$. Upon completion of the notification, $Previous_Record_Count$ is set to the value of $Total_Record_Count$ that caused the notifications to occur.

[Note: Addition of BACnetLoggingType production, **Clause 21**, appears in Addendum 135-2004b-3.]

[Note: Addition of BACnetLoggingType to **Table 23-1**, appears in Addendum 135-2004b-3.]

[Add new **Clauses 18.3.6** through **18.3.9**, and renumber subsequent clauses, p.355]

18.3.6 LOG_BUFFER_FULL – The attempted operation would result in the addition of a log record to an object whose log buffer is full.

18.3.7 LOGGED_VALUE_PURGED – A previously logged value was purged due to a change to the list of logged properties.

18.3.8 NO_PROPERTY_SPECIFIED – No data was logged due to a device or object instance equal to 4194303 in the list of logged properties.

18.3.9 NOT_CONFIGURED_FOR_TRIGGERED_LOGGING – The attempted logging operation is only allowed when the Logging_Type property has the value TRIGGERED.

[Add in alphabetic order to Clause 21, Error production, pp.406-407]

[Note: enumerations 50 to 72 appear in Addendum 135-2004b-11.]

[Note: enumerations 73 and 74 appear in Addendum 135-2004d-11.]

```
Error ::= SEQUENCE {
    ...
    error-code    ENUMERATED {
```

```

...
key-generation-error          (15),
log-buffer-full                (75),
logged-value-purged           (76),
no-property-specified         (77),
not-configured-for-triggered-logging (78),
missing-required-parameter    (16),
...
-- see log-buffer-full        (75)
-- see logged-value-purged    (76),
-- see no-property-specified  (77),
-- see not-configured-for-triggered-logging (78)
}

```

[Change **D.25**, p.483-484, by replacing Log_Enable with Enable and adding new properties]

D.25 Example of a Trend Log Object

The following is an example of a Trend Log object that periodically logs data from an object in a remote device and which performs buffer-ready notification via intrinsic reporting.

```

Property: Object_Identifier = (Trend Log, Instance 1)
Property: Object_Name = "Room 3Log"
Property: Object_Type = TREND_LOG
Property: Description = "Room 3 Temperature"
Property: Log_Enable Enable = TRUE
Property: Log_Interval = 6,000
Property: Stop_When_Full = FALSE
Property: Buffer_Size = 250
Property: Log_Buffer = (((23-MAR-1998,12:32:33.0), 72.0,(FALSE,FALSE,FALSE,FALSE)),
(23-MAR-1998,12:34:32.0),72.1,(FALSE,FALSE,FALSE,FALSE)),
...)
Property: Record_Count = 250
...
Property: Event_Time_Stamps= ((23-MAR-95, 18:50:21.2),
(*-*.*,*:*:*.*),
(23-MAR-95, 19:01:34.0))
Property: Logging_Type = POLLED
Property: Align_Intervals = FALSE
Property: Interval_Offset = 0
Property: Trigger = FALSE
Property: Status_Flags = {FALSE, FALSE, FALSE, FALSE}
Property: Reliability = NO_FAULT_DETECTED

```

[Change **Annex C**, p.464, by replacing log-enable with enable and changing production]

```

TREND-LOG ::= SEQUENCE {
  object-identifier [75] BACnetObjectIdentifier,
  object-name [77] CharacterString,
  object-type [79] BACnetObjectType,
  description [28] CharacterString OPTIONAL,
  log-enable enable [133] BOOLEAN,
  ...
  event-time-stamps [130] SEQUENCE OF BACnetTimeStamp OPTIONAL,
  --accessed as a BACnetARRAY

```

<i>profile-name</i>	[168]	CharacterString OPTIONAL
<i>logging-type</i>	[197]	<i>BACnetLoggingType</i> ,
<i>align-intervals</i>	[193]	BOOLEAN OPTIONAL,
<i>interval-offset</i>	[195]	Unsigned OPTIONAL,
<i>trigger</i>	[205]	BOOLEAN OPTIONAL,
<i>status-flags</i>	[111]	<i>BACnetStatusFlags</i> ,
<i>reliability</i>	[103]	<i>BACnetReliability</i> OPTIONAL,
}		

135-2004b-5. Define a means for a device to provide a notification that it has restarted.

Rationale

When a BACnet device restarts, it could lose some of its configuration and subscriptions. Other devices may depend on this configuration or subscription information for change of value notifications or other purposes. This new restart procedure provides a means to notify peer devices that a restart has occurred, enabling them to take appropriate action.

Addendum 135-2004b-5

[Add new **Clause 19.N**, p. 365]

19.N Device Restart Procedure

When a BACnet device restarts, there are a number of different configuration items that can be lost. For example, a device need not remember which devices have subscribed to receive change-of-value notifications or to which values they have subscribed. For this reason, other devices may be interested in determining when a device has restarted. This section outlines how a device may interoperably indicate that it has restarted.

When a device is powered on, when it restarts due to a ReinitializeDevice service (COLDSTART or WARMSTART), or when it restarts for some other reason, the device shall transmit an UnconfirmedCOVNotification request. The 'Subscriber Process Identifier' parameter shall be 0, the 'Monitored Object Identifier' parameter shall reference the Device object, the 'Time Remaining' parameter shall be 0, and the 'List of Values' parameter shall contain three values, the System_Status, the Time_Of_Device_Restart, and the Last_Restart_Reason properties of the Device object. The device shall transmit this message after the complete power-up or restart sequence has been completed so that the system-status value is accurate.

The device shall send the restart notification to each recipient in the Restart_Notification_Recipients property of the Device object.

MS/TP slave devices are not able to support this procedure, although they may support the Time_Of_Device_Restart and Last_Restart_Reason properties.

[Change **Table 12-12**, p.178-179]

Table 12-12. Properties of the Device Object Type

Property Identifier	Property Datatype	Conformance Code
...		
Profile_Name	CharacterString	O
Last_Restart_Reason	BACnetRestartReason	O ¹³
Time_Of_Device_Restart	BACnetTimeStamp	O ¹³
Restart_Notification_Recipients	List of BACnetRecipient	O ¹³

¹³ These properties are required if the device supports the restart procedure as described in Clause 19.N.

[Add new **Clause 12.11.X1**, p. 187]

12.11.X1 Last_Restart_Reason

This property, of type BACnetRestartReason, indicates the reason for the last device restart. This property shall be present if the device supports the BACnet restart procedure as described in Clause 19.N. The possible values for this property are:

UNKNOWN	The device cannot determine the cause of the last reset.
COLDSTART	A ReinitializeDevice request was received with a 'Reinitialized State of Device' of COLDSTART or the device was made to COLDSTART by some other means.
WARMSTART	A ReinitializeDevice request was received with a 'Reinitialized State of Device' of WARMSTART or the device was made to WARMSTART by some other means.
DETECTED_POWER_LOST	The device detected that incoming power was lost.
DETECTED_POWERED_OFF	The device detected that its power switch was turned off.
HARDWARE_WATCHDOG	The hardware watchdog timer reset the device.
SOFTWARE_WATCHDOG	The software watchdog timer reset the device.
SUSPENDED	The device was suspended. How the device was suspended or what it means to be suspended is a local matter.

[Add new **Clause 12.11.X2**, p. 187]

12.11.X2 Time_Of_Device_Restart

This property, of type BACnetTimeStamp, is the time at which the device was last restarted. This property shall be present if the device supports the BACnet restart procedure as described in Clause 19.N.

[Add new **Clause 12.11.X3**, p. 187]

12.11.X3 Restart_Notification_Recipients

The Restart_Notification_Recipients property is used to control the restrictions on which devices, if any, are to be notified when a restart occurs. The value of this property shall be a list of zero or more BACnetRecipients. If the list is of length zero, a device is prohibited from sending a device restart notification. The default value of the property shall be a single entry representing a broadcast on the local network. If the property is not writable, then it shall contain the default value. If the list is of length one or more, a device shall send a restart notification, but only to the devices or addresses listed. This property shall be present if and only if the device supports the BACnet restart procedure as described in Clause 19.N.

[Change production **BACnetPropertyStates**, **Clause 21**, p. 428]

```

BACnetPropertyStates ::= CHOICE {
-- This production represents the possible datatypes for properties that
-- have discrete or enumerated values. The choice must be consistent with the
-- datatype of the property referenced in the Event Enrollment Object.

    boolean-value      [0] BOOLEAN,
    ...
    life-safety-state  [13] BACnetLifeSafetyState,
    restart-reason     [14] BACnetRestartReason,
    ...

```

[Add to **Clause 21**, new production **BACnetRestartReason**, p. 429]

```

BACnetRestartReason ::= ENUMERATED {
    unknown           (0),
    coldstart         (1),
    warmstart         (2),
    detected-power-lost (3),
    detected-powered-off (4),
    hardware-watchdog (5),
    software-watchdog (6),
    suspended         (7),
    ...

```

}
 -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values 64-254 255
 -- may be used by others subject to the procedures and constraints described in Clause 23.

[Change **Table 23-1**, p. 437]

Table 23-1. Extensible Enumerations

Enumeration Name	Reserved Range	Maximum Value
...		
BACnetVTClass	0-63	65535
BACnetRestartReason	0-63	255

[Change **Annex C**, DEVICE object type description, p. 457-458]

```

DEVICE ::= SEQUENCE {
  ...
  last-restart-reason      [196]  BACnetRestartReason OPTIONAL,
  restart-notification-recipients [202] SEQUENCE OF BACnetRecipient OPTIONAL,
  time-of-device-restart   [203]  BACnetTimeStamp OPTIONAL,
  profile-name             [168]  CharacterString OPTIONAL
}

```

[Change **D.11**, Example 1, p. 471-472]

```

Property:  Active_COV_Subscriptions = (((0, (Device, Instance 12)), 300),
                                       ((Analog Input, 1), Present_Value), TRUE, 100, 1.0),
                                       ((0, (Device, Instance 40)), 600),
                                       ((Analog Input, 1), Present_Value), TRUE, 3, 1.5))
Property:  Last_Restart_Reason = DETECTED_POWERED_OFF
Property:  Restart_Notification_Recipients = ((0,X'FF')) -- This example used an MS/TP broadcast address.
Property:  Time_Of_Device_Restart = (02-SEP-2003, 12:34:56.78)

```

[Change **D.11**, Example 2, p. 472-473]

```

Property:  Database_Revision = 69
Property:  Last_Restart_Reason = DETECTED_POWERED_OFF
Property:  Restart_Notification_Recipients = ((Device, Instance 18))
Property:  Time_Of_Device_Restart = (04-OCT-2002, 02:04:06.08)

```

[Change **K.5.20**, p. 587]

K.5.20 BIBB - Device Management-Restart-B (DM-R-B)

The B device informs the A device(s) each time it restarts.

BACnet Service	Initiate	Execute
UnconfirmedCOVNotification	x	

Devices claiming conformance to DM-R-B shall support the ~~Time_Of_Device_Restart and Last_Restart_Reason~~
 Last_Restart_Reason, Restart_Notification_Recipients, and Time_Of_Device_Restart properties of the Device object.

135-2004b-6. Define a means to configure a device to periodically send time synchronization messages.

Rationale

There is need for an interoperable means for configuring a device to periodically send TimeSynchronization and UTCTimeSynchronization messages.

Addendum 135-2004b-6

[Change Table 12-13, p. 178, including inserting footnote 5 and renumbering subsequent footnotes.]

Table 12-13. Properties of the Device Object Type

Property Identifier	Property Datatype	Conformance Code
...
Profile_Name	CharacterString	O
...
UTC_Time_Synchronization_Recipients	List of BACnetRecipient	O ⁵
Time_Synchronization_Interval	Unsigned	O ¹³
Align_Intervals	BOOLEAN	O ¹³
Interval_Offset	Unsigned	O ¹³
...

⁵ Required if PICS indicates that this device is a Time Master. If this property is present, then Time_Synchronization_Interval, Align_Intervals and Interval_Offset shall be present. If present, this property shall be writable.

¹³ If either Time_Synchronization_Recipients or UTC_Time_Synchronization_Recipients is present, then this property shall be present and writable.

...

[Change Clause 12.11.30, p. 182]

12.11.30 Time_Synchronization_Recipients

~~The Time_Synchronization_Recipients property. This optional property, of type List of BACnetRecipient, is used to control the restrictions placed on a device's use of the TimeSynchronization service. The value of this property shall be a list of zero or more BACnetRecipients. If the list is of length zero, or the property is not present, the device is prohibited from automatically sending a TimeSynchronization request. If the list is of length one or more, the device may automatically send a TimeSynchronization request but only to the devices or addresses listed. If this property is present, it shall be writable. If the PICS indicates that this device is a Time Master, then the Time_Synchronization_Recipients property shall be present.~~

[Add new Clause 12.11.Y1, p. 187]

12.11.Y1 UTC_Time_Synchronization_Recipients

This optional property, of type List of BACnetRecipient, is used to control the restrictions placed on a device's use of the UTCTimeSynchronization service. The value of this property shall be a list of zero or more BACnetRecipients. If the list is of length zero, or the property is not present, the device is prohibited from automatically sending a UTCTimeSynchronization request. If the list is of length one or more, the device may automatically send a UTCTimeSynchronization request but only to the devices or addresses listed. If this property is present, it shall be writable.

[Add new **Clause 12.11.Y2**, p. 187]

12.11.Y2 Time_Synchronization_Interval

This optional property, of type Unsigned, specifies the periodic interval in minutes at which TimeSynchronization and UTCTimeSynchronization requests shall be sent. If this property has a value of zero, then periodic time synchronization is disabled. If this property is present, it shall be writable.

[Add new **Clause 12.11.Y3**, p. 187]

12.11.Y3 Align_Intervals

This optional property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) clock-aligned periodic time synchronization is enabled. If periodic time synchronization is enabled and the time synchronization interval is a factor of (divides without remainder) an hour or day, then the beginning of the period specified for time synchronization shall be aligned to the hour or day, respectively. If this property is present, it shall be writable.

[Add new **Clause 12.11.Y4**, p. 187]

12.10.Y4 Interval_Offset

This optional property, of type Unsigned, specifies the offset in minutes from the beginning of the period specified for time synchronization until the actual time synchronization requests are sent. The offset used shall be the value of Interval_Offset modulo the value of Time_Synchronization_Interval; e.g., if Interval_Offset has the value 31 and Time_Synchronization_Interval is 30, the offset used shall be 1. Interval_Offset shall have no effect if Align_Intervals is FALSE. If this property is present, it shall be writable.

[Change **Annex C**, DEVICE description, p.457-458]

```
DEVICE ::= {  
    ...  
    max-segments-accepted           [167] Unsigned,  
    utc-time-synchronization-recipients [206]SEQUENCE OF BACnetRecipient OPTIONAL,  
    time-synchronization-interval    [204]Unsigned OPTIONAL,  
    align-intervals                  [193]BOOLEAN OPTIONAL,  
    interval-offset                  [195]Unsigned OPTIONAL,  
    profile-name                     [168] CharacterString OPTIONAL  
}
```

135-2004b-7. Extend the number of character sets supported.

Note

This section was withdrawn after first public review.

135-2004b-8. Enable devices other than alarm recipients to acknowledge alarms.

Rationale

Devices that are not in the list of alarm recipients are currently unable to acknowledge alarms. This prevents workstations that have learned about an alarm through other means from acknowledging it.

Addendum 135-2004b-8

[Change **Clause 13.5.1.2**, p. 269]

13.5.1.2 Acknowledging Process Identifier

This parameter, of type Unsigned32, shall specify the 'Process Identifier' parameter ~~from the event notification to which this acknowledgment is a response. This allows the initiating object to ensure that the desired process has received the notification.~~ *that identifies the acknowledging process. The assignment of acknowledging process identifiers is a local matter.*

135-2004b-9. Allow MS/TP BACnet Data Expecting Reply frames to be broadcast.

Rationale

The network layer allows a device to broadcast on its local LAN a message to be routed to a device on some other network (see Clause 6.5.3), but the MS/TP Master Node state machine does not permit an MS/TP router to receive such a message. This addendum changes the state machine so that the MS/TP router will receive and process broadcast BACnet Data Expecting Reply frames.

Addendum 135-2004b-9

[Change **Clause 9.5.6.2**, p.87-88]

9.5.6.2 IDLE

...

ReceivedUnwantedFrame

If ReceivedValidFrame is TRUE and either

- a) DestinationAddress is not equal to either TS (this station) or 255 (broadcast), or
- b) DestinationAddress is equal to 255 (broadcast) and FrameType has a value of Token, ~~BACnet Data Expecting Reply~~, Test_Request, or a proprietary type known to this node that expects a reply (such frames may not be broadcast), or
- c) FrameType has a value that indicates a standard or proprietary type that is not known to this node,

then an unexpected or unwanted frame was received. Set ReceivedValidFrame to FALSE, and enter the IDLE state to wait for the next frame.

...

ReceivedDataNeedingReply

If ReceivedValidFrame is TRUE and DestinationAddress is equal to TS (this station) and FrameType is equal to BACnet Data Expecting Reply, Test Request, or a proprietary type known to this node that expects a reply,

then indicate successful reception to the higher layers (management entity in the case of Test_Request); set ReceivedValidFrame to FALSE; and enter the ANSWER_DATA_REQUEST state.

BroadcastDataNeedingReply

If ReceivedValidFrame is TRUE and DestinationAddress is equal to 255 (broadcast) and FrameType is equal to BACnet Data Expecting Reply,

then indicate successful reception to the higher layers; set ReceivedValidFrame to FALSE; and enter the IDLE state to wait for the next frame.

[Change **Clause 9.5.6.3**, p.88]

9.5.6.3 USE_TOKEN

...

SendNoWait

If there is a frame awaiting transmission that is of type Test_Response, BACnet Data Not Expecting Reply, a proprietary type that does not expect a reply, *or a frame of type Data Expecting Reply with a DestinationAddress that is equal to 255 (broadcast)*,

then call SendFrame to transmit the data frame; increment FrameCount; and enter the DONE_WITH_TOKEN state.

SendAndWait

If there is a frame awaiting transmission that is of type `Test_Request`, a proprietary type that expects a reply, *or a frame of type `Data Expecting Reply` with a `DestinationAddress` that is not equal to 255 (broadcast)*,

then call `SendFrame` to transmit the data frame; increment `FrameCount`; and enter the `WAIT_FOR_REPLY` state.

135-2004b-10. Revise the Clause 5 state machines to handle slow servers.

Note

This section was withdrawn after second public review.

135-2004b-11. Add new Error Codes and specify usage.

Rationale

A comprehensive set of reviews has shown the need for additional error classes and codes to accurately convey the error situation being reported.

Addendum 135-2004b-11

[Change Clause 14.1.4.1, p.295]

14.1.4.1 Error Type

This parameter consists of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18. *The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:*

<u>Situation:</u>	<u>Error Class:</u>	<u>Error Code:</u>
<i>The File object does not exist</i>	OBJECT	UNKNOWN_OBJECT
<i>'File Start Record' is out of range</i>	SERVICES	INVALID_FILE_START_POSITION
<i>Incorrect File access method</i>	SERVICES	INVALID_FILE_ACCESS_METHOD

[Change Clause 14.2.4.1, p.298]

14.2.4.1 Error Type

This parameter consists of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18. *The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:*

<u>Situation:</u>	<u>Error Class:</u>	<u>Error Code:</u>
<i>The File object does not exist</i>	OBJECT	UNKNOWN_OBJECT
<i>'File Start Record' is out of range</i>	SERVICES	INVALID_FILE_START_POSITION
<i>Incorrect File access method</i>	SERVICES	INVALID_FILE_ACCESS_METHOD
<i>Write to a read-only File</i>	SERVICES	FILE_ACCESS_DENIED

[Change Clause 15.1.1.3.1, p. 300]

15.1.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18. *The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:*

<u>Situation:</u>	<u>Error Class:</u>	<u>Error Code:</u>
<i>Specified object does not exist</i>	OBJECT	UNKNOWN_OBJECT
<i>Specified property does not exist</i>	PROPERTY	UNKNOWN_PROPERTY
<i>The element datatype does not match the property</i>	PROPERTY	INVALID_DATATYPE
<i>The data being written has a datatype not supported by the property.</i>	PROPERTY	DATATYPE_NOT_SUPPORTED
<i>The element value is out of range for the property</i>	PROPERTY	VALUE_OUT_OF_RANGE
<i>The specified property is currently not modifiable by the requestor</i>	PROPERTY	WRITE_ACCESS_DENIED
<i>There is not enough free memory for the element</i>	RESOURCES	NO_SPACE_TO_ADD_LIST_ELEMENT
<i>The property or specified array element is not a list</i>	SERVICES	PROPERTY_IS_NOT_A_LIST
<i>An array index is provided but the property is not an array</i>	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
<i>An array index is provided that is outside the range existing in the property</i>	PROPERTY	INVALID_ARRAY_INDEX

[Change **Clause 15.2.1.3.1**, p. 301]

15.2.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18. *The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:*

<u>Situation:</u>	<u>Error Class:</u>	<u>Error Code:</u>
<i>Specified object does not exist</i>	OBJECT	UNKNOWN_OBJECT
<i>Specified property does not exist</i>	PROPERTY	UNKNOWN_PROPERTY
<i>The element datatype does not match the property</i>	PROPERTY	INVALID_DATATYPE
<i>The specified property is currently not modifiable by the requestor</i>	PROPERTY	WRITE_ACCESS_DENIED
<i>A list element to be removed is not present</i>	SERVICES	LIST_ELEMENT_NOT_FOUND
<i>The property or specified array element is not a list</i>	SERVICES	PROPERTY_IS_NOT_A_LIST
<i>An array index is provided but the property is not an array</i>	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
<i>An array index is provided that is outside the range existing in the property</i>	PROPERTY	INVALID_ARRAY_INDEX

[Append to **Clause 15.3.2.1**, p. 304]

15.3.2.1 Error Class and Error Code Assignments

<u>Situation</u>	<u>Error Class</u>	<u>Error Code</u>
...
<i>The data being written has a datatype not supported by the property.</i>	PROPERTY	DATATYPE_NOT_SUPPORTED

[Append to **Clause 15.9.2.1**, p. 321]

15.9.2.1 Error Class and Error Code Assignments

<u>Situation</u>	<u>Error Class</u>	<u>Error Code</u>
...
<i>The data being written has a datatype not supported by the property.</i>	PROPERTY	DATATYPE_NOT_SUPPORTED

[Append to **Clause 15.10.2.1**, p. 323]

15.10.2.1 Error Class and Error Code Assignments

<u>Situation</u>	<u>Error Class</u>	<u>Error Code</u>
...
<i>The data being written has a datatype not supported by the property.</i>	PROPERTY	DATATYPE_NOT_SUPPORTED

[Change **Clause 16.1.2**, p.326]

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 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 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~~ *an Error-PDU with 'error class' = SECURITY and 'error code' = PASSWORD_FAILURE* shall be issued.

[Change **Clause 16.4.2**, p.330-331]

16.4.2 Service Procedure

After verifying the validity of the request, including the password, the responding BACnet-user shall pre-empt all other outstanding requests and respond with a 'Result(+)' primitive. If the request is WARMSTART or COLDSTART the responding BACnet-user will immediately proceed to perform any applicable shut-down procedures prior to reinitializing the device as specified by the requesting BACnet-user in the request. If the service request is for WARMSTART and the device is not ready due to its initial characterization being in progress, a 'Result (-)' response primitive shall be issued.

If the requested state is one of STARTBACKUP, ENDBACKUP, STARTRESTORE, ENDRESTORE, or ABORTRESTORE, then the device shall behave as described in 19.1.

If the password is invalid or is absent when one is required, a ~~'Result (-)' response primitive~~ *an Error-PDU with 'error class' = SECURITY and 'error code' = PASSWORD_FAILURE* shall be issued.

[Change **Clause 18.5**, Error Class - SECURITY, p.355]

18.5 Error Class - SECURITY

This Error Class pertains to problems related to ~~the execution of security services.~~ *security.* ~~Without exception, these errors signal the inability of the responding BACnet user to carry out the desired service in its entirety and are thus "fatal."~~

[Change **Clause 18.5.6**, p.356]

18.5.6 PASSWORD_FAILURE - ~~The 'Operator Name' and 'Operator Password' did not associate correctly. The password was incorrect.~~

[Insert new **Clause 18.7**, p.357, and renumber original **Clause 18.7** and subsequent clauses]

18.7 Error Class – COMMUNICATION

This Error Class pertains to problems related to network communications. These codes indicate problems reported by a remote device in abort and reject PDUs, or they indicate problems detected internally. These error codes are stored in properties of objects whose operation involves the network communications, such as the Trend Log object's Log_Buffer property. This Error Class shall not be conveyed in error PDUs.

18.7.1 ABORT_BUFFER_OVERFLOW - An input buffer capacity has been exceeded in this device or was reported by the remote device.

18.7.2 ABORT_INVALID_APDU_IN_THIS_STATE - An APDU was received, by this device or the remote device, that was not expected in the present state of the Transaction State Machine.

18.7.3 ABORT_PREEMPTED_BY_HIGHER_PRIORITY_TASK - The transaction was aborted to permit higher priority processing by this device or the remote device.

18.7.4 ABORT_SEGMENTATION_NOT_SUPPORTED – An abort PDU specifying an abort code of SEGMENTATION_NOT_SUPPORTED was sent or received by this device.

18.7.5 ABORT_PROPRIETARY – An abort PDU with a proprietary reason was sent or received by this device.

18.7.6 ABORT_OTHER - This device sent or received an abort PDU with a reason of OTHER.

18.7.7 REJECT_BUFFER_OVERFLOW - An input buffer capacity has been exceeded in this device or has been reported by the remote device.

18.7.8 REJECT_INCONSISTENT_PARAMETERS – The remote device sent a reject PDU with a reason of INCONSISTENT_PARAMETERS.

18.7.9 REJECT_INVALID_PARAMETER_DATA_TYPE - The remote device sent a reject PDU with a reason of INVALID_PARAMETER_DATATYPE.

18.7.10 REJECT_INVALID_TAG - This device or the remote device encountered an invalid tag while parsing a message.

18.7.11 REJECT_MISSING_REQUIRED_PARAMETER - The remote device sent a reject PDU with a reason of MISSING_REQUIRED_PARAMETER.

18.7.12 REJECT_PARAMETER_OUT_OF_RANGE - The remote device sent a reject PDU with a reason of PARAMETER_OUT_OF_RANGE.

18.7.13 REJECT_TOO_MANY_ARGUMENTS - The remote device sent a reject PDU with a reason of TOO_MANY_ARGUMENTS.

18.7.14 REJECT_UNDEFINED_ENUMERATION - The remote device sent a reject PDU with a reason of UNDEFINED_ENUMERATION.

18.7.15 REJECT_UNRECOGNIZED_SERVICE - The remote device sent a reject PDU with a reason of UNRECOGNIZED_SERVICE.

18.7.16 REJECT_PROPRIETARY – This reject reason indicates that a proprietary reject reason was sent or received by this device.

18.7.17 REJECT_OTHER - The remote device sent a reject PDU with a reason of OTHER.

18.7.18 INVALID_TAG – This error indicates that an improper tag was found when parsing the response to a confirmed service request or an unconfirmed service request.

18.7.19 NETWORK_DOWN – This error indicates that the local network connection was not established when the request was initiated.

18.7.20 TIMEOUT – This error indicates that a request timed out before a response was received from the remote device.

18.7.21 UNKNOWN_DEVICE – This error indicates that a request was not initiated because the remote device could not be found.

18.7.22 UNKNOWN_ROUTE – This error indicates that a request was not initiated because a route to the network where the remote device resides could not be found.

18.7.23 OTHER – This error indicates that a communication error occurred other than those previously enumerated for this Error Class.

[Change the Error production, **Clause 21**, p. 406-407.]

Error ::= SEQUENCE {

-- NOTE: The valid combinations of error-class and error-code are defined in Clause 18.

```
error-class  ENUMERATED {
    device      (0),
    object      (1),
    property    (2),
    resources   (3),
    security    (4),
    services    (5),
    vt          (6),
    communication (7),
    ...
},
```

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

```
error-code  ENUMERATED {
    other (0),
    abort-buffer-overflow (51),
    abort-invalid-apdu-in-this-state (52),
    abort-preempted-by-higher-priority-task (53),
    abort-segmentation-not-supported (54),
    abort-proprietary (55),
    abort-other (56),
    authentication-failed (1),
    ...
    invalid-parameter-data-type (13),
    invalid-tag (57),
    invalid-time-stamp (14),
    key-generation-error (15),
    missing-required-parameter (16),
    network-down (58),
    no-objects-of-specified-type (17),
    ...
    read-access-denied (27),
    reject-buffer-overflow (59),
    reject-inconsistent-parameters (60),
    reject-invalid-parameter-data-type (61),
    reject-invalid-tag (62),
    reject-missing-required-parameter (63),
    reject-parameter-out-of-range (64),
    reject-too-many-arguments (65),
    reject-undefined-enumeration (66),
    reject-unrecognized-service (67),
    reject-proprietary (68),
    reject-other (69),
```

```

security-not-supported          (28),
service-request-denied         (29),
timeout                         (30),
unknown-device                 (70),
unknown-object                 (31),
unknown-property               (32),
unknown-route                  (71),
-- this enumeration was removed (33),
unknown-vt-class               (34),
unknown-vt-session             (35),
unsupported-object-type         (36),
value-not-initialized          (72),
value-out-of-range             (37),
...
-- see duplicate-object-id      (49),
-- see property-is-not-an-array (50),
-- see abort-buffer-overflow    (51),
-- see abort-invalid-apdu-in-this-state (52),
-- see abort-preempted-by-higher-priority-task (53),
-- see abort-segmentation-not-supported (54),
-- see abort-proprietary       (55),
-- see abort-other              (56),
-- see invalid-tag              (57),
-- see network-down             (58),
-- see reject-buffer-overflow   (59),
-- see reject-inconsistent-parameters (60),
-- see reject-invalid-parameter-data-type (61),
-- see reject-invalid-tag       (62),
-- see reject-missing-required-parameter (63),
-- see reject-parameter-out-of-range (64),
-- see reject-too-many-arguments (65),
-- see reject-undefined-enumeration (66),
-- see reject-unrecognized-service (67),
-- see reject-proprietary       (68),
-- see reject-other             (69),
-- see unknown-device           (70),
-- see unknown-route            (71),
-- see value-not-initialized     (72),
...

```

```

}
-- Enumerated values 0-255 are reserved for definition by ASHRAE. Enumerated values
-- 256-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23. The last enumeration used in this version is 47.

```

```

}

```

135-2004b-12. Add new Reliability enumeration to objects with a Reliability property.

Rationale

A new Reliability enumeration, *COMMUNICATION_FAILURE*, is introduced and added to new object types added in Addendum 135-2004b. Addendum 135-2004b-12 adds this enumeration to existing object types, in the standard and other addenda, that have a Reliability property.

Addendum 135-2004b-12

[Change **Clause 12.1.9**, pp.132-133, Accumulator Object Type]

12.1.9 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value property or the operation of the physical input in question is "reliable" as far as the BACnet Device or operator can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, NO_SENSOR, OVER_RANGE, UNDER_RANGE, OPEN_LOOP, SHORTED_LOOP, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}

[Change **Clause 12.2.9**, p.139, Analog Input Object Type]

12.2.9 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical input in question is "reliable" as far as the BACnet Device or operator can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, NO_SENSOR, OVER_RANGE, UNDER_RANGE, OPEN_LOOP, SHORTED_LOOP, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}

[Change **Clause 12.3.9**, p.144, Analog Output Object Type]

12.3.9 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical output in question is "reliable" as far as the BACnet Device or operator can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, OPEN_LOOP, SHORTED_LOOP, NO_OUTPUT, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}

[Change **Clause 12.4.8**, p.149, Analog Value Object Type]

12.4.8 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value is "reliable" as far as the BACnet Device can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, OVER_RANGE, UNDER_RANGE, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

[Change **Clause 12.6.9**, p.158, Binary Input Object Type]

12.6.9 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical input in question is "reliable" as far as the BACnet Device or operator can determine and, if not, why. The Reliability property for this object may have any of the following values:

{NO_FAULT_DETECTED, NO_SENSOR, OPEN_LOOP, SHORTED_LOOP, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

[Change **Clause 12.7.9**, p.163, Binary Output Object Type]

12.7.9 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical output in question is "reliable" as far as the BACnet Device or operator can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, NO_OUTPUT, OPEN_LOOP, SHORTED_LOOP, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

[Change **Clause 12.8.8**, p.168, Binary Value Object Type]

12.8.8 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value is "reliable" as far as the BACnet Device or operator can determine. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

[Change **Clause 12.15.10**, p.196, Life Safety Point Object Type]

12.15.10 Reliability

The ~~reliability~~ *Reliability* property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical input(s) in question are "reliable" as far as the BACnet Device or operator can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, OPEN_LOOP, SHORTED_LOOP, MULTI_STATE_FAULT, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

[Change **Clause 12.16.10**, p.202, Life Safety Zone Object Type]

12.16.10 Reliability

The ~~reliability~~ *Reliability* property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical input(s) in question are "reliable" as far as the BACnet Device or operator can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, OPEN_LOOP, SHORTED_LOOP, MULTI_STATE_FAULT, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

[Change **Clause 12.17.8**, p.209, Loop Object Type]

12.17.8 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value of the loop in question is reliable as far as the BACnet Device or operator can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, OPEN_LOOP, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

[Change **Clause 12.18.9**, p.214, Multi-state Input Object Type]

12.18.9 Reliability

The ~~reliability~~ *Reliability* property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical inputs in question are "reliable" as far as the BACnet Device or operator can determine and, if not, why. The Reliability property is required to be present if the Fault_Values property is present. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, NO_SENSOR, OVER_RANGE, UNDER_RANGE, OPEN_LOOP, SHORTED_LOOP, MULTI_STATE_FAULT, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

[Change **Clause 12.19.9**, p.218, Multi-state Output Object Type]

12.19.9 Reliability

The ~~reliability~~ *Reliability* property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical outputs in question are "reliable" as far as the BACnet Device or operator can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, OPEN_LOOP, SHORTED_LOOP, NO_OUTPUT, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

[Change **Clause 12.20.8**, pp.222-223, Multi-state Value Object Type]

12.20.8 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value is "reliable" as far as the BACnet Device or operator can determine and, if not, why. The Reliability property is required to be present if the Fault_Values property is present. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, MULTI_STATE_FAULT, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

[Change **Clause 12.22.12**, p.232, Program Object Type]

12.22.12 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the application-specific properties of the program object or the process executing the application program are "reliable" as far as the BACnet Device can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, PROCESS_ERROR, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

[Change **Clause 12.23.9**, p.237, Pulse Converter Object Type]

12.23.9 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value and/or Count properties or the operation of the physical input in question is "reliable" as far as the BACnet Device or operator can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, NO_SENSOR, OVER_RANGE, UNDER_RANGE, OPEN_LOOP, SHORTED_LOOP, UNRELIABLE_OTHER, *COMMUNICATION_FAILURE*, CONFIGURATION_ERROR}

If Input_Reference is configured to reference a property that is not of datatype Unsigned or INTEGER, or is otherwise not supported as an input source for this object, the Reliability property shall indicate CONFIGURATION_ERROR.

[Change **Clause 12.24.13**, p.244, Schedule Object Type]

12.24.13 Reliability

The Reliability property, of type BACnetReliability, provides an indication that the properties of the schedule object are in a consistent state. All non-NULL values used in the Weekly_Schedule, the Exception_Schedule, and the Schedule_Default properties shall be of the same datatype, and all members of the List_Of_Object_Property_References shall be writable with that datatype. If these conditions are not met, then this property shall have the value CONFIGURATION_ERROR. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, CONFIGURATION_ERROR, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

If the List_Of_Object_Property_References contains a member that references a property in a remote device, the detection of a configuration error may be delayed until an attempt is made to write a scheduled value.

[Change **Addendum 135-2004e-1**, **Clause 12.17.9**, p.6, Load Control Object Type]

12.17.9 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Load Control object is reliably reporting its compliance with any load shed requests. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}

[Change **Addendum 135-2004f-1**, **Clause 12.X.8**, p.4, Access Door Object Type]

12.X.8 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical inputs or outputs which comprise this door are "reliable" as far as the BACnet Device or operator can determine and, if not, why. The Reliability property for this object may have any of the following values:

{NO_FAULT_DETECTED, MULTISTATE_FAULT, CONFIGURATION_ERROR, *COMMUNICATION_FAILURE*, UNRELIABLE_OTHER}.

[Add a new entry to **History of Revisions**, p. 598]

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

HISTORY OF REVISIONS

<i>Protocol</i>		<i>Summary of Changes to the Standard</i>
<i>Version</i>	<i>Revision</i>	
...
1	7	<p>Addendum b to ANSI/ASHRAE 135-2004 Approved by the ASHRAE Standards Committee October 12, 2008; by the ASHRAE Board of Directors October 24, 2008; and by the American National Standards Institute October 27, 2008.</p> <ol style="list-style-type: none"> 1. Add a new Event Log object type. 2. Add a new Global Group object type. (Removed after third public review.) 3. Add a new Trend Log Multiple object type. 4. Harmonize the Trend Log object with the new Event Log and Trend Log Multiple objects. 5. Define a means for a device to provide a notification that it has restarted. 6. Define a means to configure a device to periodically send time synchronization messages. 7. Extend the number of character sets supported. (Removed after first public review.) 8. Enable devices other than alarm recipients to acknowledge alarms. 9. Allow MS/TP BACnet Data Expecting Reply frames to be broadcast. 10. Revise the Clause 5 state machines to handle slow servers. (Removed after second public review.) 11. Add new Error Codes and specify usage. 12. Add new Reliability enumeration to objects with a Reliability property.

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

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

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

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

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

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

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

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