

BACnet Technical Overview

An Introduction to BACnet™

BACnet stands for **B**uilding **A**utomation **C**ontrol **n**etwork. A data communication protocol developed by ASHRAE, BACnet is the "ANSI/ASHRAE standard 135-2001" and the international standard "ISO 16484-5". Its purpose is to standardise communications between building automation devices from different manufacturers, allowing data to be shared and equipment to work together easily.



BACnet is not a thing you can buy. BACnet is literally a book that defines methods which a manufacturer can use, if it chooses to, to make systems which can be interoperable with other BACnet systems in various ways. Owners and specifiers can also use BACnet as a tool for specification of interoperable systems.

BACnet does not replace the need for specifying what one wants or needs. It simply provides some standardized tools to help enable the creation of systems that can interoperate.

BACnet is not specific to HVAC, but is intended to apply to all types of building systems: fire, security, lighting, HVAC, elevators and so forth. The existence of BACnet does not mean that all of these options are automatically available today. But it does provide a context which could enable the creation of those systems if the marketplace indicates a demand for interoperable products.

BACnet achieves these goals by defining a generalized model of how automation systems work, and how to describe the information that they contain, and how to describe standard methods that one device may use to ask another device to perform some desired action.

BACnet works by dividing the problem of interoperability into three distinct areas, and by defining methods and standards for implementing each.

All information within an interoperable BACnet device is modelled in terms of one or more information objects. Each object represents some important component of the device, or some collection of information that may be of interest to other BACnet devices.

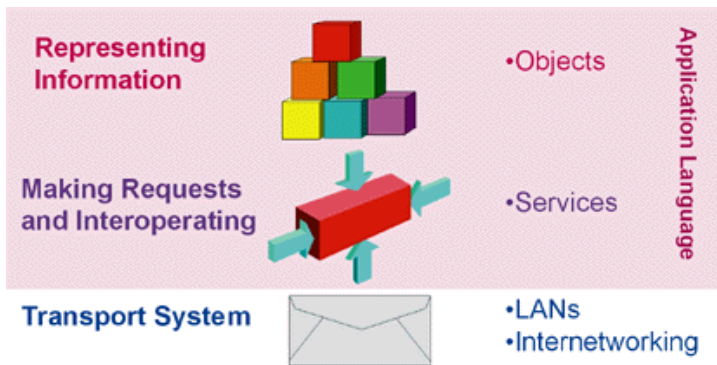
BACnet devices ask each other to perform services. For example, a device to which a temperature sensor is attached, may perform the service of reading the temperature and providing this information to another device that needs it. The model of objects and services is realized by encoding messages into a stream of numeric codes that represent the desired functions or services to be performed. The "language" of this encoding is

TOTAL CONTROL

common to all BACnet devices. BACnet devices actually exchange information and do things by sending and receiving electronic messages containing this coded language.

BACnet provides flexibility by allowing multiple types of transport systems to be used to convey these coded messages between devices. The transport system uses different types of electronic messaging standards and methods to convey coded messages. Even though different transport methods are used, the coded message content remains the same. This philosophy allows the designer or specifier to choose the most cost-effective transport method for a given application.

So How Does BACnet Work?



BACnet works by dividing the problem of interoperability into three distinct areas, and by defining methods and standards for implementing each.

All information within an interoperable BACnet device is modelled in terms of one or more information *objects*. Each object represents some important component of the device, or some collection of information which may be of interest to other BACnet devices.

BACnet devices ask each other to perform *services*. For example, a device to which a

temperature sensor is attached, may perform the service of reading the temperature and providing this information to another device which needs it.

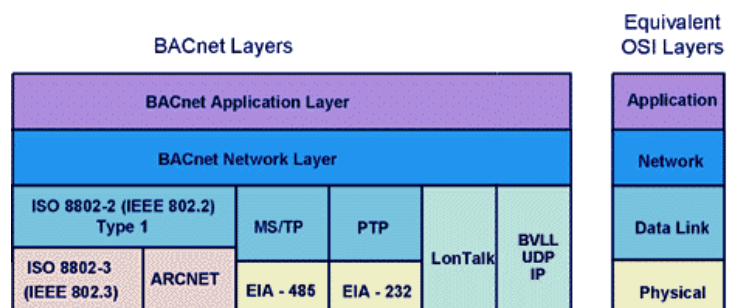
The model of objects and services is realized by encoding messages into a stream of numeric codes which represent the desired functions or services to be performed. The "language" of this encoding is common to all BACnet devices. BACnet devices actually exchange information and do things by sending and receiving electronic messages containing this coded *application language*.

BACnet provides flexibility by allowing multiple types of *transport systems* to be used to convey these coded messages between devices. The transport system uses different types of electronic messaging standards and methods to convey coded messages. Even though different transport methods are used, *the coded message content* remains the same. This philosophy allows the designer or specifier to choose the most cost-effective transport method for a given application.

The critical thing to keep in mind is that BACnet allows you to use multiple types of transport mechanisms for electronic messages, but the contents of the messages are the same language.

BACnet's Collapsed Architecture

BACnet makes use of a subset of the ISO Open Systems Interconnection (OSI) model architecture. The figure shows the correspondence between BACnet components and their position in the OSI model. Presentation, Session and Transport layer functionality is implemented in BACnet Application layer functions, or eliminated entirely as it is not needed for BACnet applications.



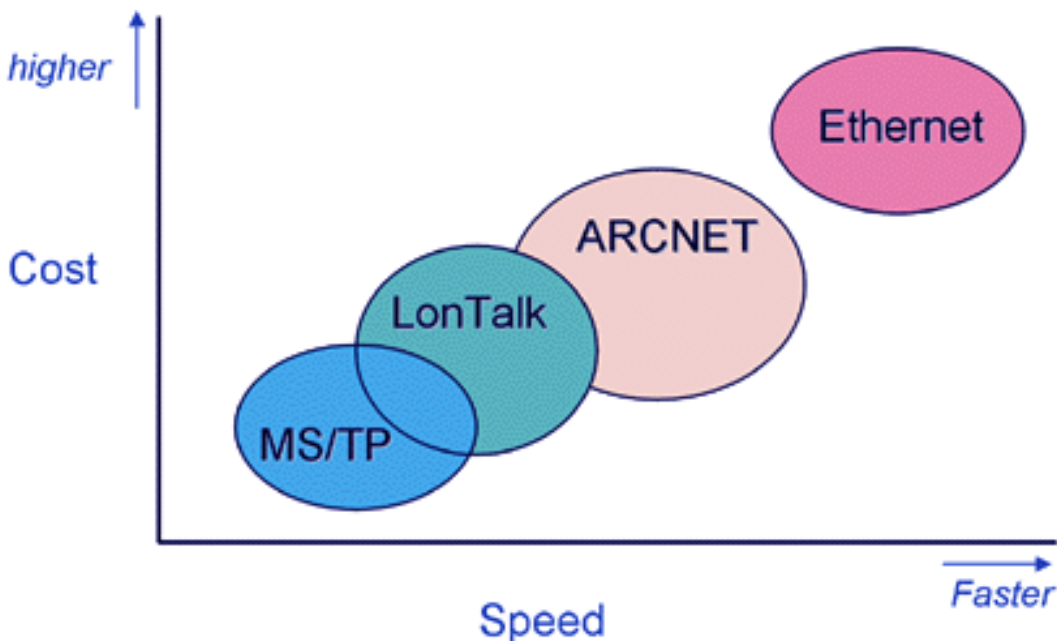
TOTAL CONTROL

BACnet makes extensive use of existing international and national standards for data link and physical layers. In each case where a unique-to-BACnet component appears, it is justified because of the lack of any suitable standard for that component. For example, PTP is a point-to-point protocol unique to BACnet. This protocol provides important features for dialled communications which are critical to many BACnet implementations. MS/TP provides a robust and low cost token-passing LAN which can be implemented without special hardware on single chip microcomputers. The BACnet Network Layer provides special functionality and a unique collapsible internetworking header which provides efficiency in lower performance networks such as PTP, MS/TP and LonTalk.

BACnet/IP encapsulates network and application layer messages within a unique BACnet virtual link layer (BVLL) which is in turn encapsulated within UDP/IP frames allowing BACnet messages to be used within IP based infrastructures.

BACnet Transport Options

With the exception of PTP, which is unique to dialled and point-to-point communications, BACnet generally provides the ability to choose the most appropriate trade-off between cost and performance in the transport mechanism.



As the diagram shows, there is a fairly clear division of increasing cost for increasing performance between the various BACnet transport options. There are certainly areas of overlap, where in some circumstances it may be possible to use several different transports to accomplish the same job, and where the cost/performance ratio may not always be the same. There are, for example, some circumstances where ARCNET can be less costly to implement for the same performance than LonTalk.

The implication of this may be important to specifiers in some circumstances. You may, for example, be able to include more bidders in a competitive situation, due to the increased flexibility in specifying. The flip side of this flexibility is that generally the different LAN types are highly incompatible without the addition of extra devices, such as routers. So one side effect of allowing mixed networks, that include more than two LAN types, is the requirement for providing router devices to couple the networks together.

You can reduce the impact of this by specifying particular LANs to be used. However, this may have the unintended side effect of eliminating some vendors from competing, since not all vendors support all of the LAN types.

Internetworking

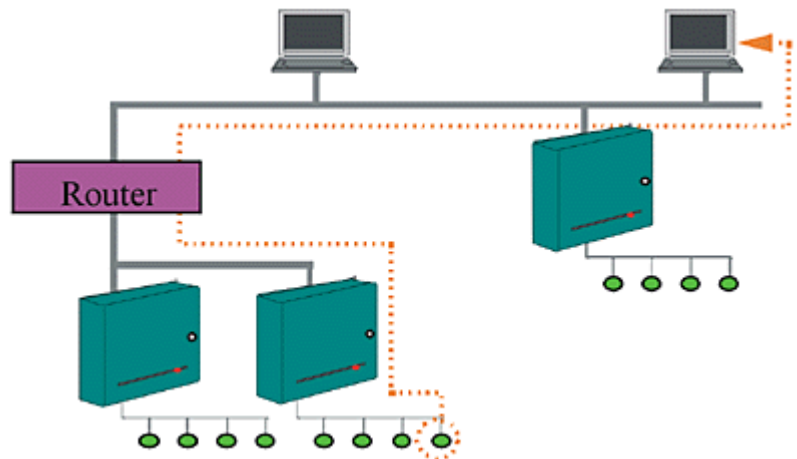
Simply put, internetworking is required whenever we need to couple dissimilar LAN technologies and control the traffic exchanged between them.

There are various situations where this occurs. Let's look at each one . . .

One of the jobs of the Network Layer is to provide a means for routing BACnet messages between logically different networks.

In the figure shown, the upper network segment has several workstations and a field panel which regularly exchange information. The lower network has several field panels which communicate with each other for control and monitoring purposes. So normally, there is no traffic between these two segments. By isolating them onto separate network segments, the upper and lower devices need not bother each other with overlapping traffic.

In those instances when, for example, one of the workstations needs some information from a device on the lower network, it can send a message which contains extra routing information which is used to convey the remote destination of the message. In this case, a special device is required which is called a router. A router is a BACnet device which couples two or more network segments together and passes messages back and forth, only when necessary. The router may bridge together two of the same type of LAN segments, such as Ethernet to Ethernet, but more typically routers also serve the function of coupling different LAN types together, for example Ethernet to MS/TP.



One very important role for routers is in dial-up communications. Dial-up is a special case for several reasons.

First, in the BACnet view the two networks on either side of a dial-up connection are separate logical networks. The router which initiates the dial-up through a modem, establishes a logical connection with the answering router. Once the connection is established, the two routers together behave like a single router in a hardwired connection. For this reason, in BACnet such dial-up routers are called half-routers.

It is possible for a router to be both a hardwired full router, and a dial-up half-router at the same time.

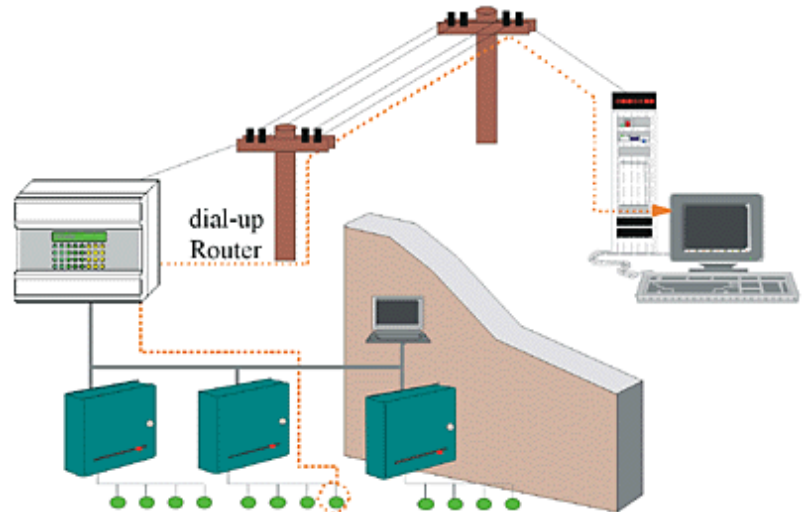
The dial-up router has a fairly complex job. Besides establishing and maintaining a connection, and the normal tasks of a full router once connected, the router must also watch for circular connections. This can occur when a network has more than one dial-up router connected to it. It is theoretically possible for network A to have two simultaneous connections with network B through two dial-up routers on each network. The routers can detect this situation and force one connection to be closed. This prevents "broadcast storms" which are streams of messages being forwarded endlessly across both networks.

Even if traffic is not an issue for your BACnet system, there are often compelling economic reasons to consider sharing LAN infrastructure. In most larger buildings, universities and campus office parks, there is often a substantial investment in LAN technology to link the computers in various buildings together for MIS and administrative functions. Typically Ethernet is used for this type of network. The building automation system with BACnet can take advantage of this infrastructure and use the same LAN for some amount of BAS

traffic, without having to incur the substantial costs of installing and maintaining the LAN wiring, repeaters, bridges, routers etc. In these cases, other types of BACnet LANs can be coupled into the "corporate or university backbone" LAN with a router.

Other than new buildings, or total retrofits, most facilities which embrace BACnet will have the problem of older systems which are already in place. How can these older systems migrate or integrate with BACnet?

The answer is to use a special device called a gateway. Unlike a router, which sends and receives BACnet messages over different LANs, a gateway must potentially not only deal with different LAN technology, but also different concepts and approaches in the application side of the message. With BACnet routers, only the LAN technology changes, but the messages themselves are the same: they always use the BACnet Application Layer and its concepts.



So the gateway must potentially do a very difficult job. Some of the ideas and concepts in the proprietary system, may have no straightforward equivalent in BACnet, and of course the reverse is also true. The gateway is not just a LAN translator, but it is also a concept translator.

One common approach to gateways is that each proprietary system is viewed as an "island" with the gateway being the only bridge to that island.

Using Existing LAN Infrastructure

One of the advantages to BACnet's transport flexibility is the option to utilise existing LAN infrastructures. While traditionally many building automation systems have used dedicated communications wiring, BACnet enables another option which in many cases has compelling financial advantages.

Since BACnet is the international standard "ISO 16484-5" and its data follows the ISO Open Systems Interconnection (OSI) model architecture there is usually little reluctance by a building owner's IT Manager to allow BACnet on their existing IT networks.

Because BACnet can use the international standard ISO8802-3 ("Ethernet") as a transport medium, in many applications BACnet devices can be added to an already existing Ethernet LAN. When this is done, the costs associated with creating a large multi-building LAN, just for the building automation system, can be eliminated. The cost of wiring, routers, repeaters, bridges and maintenance which would normally be a burden can be eliminated by using a portion of the bandwidth on the existing LAN for automation communications.

BACnet/IP can also take advantage of a client's existing Wide Area Networking, which can take the form of a secure Extranet or Intranet. This will mean estate-wide, fast, comprehensive facility management without the need of any new, dedicated or high-speed telephony or internet access.

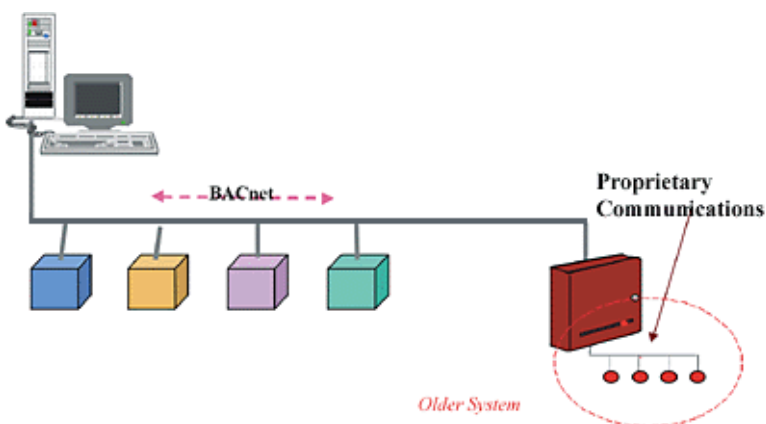
Experience has shown that properly implemented automation systems, based on BACnet, do not place a substantive additional burden on existing LANs, since the nature of BACnet communications is generally intermittent.

Gateways

Other than new buildings or total retrofits, most facilities which embrace BACnet will have the problem of older systems which are already in place. How can these older systems migrate or integrate with BACnet?

The answer is to use a special device called a gateway. Unlike a router, which sends and receives BACnet messages over different LANs, a gateway must potentially not only deal with different LAN technology, but also different concepts and approaches in the application side of the message. With BACnet routers, only the LAN technology changes, but the messages themselves are the same: they always use the BACnet Application Layer and its concepts.

So the gateway must potentially do a very difficult job. Some of the ideas and concepts in the proprietary system, may have no straightforward equivalent in BACnet, and of course the reverse is also true. The gateway is not just a LAN translator, but it is also a *concept translator*.



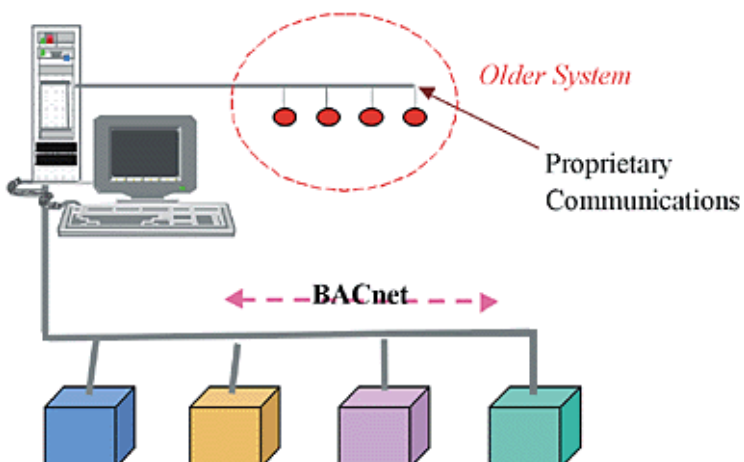
One common approach to gateways is that each proprietary system is viewed as an "island" with the gateway being the only bridge to that island.

With BACnet, it is also possible, and quite normal, to have gateways which interface to proprietary systems at lower levels. For example, a BACnet gateway which interfaces directly to field panels on a peer-to-peer network. Or a gateway to unitary controllers at a very low level.

The ability to provide BACnet gateways at every level is a unique and robust feature of BACnet.

Over the next few years, as BACnet systems and specifications become more common, many vendors will choose to offer these types of gateways to their existing proprietary systems. This approach can be a cost effective solution to wholesale replacement of existing devices with BACnet counterparts.

One of the easiest ways to integrate older systems with newer BACnet-based systems is to use a gateway. The gateway communicates with the older system in its proprietary protocol, and performs the usually complex job of mapping BACnet concepts into equivalent concepts in the older system.



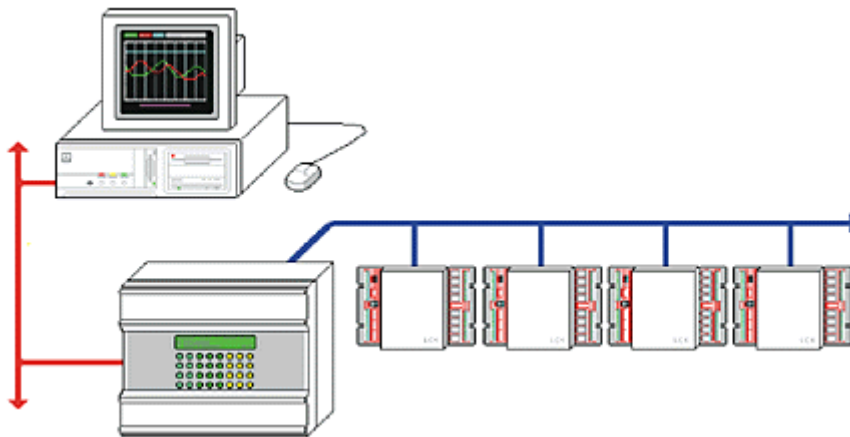
Mapping means creating objects inside the gateway which make the points on the older system appear to be BACnet objects, and behave as standard BACnet objects behave. In some cases, gateways might implement new object types which more closely represent the existing system, rather than try to behave as BACnet standard objects. Other gateway designers may choose to emulate those features required in BACnet objects which are not present in the older system.

A similar variation on this approach is to have a host PC capable of communicating

TOTAL CONTROL

both in BACnet, and in the proprietary communications of the older system. In this case, the host PC software acts as the gateway. You should be wary of this approach and be sure you know what you're getting. In some cases, vendors may implement this scheme so that the host graphics software on the PC is able to look at the old system as well as the BACnet side. But the question is: can the other BACnet devices interoperate with the old system?

This may not be a requirement in your facility. However, if you expect BACnet devices to be able to interoperate with older devices, for example reading temperatures or other sensor values, or adjusting setpoints in the old system, then it is important that the gateway software in this arrangement supports those capabilities.



You can readily tell this by examining the PICS for the gateway.

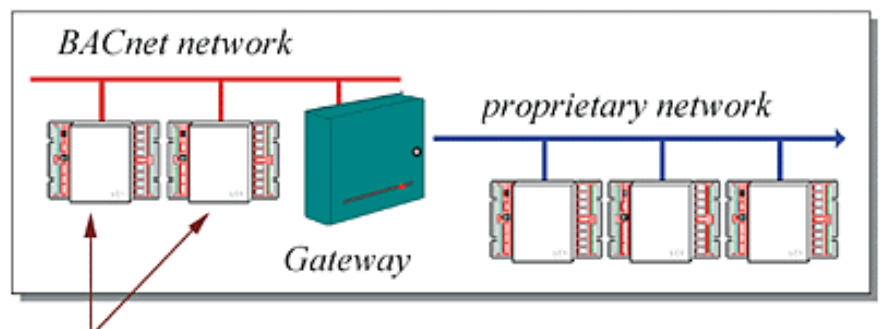
Of course, it isn't always necessary to have a gateway to the entire existing system. In some cases it may be desirable to only have gateways into selected portions of the old system, for example to a central plant control system.

When specifying or buying a BACnet gateway, it is important to keep several points in mind. As with most things, you won't get anything you don't ask for.

In the case of a gateway to an existing system, the most important question is *how much of the old system is visible through the gateway?* You may not require complete access to every point in the existing system from the BACnet side. However, a typical new BACnet owner wants to replace the existing system front end with a BACnet one, for forward expansion in the future. As such, it may be important to be able to operate all or most of the existing system from the BACnet front end. If the gateway implementation is incomplete, this may be difficult or impossible.

How closely does the gateway model existing points and functions as BACnet objects? Does the gateway emulate BACnet behaviours, or simply provide proprietary objects equivalent to the existing system? Does the gateway allow you to control and change properties on the old system? Are all the changes stored only in the gateway, or do they propagate all the way into the existing system's controllers? This could be important in a distributed system, if the gateway fails, do the new changes continue to operate?

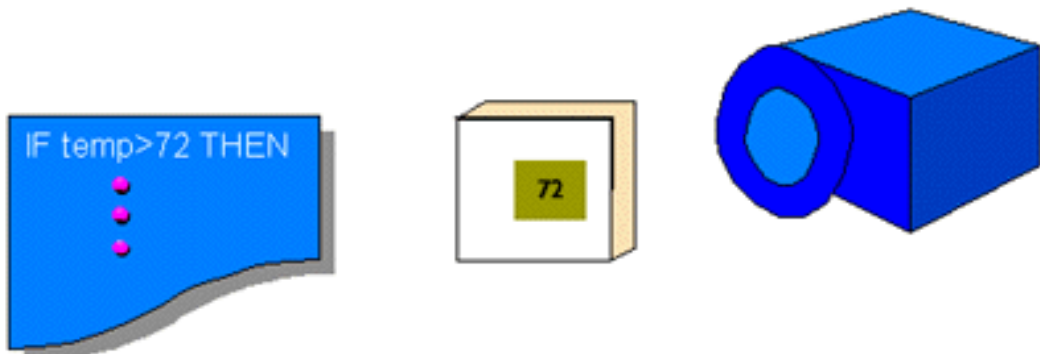
How much can the old system be expanded after the gateway is installed? If you add a new field panel, for example, to the old system does the gateway allow you to access it? Some vendors may require you to pay substantial upgrade fees for this type of expansion, so it is best to ask in advance!



TOTAL CONTROL

Objects

All information in a BACnet system is represented in terms of objects. An object is an abstract concept that allows us to talk about and organise information relating to physical inputs and outputs, as well as non-physical concepts like software, or calculations.



Objects may represent single physical "points", or logical groupings or collections of points which perform a specific function. For example, an object might represent a physical input device like a temperature sensor or thermostat, or an output device like a fan or pump or valve position. Objects can also represent non-physical concepts like program logic, schedules and historical data.

All objects in BACnet provide a set of properties that are used to get information from the object, or give information and commands to an object. You can think of an object's properties as a table with two columns. On the left is the name or identifier for the property, and on the right is the property's value. Some properties are read-only meaning that you can look at the property value, but not change it. Some properties can be changed (written).

Here is an example of a temperature sensor, which might be represented as a BACnet Analogue Input object. The example shows a few of the properties that might be available with this object, although in practice there would be many more properties than those shown.

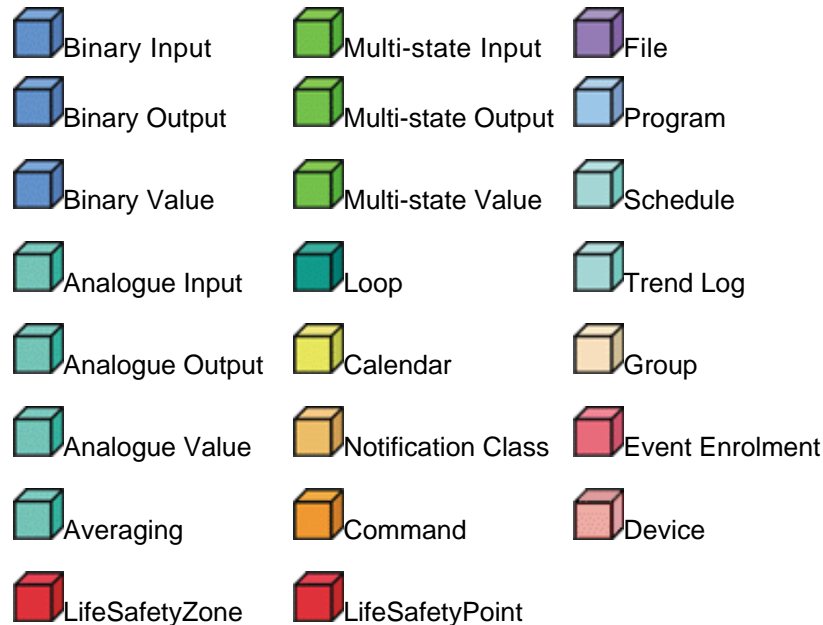


Object_Name	SPACE TEMP
Object_Type	ANALOG INPUT
Present_Value	72.3
Status_Flags	Normal, InService
High_Limit	78.0
Low_Limit	68.0

The object has a name property ("SPACE TEMP") and an object type (ANALOGUE INPUT). The Present_Value property tells us what the temperature sensor is reading at this moment (72.3 degrees). Other properties show us other information about the sensor object, such as whether it appears to be functioning normally, or High and Low Limits for alarming purposes.

TOTAL CONTROL

Although there are many potentially useful object types that might be found in building automation, BACnet defines 23 standard object types in some detail. A BACnet standard object is one whose behaviour, in terms of which properties it provides and what they do, is defined in the BACnet standard.



This set of standard objects represents much of the functionality found in typical building automation and controls systems today. BACnet devices are only required to implement the Device object. Other objects are included as appropriate to the device's functions.

Each of the standard objects in BACnet defines a set of required properties and a set of optional properties. Required properties must be implemented for each object of a given type which purports to be a BACnet standard object of that type. An optional property does not have to be implemented, but if the vendor claims that it is implemented, then it must behave as the standard describes.

Here is an example of a few of the required and optional properties for an analogue input object.

It is also possible, and expected, that some vendors will implement their own non-standard properties for some objects. BACnet allows vendors to implement any number of non-standard properties, with whatever behaviour the vendor chooses.

It is also possible, and encouraged, for vendors to implement their own additional object types. These non-standard object types may include whatever properties the vendor chooses.

Whether an object type is non-standard or not and whether a property is proprietary or not, the object property is read or written in the same manner. All you need to know is the existence of a property and what its purpose is, and you can use it just like any of the standard properties. This key fact allows vendors to extend BACnet and add functionality into BACnet arbitrarily into the future, without ever changing the standard itself.

<i>required</i>	Object Name	SPACE TEMP
	Object Type	ANALOG INPUT
	Present Value	72.3
	Status Flags	Normal,InService
<i>optional</i>	High Limit	78.0
	Low Limit	68.0

TOTAL CONTROL

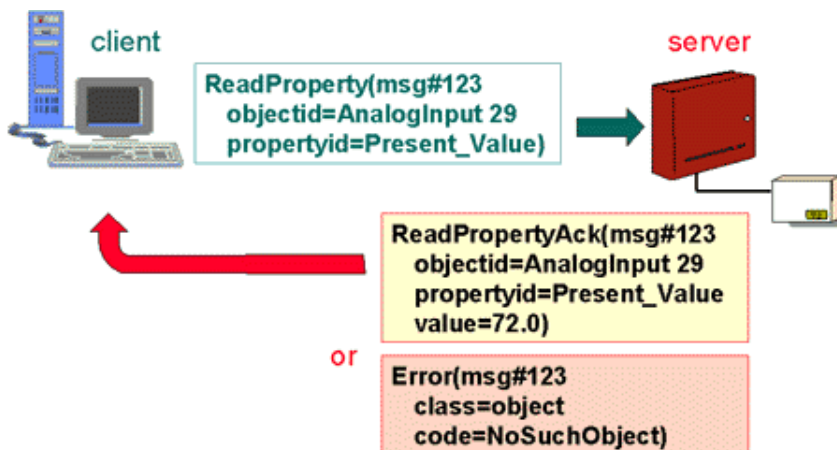
As new technology and techniques are invented, new object types can be used to represent the key parameters or information from and control of those objects. Innovation can proceed without interference from BACnet!

It is important to understand that non-standard objects and/or properties are not a bad thing. In fact, this feature may be the single most important aspect of BACnet because it allows new functionality to be incorporated into a vendor's product without requiring a change to the standard, or anyone's "permission", and without breaking existing implementations.

Having said this, it is also important to note that not all implementations of non-standard properties and objects will be useable by all BACnet Systems automatically. In particular, only properties with primitive data types such as booleans, integers, character strings and so forth, can be assured a wide range of interoperability.

Services

Services allow one BACnet device to do something on behalf of another BACnet device. For example, if one device needs to know a temperature, it can read the Present_Value property of the analogue input object which represents that temperature sensor. The asking device uses a ReadProperty service to do the asking. BACnet defines several broad categories of services. Object Access services allow devices to read and write properties, create and delete objects, manipulate lists of data and search for particular objects or properties. Device Management services let devices be controlled remotely, for example asking a device to stop communicating for a while, or asking peer devices to identify themselves, or find out if a device requires reloading and so forth. Alarm and Event services allow devices to communicate alarms and changes of state, and other exception conditions. File Transfer services allow devices to send or receive information in bulk form, for example historical trend data or downloading control programs.



Now let's look at a typical service, the ReadProperty service. This is one of the most basic services in BACnet and it is used by one BACnet device to ask another BACnet device to provide the value of one of the properties of a specific object.

You can see that the ReadProperty request requires both an Object Identifier and Property Identifier to specify which object and which property are to be read. We also provide a message number which is known as the *Invoke ID*. The reason for this is that BACnet allows us to send multiple messages

before the first reply is received. In such a case, it is possible that the replies will arrive in a different order from the requests, as some requests may be easier to fulfil sooner. The replies include the original message number thus allowing the reply to be matched with the outstanding request.

There are two possibilities. Either the service will succeed and return the requested value, or the service may fail. For example, there may be no such object defined in the target device.

So this diagram captures both the request and its parameters, as well as the possible types of replies and their parameters. Taken altogether this fully defines the service. BACnet includes definitions of many different kinds of services as mentioned earlier.

Conformance to BACnet (The Old Idea)

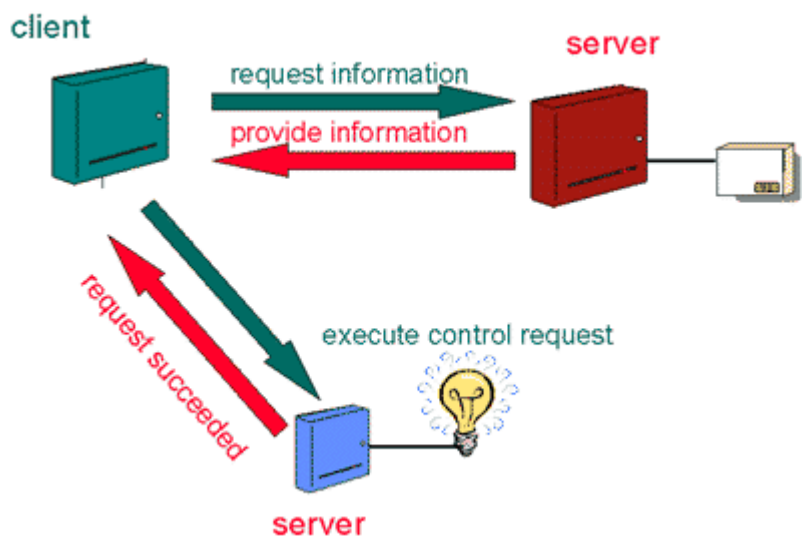
The original BACnet standard from 1995 defined six classes of conformance. Each class specified a set of application services that must be supported and indicates for each service whether the device must be able to initiate the service (client) or execute the service request (server). Some conformance classes also required the support of specific BACnet standard objects. The conformance classes were numbered 1-6 and are hierarchical. Conformance to a class greater than one requires support of all of the requirements of the lower-numbered classes.

BACnet conformance classes were a statement about the number of BACnet features that a given device chose to implement. A Class 1 device, for example, only needed to be able to receive and execute a ReadProperty BACnet service. A Class 2 device also had to be able to receive and execute a WriteProperty BACnet service etc. The problem with these classes is that they do not reflect how deeply a given device implements BACnet, and which of the hundreds of BACnet options are available. So conformance classes were basically of little value.

In addition to conformance classes, the original standard specified a concept called *functional groups*. A functional group was a set of BACnet services and/or objects that presumably would be needed if a device was to support a certain function. For example, a device that understands time and date, and can receive and execute the TimeSynchronization service, and provides the local time and date as properties of its Device object, as said to provide the "Clock Functional Group."

Conformance to BACnet (The New Idea)

Regrettably, the combination of standard object types, conformance classes and functional groups proved to be too obscure and unwieldy for specifiers and owners. As a result, one of the first projects undertaken after the release of the BACnet standard was to revamp the whole idea of how conformance to BACnet should be described and tested. The breakthrough concept was developed by Mike Newman from Cornell University. He proposed the definitions for what he called **"BACnet Interoperability Building Blocks" (BIBBs)**. The idea was to break the problem of interoperability down into common types of functions, intuitive to specifiers and owners, and then to define a name and a set of simple BACnet requirements for each of these. BIBBs were incorporated into BACnet in Addendum D which has become Annex K of the standard.



The BIBBs fall into five principal areas: data sharing, alarms and events, scheduling, trending and device management. In each of these areas there are a number of kinds of interoperable function one might need to perform. Each function always has two entities involved, normally an "asking device" and an "answering device", although you might prefer to think of these as "client" and "server", or "user" and "provider."

TOTAL CONTROL

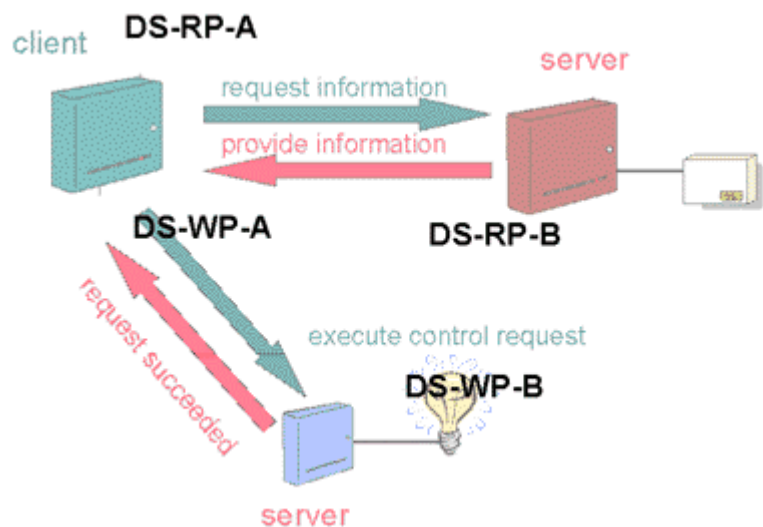
In the simplest example, say that one device has a temperature sensor whose temperature is accessible as a property of a BACnet object. Another device would like to find out this temperature. The device that has the sensor we'll call the "server." In order to interoperate in this manner, the server device must be able to receive a ReadProperty service request and execute it and return a result. The BIBB for this kind of capability is called "DS-RP-B" (**DS** meaning Data Sharing, **RP** meaning ReadProperty, **B** meaning the server device). The device that wants to know the temperature we'll call the "client." The client device must be able to initiate the ReadProperty service request and accept the response when it arrives. The BIBB for this capability is called "DS-RP-A" (**DS** meaning Data Sharing, **RP** meaning ReadProperty, **A** meaning the client or asking device).

So to have an interaction like this, two BIBBs are defined, one for the asking and one for the answering role in the interaction. Of course, other kinds of interoperations are possible and desirable. Let's say the same client has a need to turn on a light. The light is physically controlled by another BACnet device. In this example, the device which controls the light is also a BACnet server. But rather than asking for information our client now wants to request the execution of some action (turn on a light). The client wants to know as a reply simply the request succeeded or failed. In

BACnet this kind of request would usually be performed by writing to or changing the value of a property of an object that represents the light, for example a Binary Output object Present_Value property. The client turns on the light using a WriteProperty service to change the Present_Value to "on." In this case the DS-WP-A and DS-WP-B BIBBs would be required by the client and server respectively.

You can see that each of these operations is clearly defined by a BIBB. In general every interoperation in BACnet has a pair of BIBBs that define the client and server components of the interoperation.

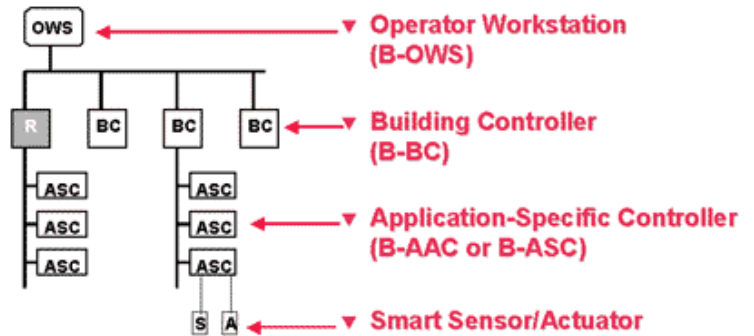
BIBBs have been defined for a large number of interoperable features. Specification of BACnet in large part is accomplished by identifying the BIBBs that are required for each kind of interoperability that is required for a given situation.



TOTAL CONTROL

Device Profiles

While it is possible, and in many cases desirable, to always use a detailed specification of BIBBs, there are many common kinds of interactions that traditionally go together, or are commonly found in certain types of devices. Addendum D also proposed the concept of a **Device Profile**, which is basically a named collection of particular BIBBs. The profile names are intended to convey the typical kind of device for which a given profile is or may be useful. This concept is now Annex L of the standard.



Annex L defines six standard profiles that describe typical collections of BACnet capability in terms of the BIBBs that might be supported by each profile.

BIBB required	B-OWS	B-BC	B-AAC	B-ASC	B-SA	B-SS	
DS-RP-A	✓	✓	✓	✓	✓	✓	Data Sharing
DS-RP-B	✓	✓	✓	✓	✓	✓	
DS-WP-A	✓	✓	✓	✓	✓	✓	
DS-WP-B	✓	✓	✓	✓	✓		
DS-RPM-A	✓	✓	✓	✓	✓		
DS-RPM-B	✓	✓	✓	✓	✓		
DS-WPM-A	✓	✓	✓	✓	✓		
DS-WPM-B	✓	✓	✓	✓	✓		
AE-N-A	✓	✓	✓	✓	✓		
AE-N-B	✓	✓	✓	✓	✓		
AE-ACK-A	✓	✓	✓	✓	✓		
AE-ACK-B	✓	✓	✓	✓	✓		
AE-INFO-A	✓	✓	✓	✓	✓		
AE-INFO-B	✓	✓	✓	✓	✓		
AE-ESUM-A	✓	✓	✓	✓	✓		
AE-ESUM-B	✓	✓	✓	✓	✓		
SCHED-A	✓	✓	✓	✓	✓		
SCHED-B	✓	✓	✓	✓	✓		
T-VMT-A, T-ATR-A	✓	✓	✓	✓	✓		
T-VMT-B, T-ATR-B	✓	✓	✓	✓	✓		
DM-DOB-A	✓	✓	✓	✓	✓	Trending	
DM-DOB-B	✓	✓	✓	✓	✓		
DM-DCC-A	✓	✓	✓	✓	✓	Device and Network Management	
DM-DCC-B	✓	✓	✓	✓	✓		
DM-TS-A	✓	✓	✓	✓	✓		
DM-UTC-A	✓	✓	✓	✓	✓		
DM-TS-B or DM-UTC-B	✓	✓	✓	✓	✓		
DM-RD-A	✓	✓	✓	✓	✓		
DM-RD-B	✓	✓	✓	✓	✓		
DM-BR-A	✓	✓	✓	✓	✓		
DM-BR-B	✓	✓	✓	✓	✓		
DM-DOB-A	✓	✓	✓	✓	✓		
DM-DOB-B	✓	✓	✓	✓	✓		
NM-CE-A	✓	✓	✓	✓	✓		

It's important to keep in mind that profiles are a tool for succinctly describing a collection of interoperability features, not a judgement about which features are more or less important. Some specifiers or owners have made the mistake of thinking of profiles as some kind of minimum recommendation. In fact, in many instances it would be highly undesirable to limit ones choices to only devices that provide the features outlined by standard profiles. In general, specifications should be written around specific BIBBs that define the interoperability desired.

Testing and Certification

The BACnet standard, even with all of its Addenda, does not specify how a given BACnet system should be tested or certified. BACnet defines how particular interactions *should work* and is primarily aimed at designers of devices and systems that have the desire to cooperatively interoperate reliably. All along, it was clear that a method for testing and certifying conformance to BACnet would be needed, and several different but related efforts took place.

One of the key charters for the *Standing Standards Project Committee* SSPC-135 which was formed after the public release of the standard was to create a companion *testing standard* for BACnet. This document details the procedures that must be following to verify that a BACnet device does what it claims to do, and therefore has a chance of interoperating correctly without problems. After several years of effort, this draft standard ASHRAE 135.1P was released for public review in 2000. The procedures themselves are an outgrowth of years of practical testing and experimentation performed by the NIST BACnet Interoperability Testing Consortium. After two public reviews this new standard was published by ASHRAE in June 2003 as ASHRAE Standard 135.1.

Dismayed with the slow progress toward the emergence of a testing agency for BACnet, a non-profit group was formed called the BACnet Manufacturer's Association (BMA). One of their goals was to catalyze the creation of a testing agency, so they formed an independent third party company called the BACnet Testing Laboratory (BTL) with the charter to establish a testing and certifying program for BACnet devices using the draft testing procedures from 135.1P. Today this agency is proactively developing their certification procedures and has accepted applications for devices to be tested and ultimately to receive a BTL "mark." BTL has been performing actual product testing since 2001. The BTL is currently testing a small group of B-AAC devices, and is also capable of testing B-ASC, B-SA and B-SS devices (see Annex L in the BACnet standard for definitions of these acronyms). The BTL is preparing to start testing a small group of B-BC devices sometime in the spring of 2004.



The BACnet Interest Group Europe (BIG-EU) started its own product testing and certification program in 2004. The BMA and the BIG-EU are working together on testing requirements and testing tools, and it is their intention that the two programs will use a common test suite.

It is worth noting that as of November 2003, the BTL is still limited in the scope of its testing and certification program. At present there are no "client side" tests for example. The procedures for doing this kind of testing are defined in 135.1 so it is only a matter of time before the real-world testing can catch up.