Volume 8 • Issue 5
SEPTEMBER-OCTOBER 2007

the EXTENSION
A Technical Supplement to Control Network

Object Modeling a Physical BACnet Device

By George Thomas

This is a two-part series. Part 1 is entitled Object Modeling a Physical BACnet Device while Part 2 is entitled Achieving BACnet Compliance. Together they explain how BACnet devices are constructed —with emphasis on the latest BACnet/IP standard.

Introduction

Modern control networks such as EtherNet/IP, Modbus/TCP and BACnet/IP use Ethernet for communications due to its high speed, lowering cost and in some instances, the necessity to operate over structured wiring. Although the study of Ethernet does not require an understanding of application protocols, knowledge of application protocols has become increasingly important as modern networks are deployed. The latest protocols are all based upon Object Modeling which can be quite confusing to someone who has not been exposed to this abstract concept. This paper introduces object modeling, object properties, and services as they pertain to a physical BACnet/IP device. Although the majority of BACnet devices support the master-slave/token-passing (MS/TP) network, newer devices now function over Ethernet.

BAS Remote as a Physical Device

Although any other product could have been used, we will use as our physical device example the BAS Remote by Contemporary Controls for this discussion. This product is intended as a remote input/output device complying with the BACnet/IP standard. When connected to an Ethernet infrastructure, the BAS Remote provides eight input/output points that can be accessed by any device on a BACnet network. Six of those points are universal input/outputs— meaning they can be field configurable to be either an analog or digital input or output. Two are fixed as relay outputs. The BAS Remote supports the BACnet/IP standard as defined in Annex J of the BACnet standard and therefore, requires no router in order to connect to Ethernet. But how do other devices access the BAS Remote? To explain how this works, one needs to study BACnet's object model and services.

The BACnet Standard

The current revision of the BACnet standard is ANSI/ASHRAE 135-2004 from the American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc. If you purchase the standard from the ASHRAE bookstore, you will receive a 600-page bound copy along with a stack of errata. In the errata are revisions that have yet to make it to the printed bound copy, and interpretations of the standard made by committee members in response to questions, proving that the BACnet standard continues to evolve. Like any other standard, the BACnet standard is not written as a tutorial on BACnet although some exist on the web. However, since the BAS Remote is a relatively simple BACnet device, studying this product offers an opportunity to learn how a device becomes BACnet-compliant.

Object Modeling the BAS Remote

Observing the BAS Remote sitting on a table, it appears in its physical form as a plastic enclosure with screw connectors for powering the unit and for connecting to field devices such as sensors and actuators (see Figure 1). According to the BAS Remote

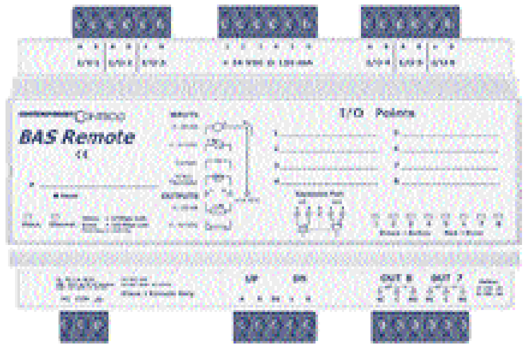

Figure 1—The BAS Remote has six universal input/output points and two relay outputs.

data sheet, in its simplest form it has six universal input/output points and two relay outputs. The six universal I/Os can be individually configured to be an analog input, analog output or digital input. In addition, the circuitry is flexible in that it can accommodate several voltage and current levels such as 0–10 VDC or 4–20 mA. Contact closures or thermistors can be sensed directly. There is one RJ-45 connector for the Ethernet network. To someone in the industry, the BAS Remote would be considered an eight-point remote I/O device with an Ethernet connection. Obviously, the Ethernet connection allows network access to the various I/O points but how is this accomplished in the BACnet world? Once configured for the needs of the application, how can each physical point on the BAS Remote be queried from any point on the network without any prior knowledge of the capabilities of the BAS Remote? Object modeling is the answer.

Although the BAS Remote is an eight-point I/O device, we know little more. To understand this device or any other BACnet device, it is best to view these devices as a collection of objects. An object is an abstract concept that allows us to identify both physical items such as I/O points, and non-physical concepts such as software and calculations. In Figure 2, we notice that the BAS Remote can be represented

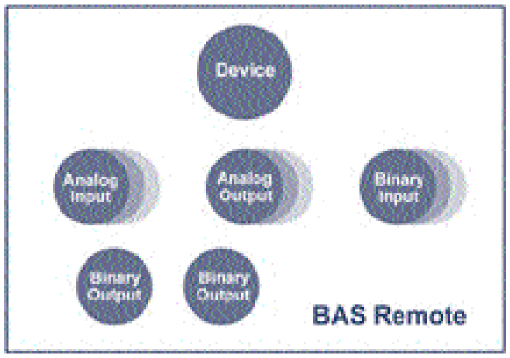

Figure 2—The BAS Remote can be described as a collection of objects with one identifying the device itself.

as a collection of objects of varying types such as Analog Input, Analog Output, Binary Input, Binary Output and Device. Since the BAS Remote has eight points of I/O, there will be eight objects created plus one for the device itself. Since six of the I/O points are universal, the number of each object type will vary depending upon the final configuration. This is not true for the Device object since only one is allowed per physical device. There will be two Binary Output objects to correspond to the two relay outputs on the BAS Remote. In total, the BAS Remote can be modeled as a collection of nine objects with up to five object types. As shown in Table 1, the current BACnet standard identifies 25 object types that would typically be found in building automation systems. BACnet-compliant devices are not required to implement all 25.

Objects Have Properties

Although objects can represent physical points, they can also describe processes or internal operations. Each object has a listing of properties that tell us about the nature of the object. This object listing varies with the object type. What the BACnet standard does is define the object types and the property fields for each object type. In this way, uniformity is ensured among vendors that will lead to interoperability between different vendor systems. The intent is to make the objects "network visible"—any object can be queried from any point along the network. As the standard evolved, object types have been added from the initial 18 to the current 25. These 25 object types are referred to as *standard objects*. A BACnet device need not support all object types, but if an object type is supported, it must comply with the standard object model for that object type.

Each object type has a list of required properties and optional properties. Optional properties can be included at the manufacturer's discretion.

There is one object type that must be included in any BACnet-compatible device and that is the *Device object*. There must be one and only one instance of the Device object whose structure is shown in Table 2. An instance number is the way of identifying items using object modeling. The Object Identifier must be unique within the complete BACnet network. It is comprised of the object type Device and an instance number. The default instance is 2749, but there is no significance to this value since it is changed by the installer through the web page during commissioning. The Object Name can be set as well to some meaningful description such as the location of the device. The remaining fields are set by the manufacturer.

Accumulator
Analog Input
Analog Output
Analog Value
Averaging
Binary Input
Binary Output
Binary Value
Calendar
Command
Device
Event Enrollment
File
Group
Life Safety Point
Life Safety Zone
Loop
Multi-state Input
Multi-state Output
Multi-state Value
Notification Class
Program
Pulse Converter
Schedule
Trend Log

Table 1—There are 25 BACnet object types. The BAS Remote uses five which are highlighted.

Property Identifier	Value
Object_Identifier	(Device, Instance 2749)
Object_Name	"RE1 Penthouse"
Object_Type	DEVICE
System_Status	(OPERATIONAL)
Vendor_Name	"Contemporary Controls"
Vendor_Identifier	245
Model_Name	"BASR-8M"
Firmware_Revision	"1.0"
Application_Software_Version	"1.0"
Protocol_Version	2
Protocol_Revision	
Protocol_Services_Supported	(List of services)
Protocol_Object_Types_Supported	(List of object types)
Object_List	(List of all the objects)
Max_APDU_Length_Accepted	1476
Segmentation_Supported	(NO SEGMENT)
APDU_Timeout	(3000 MSEC)
Number_Of_APDU_Retries	0
Device_Address_Binding	
Database_Revision	1

Table 2—The required properties of the Device object are listed. From this object, much can be learned about the device.

From the other properties you can learn much about the device. All the object types present in the device are listed under Protocol_Object_Types_Supported. All the objects within the device along with their instance number are included under Object_List.

In our example of the BAS Remote, there could be five object types and nine objects listed. We will study one of those objects of the type Analog Input.

In Table 3 you will see all the required properties of one Analog Input instance which the manufacturer identifies as AI, Instance 2 meaning that the second physical point on the BAS Remote happens to be configured as an analog input. All properties in any object must be able to be read. Point 2 on the BAS Remote is connected to a thermistor located on the roof of the building in order to measure outside temperature. The actual temperature appears in the Present_Value property as 69.3. The unit of measure is degrees Fahrenheit as indicated in the Units property. The installer has identified the point as "Outside temperature" in the Object_Name property meaning that another BACnet device on the network can search for "Outside temperature" and obtain the actual outside temperature without caring which device or which point had the result. This demonstrates the power of object modeling.

Property Identifier	Remarks
Object_Identifier	(Analog Input, Instance 2)
Object_Name	"Outside temperature"
Object_Type	ANALOG INPUT
Present_Value	69.3
Status_Flags	
Event_State	
Out_Of_Service	(OPERATIONAL)
Units	(degrees Fahrenheit)

Table 3—The required properties of the Analog Input object for Instance 2. Notice that the value of the point is maintained in engineering units.

Devices Provide Services

Having just objects with properties is not sufficient. We need services so that one BACnet device can access data from another or to command another device to take action. In order to learn what the outside temperature is, we would use the service called ReadProperty which allows the reading of properties within an object. In fact this service is the one service that all BACnet devices must process. For example, the BAS Remote's point 2 knows the outside temperature. Another device functioning as a client makes a ReadProperty request to the BAS Remote which functions as the server. The request is made to the Present_Value property in the Analog Input, Instance 2 object. The BAS Remote receives the request and executes the command by providing the information.

A *confirmed* service is one where a service request is initiated and a response with data is expected. An *unconfirmed* service expects no reply. Depending upon the capabilities of a device, it may be able to initiate a confirmed service request or respond to a confirmed service request. Some devices can do both. The same applies to unconfirmed services.

There are 38 possible service requests and they are grouped into five categories (see Table 4).

- Alarm and Event Services (11)
- File Access Services (2)
- Object Access Services (10)
- Remote Device Management Services (12)
- Virtual Terminal Services (3)

Alarm and Event Services	
AcknowledgeAlarm	C
ConfirmedCOVNotification	C
ConfirmedEventNotification	C
GetAlarmSummary	C
GetEnrollmentSummary	C
GetEventInformation	C
LifeSafetyOperation	C
SubscribeCOV	C
SubscribeCOVProperty	C
UnconfirmedCOVNotification	U
UnconfirmedEventNotification	U

File Access Services	
AtomicReadFile	C
AtomicWriteFile	C

Object Access Services	
AddListElement	C
CreateObject	C
DeleteObject	C
ReadProperty	C
ReadPropertyConditional	C
ReadPropertyMultiple	C
ReadRange	C
RemoveListElement	C
WriteProperty	C
WritePropertyMultiple	C

Remote Device Management Services	
ConfirmedPrivateTransfer	C
ConfirmedTextMessage	C
DeviceCommunicationControl	C
I-Am	U
I-Have	U
ReinitializeDevice	C
TimeSynchronization	U
UnconfirmedPrivateTransfer	U
UnconfirmedTextMessage	U
UTCTimeSynchronization	U
Who-Has	U
Who-Is	U

Virtual Terminal Services	
VT-Close	C
VT-Data	C
VT-Open	C

Table 4—38 services are grouped into five categories. Services are confirmed (C) or unconfirmed (U).

The ReadProperty service is included in the Object Access Services group as well as the WriteProperty service necessary to set outputs. These are both confirmed services. With these two simple operations, we can read and write I/O points.

Other interesting services are the Who-Is and I-Am which are unconfirmed services used to "discover" devices. With the Who-Is request, an initiating device is trying to determine the device object identifier; the actual network address of a device, or both. A range of device object identifiers can be sent to restrict the search. In the most basic request, the Who-Is is sent as a broadcast message with no device object identifier restrictions. Since it is a broadcast message, all devices on the network will hear the request, but only those devices that can execute an I-Am service will respond. Each device capable of responding will send out their device object identifier along with some limited information on the device itself including the Vendor Identifier. With BACnet/IP, the response is sent as a broadcast UDP message so the source address can be learned from the response.

The initiating device or any other listening device on the network can then construct a table of device object identifiers versus IP addresses so that future communication would not require this discovery process. Although, the I-Am response logically follows the Who-Is request, an I-Am can be initiated at any time. It is usually sent when a device joins the network and announces to devices on the network that it is present.

A similar set of services exist with the Who-Has and I-Have. In this case, the initiator is trying to determine the device object identifier, network address or both, which contains a particular object identifier or object name. For example, we are trying to learn where we can access outside temperature by entering the Object_Name "Outside temperature." Since we do not know the object identifier, we will simply send out the object name in the Who-Has request. The response to the Who-Has is an I-Have which returns the device object identifier, object identifier, and object name. The network address can be learned as well. In our example, the BAS Remote has the Object_Name "Outside temperature" so it would respond with a I-Have message with the device object identifier of 2749, the object identifier Analog Input, Instance 2, and the object name "Outside temperature." Although an I-Have is the proper response to a Who-Has, an I-Have can be initiated at any time without a Who-Has.

With 38 possible services, discussing each service would make for a lengthy discussion. However, the concepts are the same as with the simple ReadProperty and WriteProperty services. As we will learn later, your simpler devices are only required to implement a fraction of the available services.

Summary

Although object modeling appears complex and awkward, it is the modern method of making devices "network visible." Objects are used to describe physical devices and the types of objects are defined in the BACnet standard. Objects have properties and the property list varies with the object type. In order to use objects to our advantage, devices must be able to provide services. The types of services are also defined in the BACnet standard. Objects, object properties, and services begin to define BACnet-compliant devices.

References

ANSI/ASHRAE Standard 135-2004 BACnet—A Data Communication Protocol for Building Automation and Control Networks, American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc.

Direct Digital Control of Building Systems Theory and Practice, H. Michael Newman, John Wiley and Sons, Inc., 1994

http://www.polarsoft.biz/langbac.html, **The Language of BACnet—Objects, Properties and Services**, Bill Swan, Alerton Technologies, Inc.

http://www.ccontrols.com/pdf/TD0403000D.pdf, Building Automation System Remote, Contemporary Controls

CONTEMPORARY CONTROLS®
www.ccontrols.com