BSR/ASHRAE Addendum *co* to
ANSI/ASHRAE Standard 135-2020

Public Review Draft

Proposed Addendum *co* to Standard 135-2020, BACnet® - A Data Communication Protocol for Building Automation and Control Networks

First Public Review (September 2023)
(Draft shows Proposed Changes to Current Standard)

This draft has been recommended for public review by the responsible project committee. To submit a comment on this proposed standard, go to the ASHRAE website at www.ashrae.org/standards-research--technology/public-review-drafts and access the online comment database. The draft is subject to modification until it is approved for publication by the Board of Directors and ANSI. Until this time, the current edition of the standard (as modified by any published addenda on the ASHRAE website) remains in effect. The current edition of any standard may be purchased from the ASHRAE Online Store at www.ashrae.org/bookstore or by calling 404-636-8400 or 1-800-727-4723 (for orders in the U.S. or Canada).

This standard is under continuous maintenance. To propose a change to the current standard, use the change submittal form available on the ASHRAE website, www.ashrae.org.

The appearance of any technical data or editorial material in this public review document does not constitute endorsement, warranty, or guaranty by ASHARE of any product, service, process, procedure, or design, and ASHRAE expressly disclaims such.

ASHRAE, 180 Technology Parkway NW, Peachtree Corners, GA 30092

[This foreword, the table of contents, the introduction, 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

The purpose of this addendum is to present a proposed change for public review. 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 proposed changes are summarized below.

135-2020*co*-1 Clarify Reliability-Evaluation, p. 3
135-2020*co*-2 Event and Fault Parameter Consistency, p. 5

In the following document, language to be added to existing clauses of ANSI/ASHRAE 135-2020 is indicated through the use of *italics*, while deletions are indicated by ~~strikethrough~~. Where entirely new subclauses are proposed to be added, plain type is used throughout. Only this new and deleted text is open to comment at this time. All other material in this document is provided for context only and is not open for public review comment except as it relates to the proposed changes.

The use of placeholders like XX, YY, ZZ, X1, X2, NN, x, n, ? etc., should not be interpreted as literal values of the final published version. These placeholders will be assigned actual numbers/letters only after final publication approval of the addendum.

135-2020*co*-1 Clarify Reliability-Evaluation

Rationale
Specify the expected actions when Out_Of_Service is TRUE and Reliability is written.
Move and clarify the reliability-evaluation language to improve the understanding of the different fault conditions.

[Change **Clause 12.1.8 Reliability**, p. 163]

12.1.8 Reliability

Several object types defined in this clause have a property called "Reliability" that indicates the existence of fault conditions for the object. Reliability-evaluation is the process of determining the value for this property. *See Clause 13.2.2.X*. ~~Reliability-evaluation is separate from, but used by, event reporting and may be supported by objects that do not support event reporting.~~

~~The first stage of reliability-evaluation is internal to the object and is completely defined by the device's vendor. The second stage, which is only found in certain object types, is the application of a fault algorithm. Faults detected in the first stage shall take precedence over faults detected in the second stage. See Clause 13.4 for fault algorithm definitions and see the object type definitions to determine the fault algorithm supported by any particular object type.~~ The different values that the Reliability property can take on are described below. Note that not all values are applicable to all object types.
…

[Add **Clause 13.2.2.X**, p. 641]

13.2.2.X Reliability-Evaluation

Reliability-evaluation is the process of determining the value for the Reliability property. Reliability-evaluation is separate from, but used by, event reporting and may be supported by objects that do not support event reporting.

The first stage of reliability-evaluation is internal to the object. The first stage checks for internal faults such as configuration errors, hardware faults, or communication failures. Except where required by an object type definition, the specific conditions checked for, the methods used, and the reliability values supported are defined by the device's vendor. Faults detected in the first stage shall take precedence over faults detected in the second stage.

The second stage, which is only found in certain object types, is the application of a standardized fault algorithm. The standard fault algorithms are defined in Clause 13.4. Each object type definition in Clause 12 indicates whether or not a standardized fault algorithm may be applied by objects of that type, and which standardized fault algorithms may be applied.

[Add **Clause 13.2.2.Y**, p. 641]

13.2.2.Y Simulating Events Using Out_Of_Service

The value of the object's Out_Of_Service property does not affect the event-state-detection process.

While Out_Of_Service is TRUE, writing to the object's monitored value can be used to simulate specific event states including values that would cause the application of an event algorithm or a fault algorithm. An application of a fault algorithm shall also result in a change to the value of the Reliability property.

While Out_Of_Service is TRUE, writing to the object's Reliability property can be used to simulate internal faults. See Clause 13.2.2.X. A successful write to the Reliability property shall cause the event state to be evaluated and block the object from updating the Reliability property. Writing the Out_Of_Service property to FALSE shall remove the block allowing the object to update the Reliability property.

It is a local matter what reliability values are supported when Out_Of_Service is TRUE. An unsuccessful attempt to write to the Reliability property shall return a Result(-) with 'Error Class' = PROPERTY and 'Error Code' = INVALID_VALUE_IN_THIS_STATE.

[Change **Clause 13.3.17 NONE Event Algorithm**, p. 673]

13.3.17 NONE Event Algorithm

The NONE event algorithm is a placeholder for the case where no event algorithm is applied by the object. This placeholder is used in those objects that are able to indicate which event algorithm they apply. For example, an Event Enrollment object's Event_Type property is set to NONE to indicate that it is not applying an event algorithm.
~~This event algorithm has no parameters, no conditions, and does not indicate any transitions of event state.~~

~~The NONE algorithm is used when only fault detection is in use by an object.~~

[Change **Clause 13.4.1 NONE Fault Algorithm**, p. 676]

13.4.1 NONE Fault Algorithm

The NONE fault algorithm is a placeholder for the case where no *standardized* fault algorithm is applied by the object *in the second stage of reliability-evaluation. This placeholder is used in those objects that are able to indicate which standardized fault algorithm they apply*. *For example, an Event Enrollment object's Fault_Type property is set to NONE to indicate that it is not applying a standardized fault algorithm.*

~~This fault algorithm has no parameters, no conditions, and does not indicate any transitions of reliability.~~

[Change **Clauses 12.10**, **12.11, 12.21, 12.22, 12.24, 12.36, 12.38, 12.42, 12.45, 12.46, 12.47, 12.48, 12.56, 12.62, 12.63, 12.64**]

[object type] objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. [object type] *objects that are capable of detecting faults, shall only perform the first stage of reliability-evaluation and shall not apply a standardized fault algorithm.* [object type] objects that support intrinsic reporting shall ~~apply the NONE~~ *not apply an* event algorithm.
…

[Change **Table 12-15, Table 12-15.1**, **Table 12-15.2**, **Table 12-15.3**]

NONE	~~none~~*n/a*	~~none~~*n/a*

135-2020*co*-2 Event and Fault Parameter Consistency

Rationale
Several event and fault algorithms have parameters that may be configured in conflict with each other. This may have different effects such as the algorithm being switched off, OFFNORMAL or even LIFESAFETY_ALARM states not being detected, ambiguity in resulting event state, or infinite event state transition loops.

The CHANGE_OF_CHARACTERSTRING, CHANGE_OF_LIFE_SAFETY, and the CHANGE_OF_STATE event algorithms have paralleling fault algorithms with which the offnormal values lists should be coordinated. Having a particular value in both an event parameter value list and in the fault parameter values list leads to the value in the event parameter being ignored, and only TO_FAULT event transitions are indicated for that particular value.

The CHANGE_OF_LIFE_SAFETY event algorithm, as currently specified, even goes into an infinite loop between OFFNORMAL and LIFESAFETY_ALARM event state if a particular state value is present in both pAlarmValues and pLifeSafetyAlarmValues parameters.

The DOUBLE_OUT_OF_RANGE, OUT_OF_RANGE, SIGNED_OUT_OF_RANGE, UNSIGNED_OUT_OF_RANGE, and UNSIGNED_RANGE event algorithms all have a low limit and a high limit parameter. These limits, when configured with pLowLimit > pHighLimit, may cause the algorithm, for particular monitored values, to loop between HIGH_LIMIT and LOW_LIMIT event states.

The FLOATING_LIMIT algorithm, when configured with negative values in pLowLimit or pHighLimit, may cause the algorithm, for particular monitored values, to continuously loop between HIGH_LIMIT and LOW_LIMIT event states.

The FAULT_CHARACTERSTRING, FAULT_LIFE_SAFETY, and FAULT_STATE algorithms pFaultValues parameter should not include a value that is also present in an event algorithm parameter of the object. Otherwise, this value is never indicated as a FAULT event state transition only.

The FAULT_OUT_OF_RANGE fault algorithm parameters, when configured to be pMinimumNormalValue > pMaximumNormalValue, may cause the algorithm, for particular monitored values, to continuously toggle between UNDER_RANGE and OVER_RANGE reliability.

Objects containing such conflicting event and fault parameters are taking a reliability status of CONFIGURATION_ERROR. This inhibits execution of the event algorithm and the fault algorithm if present, on such conflicting configurations. Respective language is added to event and fault parameter properties describing the situations considered a conflicting configuration.

[Change **Clause 12.12.21**, Event Enrollment object type, p. 237]

12.12.21 Reliability

…
The Reliability property shall take on a value of CONFIGURATION_ERROR under these conditions:

- *For the CHANGE_OF_CHARACTERSTRING event algorithm and FAULT_CHARACTERSTRING fault algorithm, if a particular value is present in both the List_Of_Alarm_Values parameter in the Event_Parameters property and in the List_Of_Fault_Values parameter in the Fault_Parameters property, or*
- *for the CHANGE_OF_LIFE_SAFETY event algorithm and FAULT_LIFE_SAFETY fault algorithm, if a particular value is present in more than one of the List_Of_Alarm_Values parameter or the List_Of_Life_Safety_Alarm_Values parameter in the Event_Parameters property and the List_Of_Fault_Values parameter in the Fault_Parameters property, or*
- *for the CHANGE_OF_STATE event algorithm and the FAULT_STATE fault algorithm, if a particular value is present in both the List_Of_Values parameter in the Event_Parameters property and the List_Of_Fault_Values parameter in the Fault_Parameters property, or*
- *for the DOUBLE_OUT_OF_RANGE, OUT_OF_RANGE, SIGNED_OUT_OF_RANGE, UNSIGNED_OUT_OF_RANGE, and UNSIGNED_RANGE event algorithms, if the Low_Limit parameter is greater than the High_Limit parameter, or*

- *for the FLOATING_LIMIT event algorithm, if the Low_Diff_Limit parameter or the High_Diff_Limit parameter or both have numerically negative values.*

[Change **Clause 12.37.8,** CharacterString Value object type, p. 417;
Change **Clause 12.18.9,** Multi-state Input object type, p. 270;
Change **Clause 12.20.8,** Multi-state Value object type, p. 283;
Change **Clause 12.26.8,** Access Door object type, p. 328]

12.?.? Reliability

…
The Reliability property shall take on a value of CONFIGURATION_ERROR if a particular value is present in both the Alarm_Values property and in the Fault_Values property.

[Change **Clause 12.3.9**, Analog Output object type, p. 174;
Change **Clause 12.23.9**, Pulse Converter object type, p. 303]

12.?.9 Reliability

…
The Reliability property shall take on a value of CONFIGURATION_ERROR if both limits are enabled in the Limit_Enable property, and the value of the High_Limit property is less than the value of the Low_Limit property.

[Change **Clause 12.2.9**, Analog Input object type, p. 168;
Change **Clause 12.4.8**, Analog Value object type, p. 181;
Change **Clause 12.39.8**, Large Analog Value object type, p. 430;
Change **Clause 12.43.8**, Integer Value object type, p. 453;
Change **Clause 12.44.8**, Positive Integer Value object type, p. 460;
Change **Clause 12.61.9**, Accumulator object type, p. 603]

12.?.? Reliability

…
The Reliability property shall take on a value of CONFIGURATION_ERROR under these conditions:
- *If both limits are enabled in the Limit_Enable property, and the value of the High_Limit property is less than the value of the Low_Limit property, or*
- *if the value of Fault_High_Limit property is less than the value of the Fault_Low_Limit property.*

[Change **Clause 12.17.8**, Loop object type, p. 263]

12.17.8 Reliability

…
The Reliability property shall take on a value of CONFIGURATION_ERROR under these conditions:
- *If the value of the Error_Limit property is less than zero, or*
- *if the value of the Low_Diff_Limit property is less than zero.*

[Change **Clause 12.15.10,** Life Safety Point object type, p. 247;
Change **Clause 12.16.10,** Life Safety Zone object type, p. 254]

12.?.10 Reliability

…
The Reliability property shall take on a value of CONFIGURATION_ERROR if a particular value is found in more than one of the Fault_Values property, the Alarm_Values property, or the Life_Safety_Alarm_Values property.

