BSR/ASHRAE Addendum *cu* to
ANSI/ASHRAE Standard 135-2024

Public Review Draft

Proposed Addendum *cu* to Standard 135-2024, BACnet® - A Data Communication Protocol for Building Automation and Control Networks

First Publication Public Review (March 2025)
(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 20092

[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.

In the following document, language to be added to existing clauses of ANSI/ASHRAE 135-2024 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.

135-2024*cu*-1 BACnet Directory Services

Rationale

The existing standard has several mechanisms for device and object discovery. In practice these mechanisms can produce undesirable side effects such as broadcast traffic. This is particularly bad in systems with many BACnet clients independently doing device and object discovery.

This addendum proposes a new kind of device – a directory server – that can be used to centralize knowledge about the device characteristics and objects in all BACnet devices in a given system.

[Insert New Clause, 12.X, pg. ?]

12.X Directory Object Type

The Directory object type defines a standardized object that represents a directory of devices and their objects that can be queried using the Directory-Query service. There shall be at most a single Directory object, always instance 1, in any device that is a BACnet Directory Server (BDS). Devices that are not directory servers shall not contain any Directory objects.

Table 12-X. Properties of the Directory Object

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	R
Discovery_Status	BACnetDiscoveryState	R
Directory_Revision	Unsigned	R
Enable	Boolean	R
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	R
Event_Detection_Enable	BOOLEAN	O[1]
Notification_Class	Unsigned	O[1]
Event_Enable	BACnetEventTransitionBits	O[1]
Acked_Transitions	BACnetEventTransitionBits	O[1]
Notify_Type	BACnetNotifyType	O[1]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[1]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[1]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[1]
Event_State	BACnetEventState	O[1]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[2]
Auditable_Operations	BACnetAuditOperationFlags	O[2]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties shall be present only if the object supports intrinsic reporting.
[2] This property shall be present only if the device supports audit reporting.

12.X.1 Object_Identifier

This read-only 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. The value of this property shall always be (DIRECTORY,1).

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 read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be DIRECTORY.

12.X.4 Description

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

12.X.5 Discovery_Status

This read-only property, of type BACnetDiscoveryState, shall indicate the current status of the directory of devices and objects. If the directory server has been disabled because its Enable property has the value FALSE, then this property value shall be DISABLED. If the directory server has not yet been configured, then this property value shall be UNCONFIGURED. If the directory server has been configured, and is in the process of discovering devices and their objects, then this property value shall be INPROGRESS. If the directory server has finished discovery of all devices that it can find, then this property value shall be COMPLETE.

12.X.6 Directory_Revision

This read-only property, of type Unsigned, shall indicate the current revision number of the directory. Each time the directory is changed ~~due to discovery or update,~~ this property shall be incremented.

12.X.7 Enable

This property, of type Boolean, shall indicate whether the directory server that contains this object is enabled for operation (TRUE) or not (FALSE).

12.X.8 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Network. The four flags are:

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property 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.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the Directory object, and the underlying database are "reliable" as far as the BACnet device can determine and, if not, why.

12.X.10 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.X.11 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.X.12 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (F, T, T) at a minimum.

12.X.13 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.3). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.X.14 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.X.15 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.X.16 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.X.17 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.X.18 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.X.19 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.X.20 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.X.21 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.X.22 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.X.23 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.X.24 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.X.25 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall 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. 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. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

[Add to Table 13-5. p.680]

Table 13-5. Properties Reported in CHANGE_OF_RELIABILITY Notifications

Object Type	Properties
...	
Directory	*Discovery_Status* *Directory_Revision* *Enable*

[Add to Clause 16]

Directory Servers

A BACnet Directory Server (BDS) is a BACnet device that maintains a directory of devices and their objects. The BDS directory may be configured manually, or populated automatically using a discovery procedure, or both. The directory maintains information about devices in the BACnet internetwork, as well as the objects in those devices. The methods used by the BDS to automate the discovery, as well as the format and structure of any database it may use, are a local matter.

Conceptually a BDS could use the Who-Is service to find available BACnet devices, and record specific information returned in I-Am messages. It could also listen for I-Am messages that might be broadcast locally or globally. Upon discovering a device, the BDS could then supplement its information by explicitly reading properties of the Device object such as the Device Object_Name and Object_List. Once the Object_List is known for a device, the BDS could read individual object's properties such as the Object_Name. Once a device and its objects have been discovered, or manually added to the directory or imported from external sources, the BDS could periodically check for updates to the objects in each device, for example by reading the Database_Revision property of the Device object and comparing it to saved information for that device. Any device found to have a newer revision could then be re-inspected and the directory updated.

If the BDS is disabled because the (Directory,1) object Enable property has the value FALSE then no discovery shall take place. If the Enable property is writable, and is written with a FALSE value, then any discovery operation in progress shall be stopped.

Independent of the background process that manages device and object discovery, the BDS can process incoming DirectoryQuery service requests to provide client devices with information about devices and their objects. Client devices can locate BDS devices using the Who-Has service. Figure 16-X shows these relationships.

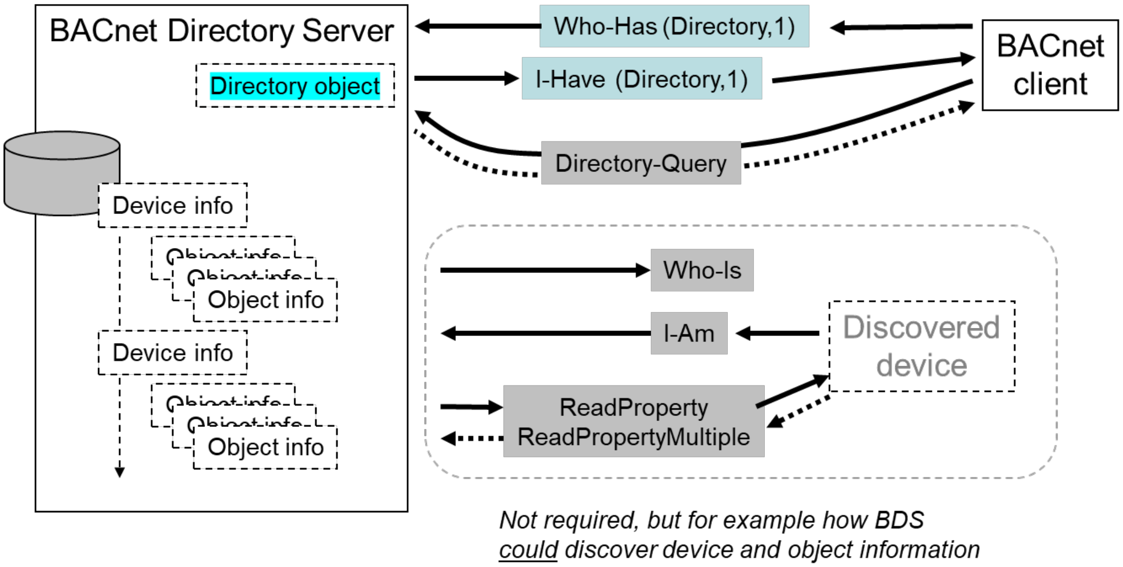

Figure 16-X. BACnet Directory Server and Client Devices

Locating BACnet Directory Servers

Each BDS shall implement the Who-Has service and shall contain one and only one Directory object, always instance 1. BACnet client devices wishing to locate a BDS shall transmit a Who-Has request specifying (Directory,1) as a global broadcast. The BDS(s) reply with an I-Have that includes the server (Directory,1) object. The client may read the (Directory,1) object property Enable to determine if that BDS is enabled for directory queries or not. Alternatively, the client can attempt to send a DirectoryQuery request which will return an error if the BDS is disabled. When there is more than one BDS, the choice of which BDS to use shall be a local matter. Clients shall restrict the frequency of Who-Has requests in search of BDS such that if no BDS replies to the Who-Has request within 60 seconds, the client shall refrain from retrying for a period of 3600 seconds.

The intent is that each BDS shall contain exactly the same directory, such that a device may choose any BDS that is available. The methods that are used to assure that multiple BDS directories are synchronized shall be a local matter.

DirectoryQuery Service

The DirectoryQuery service is used by a sending BACnet-user to query BDS(s) about their known devices and objects. The service supports a complex query with several optional elements. In various combinations this allows the BACnet-user to query many use cases including:

- Find a device based on its device instance number
- Find a device based on its device Object_Name
- Find all devices optionally qualified by network number
- Find devices within a range of device instances
- Find devices whose Device Object_Name matches a string pattern
- Find objects in a given device optionally qualified by object type
- Find objects whose Object_Name matches a string pattern

Pattern Matching in Object_Name and Tags Properties

The ‘Device Qualifier’ and ‘Object Name Qualifier’ parameters provide the ability to filter the returned results based on pattern matching in Object_Name properties of objects in the query results. When pattern matching is specified, the case shall not be significant, e.g. “ABC”, “Abc” and “aBc” are all the same with respect to pattern matching. The pattern string shall not contain any double quote characters (“). There are two special characters that affect pattern matching.

The character question mark (?) shall match any single character in that position, for example “A?C” would match “AAC” and “ABC”.

The character asterisk (*) may only be used as the first or last character of the pattern string. If the pattern string ends with asterisk, then only those objects whose Object_Name begins with the portion of the pattern string that precedes the asterisk will be matched. For example, “AB*” would match “ABC”, “ABCDEF” and “AB”. If the pattern string begins with an asterisk then only Object_Names that end with the string that follows the asterisk will be matched. For example, “*AB” would match “CAB”, “BIGBLAB” and “XYAB”. If the pattern string begins and ends with asterisk, then only those Object_Names that contain the string between the asterisks will be matched. For example, “*AB*” would match “ABC”, “DOCKABLE” and “TAKEACAB”.

Structure

The structure of the DirectoryQuery service primitives is shown in Table 16-X. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-X. Structure of DirectoryQuery Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Device Qualifier	M	M(=)		
Network Qualifier	U	U(=)		
Object Type Qualifier	U	U(=)		
Object Name Qualifier	U	U(=)		
Response Includes	M	M(=)		
Include Proprietary Details	U	U(=)		
Start Cursor	U	U(=)		
Max Results	U	U(=)		
Result (+)			S	S(=)
Directory Revision			M	M(=)
Device Instances			C	C(=)
Device Details			C	C(=)
More Cursor			C	C(=)
Result (-)			S	S(=)
Error Type			M	M(=)

Argument

This parameter shall convey the parameters for the DirectoryQuery confirmed service request.

Device Qualifier

This parameter, of type BACnetDeviceQualifier, shall indicate the selection criteria for choosing device instances whose device information will be returned in the ‘Device Instances’ or ‘Device Details’ parameters of the ‘Result (+)’. The list of candidate instances may be further reduced by additional qualifiers that may be provided in the request.

If the ‘Device Qualifier’ is ‘all’, then all device instances shall be returned. If ‘Device Qualifier’ is ‘instance-set’ then only those device instances in the ‘instance-set’ list that are known to BDS shall be returned. If ‘Device Qualifier’ is ‘instance-range’ then only those device instances within the inclusive range starting with ‘instance-low’ and ending with ‘instance-high’ that are known to BDS shall be returned. If the ‘Device Qualifier’ is ‘pattern’ then only those devices whose Device Object_Name matches the ‘pattern’ shall be returned (see 16.13.2.1).

Network Qualifier

This parameter, of type BACnetNetworkQualifier, shall convey either a list of BACnet network numbers, or a range of BACnet network numbers. Only those devices matching ‘Device Qualifier’ whose network number is in this list or range shall be returned in the ‘Device Instances’ or ‘Device Details’ parameters of the ‘Result (+)’. If this parameter is not present, or the list is empty, then all devices matching other qualifiers shall be returned.

Object Type Qualifier

This parameter, of type BACnetLIST of BACnetObjectType, shall convey a list of BACnet object types. Only those devices matching previous qualifiers that contain objects whose Object_Type is in this list shall be returned in the ‘Device Instances’ or ‘Device Details’ parameters of the ‘Result (+)’.

Object Name Qualifier

This parameter, of type CharacterString, shall indicate a pattern to be matched against the Object_Name of all objects contained in devices matching previous qualifiers. Only those devices matching previous qualifiers that contain objects whose names match the pattern shall be returned in the ‘Device Instances’ or ‘Device Details’ parameters of the ‘Result(+)’ (see 16.13.2.1).

Response Includes

This parameter, of type BACnetResponseIncludes, shall indicate what specific information shall be returned in the ‘Result(+)’.

‘Response Includes’ choice	*what is returned*
instances	‘Device Instances’
basic-details	‘Device Details’ without the optional device details and with an empty list of ‘objects’.
full-details	‘Device Details’ including the optional ‘extended-details’ and with an empty list of ‘objects’.
basic-objects	‘Device Details’ including the optional ‘extended-details’ and with a list of ‘objects’ that includes all matching objects based on object qualifiers.
full-objects	‘Device Details’ including the optional ‘extended-details’ and with a list of ‘objects’ that includes all matching objects based on object qualifiers. Each of the objects shall include ‘object-name’ and ‘last-updated’.

Include Proprietary Details

This parameter, of type Boolean, shall specify whether the response shall include proprietary details or not. If this parameter is not present it shall default to FALSE.

Start Cursor

This parameter, of type Unsigned32, shall specify a number meaningful to the BDS that was returned in a previous DirectoryQuery result ‘More Cursor’ parameter. This number represents the position in a long list of results from which this request’s results should begin.

Max Results

This parameter, of type Unsigned, shall specify the maximum number of results that BDS may return in any single ‘Result(+)’ response. If the ‘Max Results’ parameter is provided, and the number of results that match a given query is greater than ‘Max Results’, then only the first ‘Max Results’ result elements shall be returned for ‘Device Instances’ or ‘Device Details’.

Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded. A successful result includes the following parameters:

Directory Revision

This parameter, of type Unsigned, shall indicate the current revision of the directory, mirroring the value of the (Directory,1) object’s Directory_Revision property.

Device Instances

This conditional parameter, of type BACnetLIST of Unsigned, shall identify the device instance numbers of those devices that match the combined criteria in the qualifier parameters of the request. This parameter shall only be present when the ‘Response Includes’ is ‘instances’.

Device Details

This conditional parameter, of type BACnetLIST of BACnetDeviceDetails, shall identify detailed information about the found device(s) and object(s). This parameter shall only be present when ‘Response Includes’ is not ‘instances’. Within the BACnetDeviceDetails, the ‘last updated’ element(s) of a device detail or object detail shall indicate the date and time when the BDS directory was last updated with these details from the device. Within a given BACnetObjectDetails, if the corresponding object contains a Profile_Name property then it shall be reflected in the returned object detail. If the corresponding object contains a Tags property then it shall be reflected in the returned object detail. If the ‘Include Proprietary Details’ parameter of the request was TRUE, then the device details shall include proprietary details about the device.

More Cursor

This conditional parameter, of type Unsigned32, shall specify a number meaningful to the BDS that represents the position in a long list of results from which subsequent DirectoryQuery request's results should begin. This parameter shall only be present when there are more results available than have been returned in this result.

Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed in its entirety. The reason for the failure shall be specified by the 'Error Type' parameter.

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:

Situation	Error Class	Error Code
Specified cursor is no longer valid	SERVICES	INVALID_CURSOR
Directory object Enable is FALSE	SERVICES	DIRECTORY_DISABLED
Directory can't be accessed	SERVICES	DIRECTORY_QUERY_FAILED

Service Procedure

After verifying the validity of the request, the responding BACnet-user shall first attempt to connect to its local directory. If the directory is disabled, as indicated by the Directory object property Enable having a value of FALSE, a 'Result(-) primitive shall be generated with the 'Error Class' SERVICES and 'Error Code' DIRECTORY_DISABLED. Otherwise, if the directory cannot be opened or some other internal condition prevents the directory query operation, a 'Result(-) primitive shall be generated with the 'Error Class' SERVICES and 'Error Code' DIRECTORY_QUERY_FAILED. Otherwise attempt to query the directory in search of device and object information based on the request parameter(s). If the access is successful, a 'Result(+)' primitive shall be generated.

[Add to Annex K.5 Device Management BIBBs, p.1149]

K.5.X BIBB - Device Management - BACnet Directory Services-A (DM-BDS-A)

The A device seeks to locate BACnet Directory Server(s) that can provide information about devices and their objects and will query the BDS to discover this kind of information.

BACnet Service	Initiate	Execute
Who-Has	x	
I-Have		x
DirectoryQuery	x	

K.5.X BIBB - Device Management - BACnet Directory Services-B (DM-BDS-B)

The B device maintains a database of discovered/configured devices and their objects and properties and provides a BACnet Directory Server that can respond to queries for information about those devices and objects.

BACnet Service	Initiate	Execute
Who-Has		x
I-Have	x	
DirectoryQuery		x
Who-Is	x	
I-Am		x
ReadProperty	x	
ReadPropertyMultiple	x	

[Change Annex L DESCRIPTIONS AND PROFILES OF STANDARDIZED BACnet DEVICES, p.1136]

- Miscellaneous. This family is composed of B-RTR, B-GW, B-BBMD, B-ACDC, B-ACCR,~~ and~~ B-SCHUB, *and B-BDS*.

[Add column to tables in L.7 Miscellaneous Profiles, p.1149]

Data Sharing

B-BDS
DS-RP-A
DS-RPM-A
DS-RP-B

Device & Network Management

B-BDS
DM-DDB-A
DM-DDB-B
DM-DOB-B

[Change Clause 21]

```
BACnetConfirmedServiceChoice ::= ENUMERATED {
        acknowledge-alarm                   (0),
…
        -- subscribe-cov-property-multiple  (30)
        -- confirmed-cov-notification-multiple (31)
        directory-query                     (x)
        }

BACnet-Confirmed-Service-Request ::= CHOICE {
        -- Alarm and Event Services
        acknowledgeAlarm        [0] AcknowledgeAlarm-Request,
        …
        -- Directory Services
        DirectoryQuery                      [x] DirectoryQuery-Request
        }

BACnet-Confirmed-Service-ACK ::= CHOICE {
        …
        -- Alarm and Event Services
        getAlarmSummary         [3] GetAlarmSummary-ACK,
        …
        -- Directory Services
        DirectoryQuery                      [x] DirectoryQuery-ACK
        }

DirectoryQuery-Request ::= SEQUENCE {
        deviceQualifier                     [0] BACnetDeviceQualifier,
        networkQualifier                    [1] BACnetNetworkQualifier OPTIONAL,
        objectTypeQualifier                 [2] SEQUENCE OF BACnetObjectType OPTIONAL,
        objectNameQualifier                 [3] CharacterString OPTIONAL,
        responseIncludes                    [4] BACnetResponseIncludes,
        includeProprietaryDetails           [5] BOOLEAN OPTIONAL,
        startCursor                         [6] Unsigned32 OPTIONAL,
        maxResults                          [7] Unsigned OPTIONAL
        }

DirectoryQuery-ACK ::= SEQUENCE {
-- Note [1][2] are mutually exclusive with each other
        directoryRevision                   [0] Unsigned,
        deviceInstances                     [1] SEQUENCE OF Unsigned OPTIONAL,
        deviceDetails                       [2] SEQUENCE OF BACnetDeviceDetails OPTIONAL,
        moreCursor                          [3] Unsigned32 OPTIONAL
        }

BACnetServicesSupported ::= BIT STRING {
…
        directory-query                     (x),
        }

BACnetError ::= CHOICE {
…
DirectoryQuery                              [x] Error
```

```
Error ::= SEQUENCE {
…
        directory-disabled              (x),
        directory-query-failed          (x+1),
        invalid-cursor                  (x+2),

BACnetDeviceQualifier ::= CHOICE {
        all                             [0] null,
        instance-set                    [1] SEQUENCE OF Unsigned,
        instance-range                  [2] SEQUENCE {
                instance-low                    Unsigned,
                instance-high                   Unsigned
                                            },
        pattern                         [3] CharacterString
        }

BACnetDeviceDetails ::= SEQUENCE {
        device-instance                 [0] Unsigned,
        network-number                  [1] Unsigned16,
        mac-address                     [2] OctetString,
        vendor-id                       [3] Unsigned16,
        max-apdu                        [4] Unsigned,
        segmentation                    [5] BACnetSegmentation,
        last-updated                    [6] BACnetDateTime,
        extended-details                [7] SEQUENCE {
        device-name                         [0] CharacterString,
        last-database-revision              [1] Unsigned,
        serial-number                       [2] CharacterString OPTIONAL,
        protocol-revision                   [3] Unsigned,
        protocol-services-supported         [4] BACnetServicesSupported
                                            } OPTIONAL,
        objects                         [8] SEQUENCE OF BACnetObjectDetails,
        proprietary-details             [9] ABSTRACT-SYNTAX.&Type OPTIONAL
        }

BACnetNetworkQualifier ::= CHOICE {
        network-set                     [1] SEQUENCE OF Unsigned16,
        network-range                   [2] SEQUENCE {
                instance-low                    Unsigned16,
                instance-high                   Unsigned16
                                            }
        }

BACnetResponseIncludes :: = ENUMERATED {
        instances                       (0),
        basic-details                   (1),
        full-details                    (2),
        basic-objects                   (3),
        full-objects                    (4)
        }

BACnetObjectDetails ::= SEQUENCE {
        object-identifier               [0] BACnetObjectIdentifier,
        last-updated                    [1] BACnetDateTime,
        object-name                     [2] CharacterString OPTIONAL,
        profile-name                    [3] CharacterString OPTIONAL,
```

    tags   [4] BACnetARRAY[N] of BACnetNameValue OPTIONAL
    }

***BACnetDiscoveryState**:: = ENUMERATED {*
    unconfigured   (0),
    inprogress   (1),
    complete   (2),
    disabled   (3)
    }

BACnetObjectType ::= Enumerated { -- see below for numerical order
…
    device   (8),
    directory   (n),
    ...
    -- color-temperature   (64),
    ...
    -- directory   (n),

BACnetObjectTypesSupported ::= BIT STRING {
…
    directory   (n),

BACnetPropertyIdentifier ::= ENUMERATED { -- see below for numerical order
…
    discovery-status   (n),
    directory-revision   (n+1),

BACnetPropertyStates ::= CHOICE {
…
    response-includes   [n]   BACnetResponseIncludes,
    discovery-state   [n+1]   BACnetDiscoveryState,

[Add a new entry to **History of Revisions**, p. 1364]

(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

. . .	. . .	**. . .**
1	X	**Addendum *cu* to ANSI/ASHRAE Standard 135-2024** Approved by ASHRAE on MONTH DAY, 20XX; and by the American National Standards Institute on MONTH DAY, 20XX. 1. Addition of BACnet Directory Services and Directory Object Type