



The following article was published in ASHRAE Journal, December 2004. © Copyright 2004 American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc. It is presented for educational purposes only. This article may not be copied and/or distributed electronically or in paper form without permission of ASHRAE.

# <title>XML and Building Automation</title>

By Alex Chervet

Using XML in building automation systems (BAS) has become a hot topic—but do the results live up to the hype? This article explains the fundamentals of XML in nontechnical terms and provides tools to understand how XML related to building automation protocols such as LonMark® and BACnet®.

## What is XML?

XML (extensible markup language) is a text file arranged in a specific format, which is easily transported among computer systems and across operating systems. For example, an XML file cre-

ated on an Apple Macintosh can be sent to a computer running Microsoft Windows or Linux, or to a mainframe, or Unix server or embedded computing platform such as a BAS controller. This transportability makes it easy to understand why

a company's information technology (IT) department and software vendors have embraced XML.

XML uses tags (words surrounded in angle brackets such as <root>) to identify elements. At first glance the angle brackets make an XML document seem similar to an HTML (hypertext markup language) or Web page document. However, HTML documents and XML documents are different and serve different purposes.

---

## About the Author

**Alex Chervet** is product marketing manager at Echelon Corp. in San Jose, Calif.



HTML tells a Web browser how to render or display a document. XML describes the data contained in a document. Consider the following e-mail message.

To: husband@hotmail.com  
From: wife@hotmail.com  
CC:  
Subject: Milk

It's Friday. Don't forget to pick up milk for the weekend.

Coded as HTML this e-mail message might look like this:

```
<html>
  <title>Email</title>
  <body>
    <b>To:</b>husband@hotmail.com<br>
    <b>From:</b> wife@hotmail.com<br>
    <b>CC:</b><br>
    <b>Subject:</b>Milk<br>
    <br>
    It's Friday. Don't forget to pick up
milk for the weekend.
  <br>
  </body>
</html>
```

The effect of this coding causes the document to be rendered in a Web browser with the field labels in bold type.

**To:** husband@hotmail.com  
**From:** wife@hotmail.com  
**CC:**  
**Subject:** Milk  
It's Friday. Don't forget to pick up milk for the weekend.

HTML tags define how to display information and, in this case, where to use the bold attribute, and where to insert line breaks.

Coded as XML, the message might look like:

```
<email>
  <title>Email</title>
```

```
  <header>
    <to>husband@echelon.com</to>
    <from>wife@hotmail.com</from>
    <cc></cc>
    <subject>Milk</subject>
  </header>
  <body>It's Friday. Don't forget to pick
up milk for the weekend.</body>
</email>
```

Notice that the XML tags no longer define a rendering attribute such as bold or underline. Instead, the tags tell us something about the text (data) that appears between the tag pair. The tag `<to></to>` tells us to whom the email should be sent. The tag `<subject></subject>` tells us what the message will be about, and the tag `<body></body>` gives us the content of the message.

What if the text had been in French:

À: husband@hotmail.com De: wife@hotmail.com  
copie:  
Objet: Lait  
C'est Vendredi. N'oubliez pas d'apportez du lait pour le week-end

Most people, even non-French speaking people, would guess from the e-mail addresses that this text is, in fact, an e-mail, but they might not know the sender or recipient, and they might think that the subject line was left blank instead of the Cc field. There is the possibility of error. However, coded as XML, all possibility of error is removed.

```
<email>
  <title>Email</title>
  <header>
    <to>husband@hotmail.com</to>
    <from>wife@hotmail.com</from>
    <cc></cc>
    <subject>Lait</subject>
  </header>
  <body>C'est Vendredi. N'oubliez pas
d'apportez du lait pour le week-end</body>
</email>
```

By examining the XML tags, the sender, recipient, subject and body of the message are obvious — even if you don't read French.

The XML tags describe the data but tell us nothing about how to display the data. How the data is displayed in a Web browser or a word processor is not described in the XML file. The XML only describes the data.

Thus, XML separates data from presentation. This turns out to be extremely useful as we'll see later.

### Converting XML

Because an XML file contains only data, it can be used as the source for a file, a chart, a Web page, a document, or a human/machine interface (HMI). In the IT industry, this is often called repurposing an XML document. Repurposing the XML file into the specific format you want is usually done by applying an extensible style sheet (XSL) transform.<sup>1</sup>

A transform is something that can be applied against some XML to translate it into many types of files that can be used in many different software programs. Using a transform, any piece of XML can be translated into any other text format — including more XML. Let's look at a real-world example.

The following XML was retrieved from a building automation supervisor/gateway widely used with ANSI/EIA709.1-based device networks. This snippet represents a portion of a data log (some modifications have been made to remove product references):

```
<DataLogger>
  <Log>
    <UCPTindex>0</UCPTindex>
    <UCPTfileName>/root/data/log0.dat</UCPTfileName>
    <UCPTstart>2002-10-30T01:15:00-08:00</UCPTstart>
    <UCPTstop>2002-11-10T17:45:00-08:00</UCPTstop>
    <UCPTlogLevel>36.0</UCPTlogLevel>

    <Element>
      <UCPTpointName>NVL_nvoPcValueDif_1</UCPTpointName>
      <UCPTlocation>pml1964</UCPTlocation>
      <UCPTlogSourceAddress>0.0</UCPTlogSourceAddress>
      <UCPTlogTime>2002-10-30T01:15:00-08:00</UCPTlogTime>
      <UCPTvalue>123.6</UCPTvalue>
      <UCPTunit>KW</UCPTunit>
      <UCPTpointStatus>AL_NO_CONDITION</UCPTpointStatus>
    </Element>
  </Element>
</DataLogger>
```

```
<UCPTpointName>NVL_nvoPcValueDif_1</UCPTpointName>
<UCPTlocation>pml1964</UCPTlocation>
<UCPTlogSourceAddress>0.0</UCPTlogSourceAddress>
<UCPTlogTime>2002-10-30T01:30:00-08:00</UCPTlogTime>
<UCPTvalue>124.8</UCPTvalue>
<UCPTunit>KW</UCPTunit>
<UCPTpointStatus>AL_NO_CONDITION</UCPTpointStatus>
</Element>
<Element>
  <UCPTpointName>NVL_nvoPcValueDif_1</UCPTpointName>
  <UCPTlocation>pml1964</UCPTlocation>
  <UCPTlogSourceAddress>0.0</UCPTlogSourceAddress>
  <UCPTlogTime>2002-10-30T01:45:00-08:00</UCPTlogTime>
  <UCPTvalue>122.4</UCPTvalue>
  <UCPTunit>KW</UCPTunit>
  <UCPTpointStatus>AL_NO_CONDITION</UCPTpointStatus>
</Element>
<Element>
  <UCPTpointName>NVL_nvoPcValueDif_1</UCPTpointName>
  <UCPTlocation>pml1964</UCPTlocation>
  <UCPTlogSourceAddress>0.0</UCPTlogSourceAddress>
  <UCPTlogTime>2002-10-30T02:00:00-08:00</UCPTlogTime>
  <UCPTvalue>122.4</UCPTvalue>
  <UCPTunit>KW</UCPTunit>
  <UCPTpointStatus>AL_NO_CONDITION</UCPTpointStatus>
</Element>
<Element>
  <UCPTpointName>NVL_nvoPcValueDif_1</UCPTpointName>
  <UCPTlocation>pml1964</UCPTlocation>
  <UCPTlogSourceAddress>0.0</UCPTlogSourceAddress>
  <UCPTlogTime>2002-10-30T02:15:00-08:00</UCPTlogTime>
  <UCPTvalue>120</UCPTvalue>
  <UCPTunit>KW</UCPTunit>
  <UCPTpointStatus>AL_NO_CONDITION</UCPTpointStatus>
```

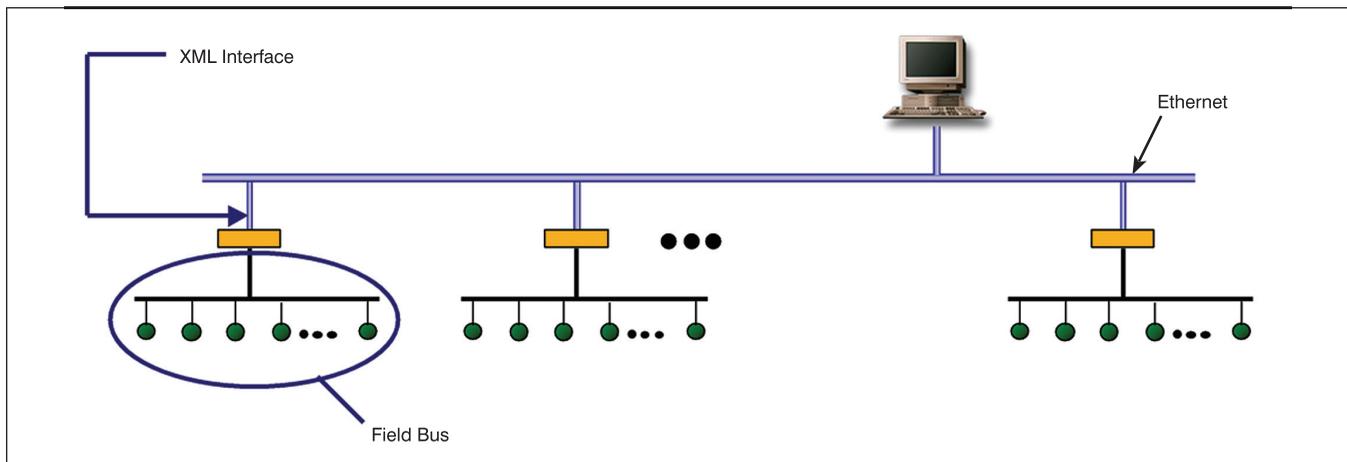


Figure 1: XML is a perfect fit at the gateway/supervisory level.

```

TITION</UCPTpointStatus>
  </Element>
</Log>
</DataLogger>

```

By applying the following simple XSL transform, the above XML converts easily to CSV (a comma-delimited file that can be opened in Microsoft Excel).

#### Transform

```

<xsl:template match="/">
  <xsl:for-each select="DataLogger/Log/
Element">
  <xsl:value-of select="UCPTlocation"/>,
  <xsl:value-of select="translate(substri
ng(UCPTlogTime, 1, 10), '-','/')"/>,
  <xsl:value-of select="substring(UCPTlo
gTime, 12, 8)"/>,
  <xsl:value-of select="UCPTvalue"/><br></
br>
  </xsl:for-each>
</xsl:template>

```

#### Resulting CSV

```

pml1964, 2002/10/30, 01:15:00, 123.6
pml1964, 2002/10/30, 01:30:00, 124.8
pml1964, 2002/10/30, 01:45:00, 122.4
pml1964, 2002/10/30, 02:00:00, 122.4
pml1964, 2002/10/30, 02:15:00, 120

```

For each XML element, the transform did the following:

- Selected only the subset of elements we were interested in (location, date, time, value);
- Inserted commas between values to create the CSV format;
- Converted the date from Uniform Time Code format (UTC) to yyyy/mm/dd format;

- Converted the time from UTC to hh:mm:ss format;
- Removed extraneous data such as alarm status, engineering units, source address and point name;

#### Quick Review

- XML is nothing more than structured text;
- XML is NOT a form of HTML;
- XML describes the data, not how the data eventually is presented.
  - Conversely, HTML describes how data is presented but does not describe the data itself;
- Transforms convert XML into any other format; and
- Because XML is "just text" it:
  - Cannot be owned or controlled by any single company
  - Works with any computer
  - Is human readable making it easy to debug
  - Is optimized to travel over the Web
  - Is applicable to any type of data.

#### HVAC Applications

The protocol wars of the last decade have left many in the industry confused about the best way to connect systems together. Over the course of the next decade, the confusion may increase as HVAC and other building control systems connect to IT systems.

The reality is that XML is an opportunity for HVAC professionals and their companies. The benefits of using XML to connect HVAC or building automation systems is potentially huge, and will benefit companies seeking to increase efficiency, becoming more competitive and offering expanded services.

Consider the example of a building owner or tenant that wants to reconcile actual energy used in a building with a utility bill.

The utility bill is entered into the company's operational systems via electronic transfer from a corporate IT application

or the utility itself. Now, the IT manager needs information from the BAS. Likely, he would prefer an XML file rather than a LonMark or BACnet object, because IT professionals are already familiar with XML.

The IT department's preference for XML and the incredible marketing push from Microsoft, IBM, Sun, ORACLE, and others is not lost on manufacturers of building automation equipment. Major vendors, as well as their suppliers, have announced products or plans for products based on XML.

Okay, so we have this new data format optimized to travel over the World Wide Web infrastructure. Does this mean my occupancy sensor is going to send XML messages to my lighting controller? Probably not.

Remember, XML is a data format — not a protocol. Even if the occupancy sensor did provide information in XML format, the problem of moving the text from the occupancy sensor to the lighting controller still exists.

Protocols such as BACnet (ANSI/ASHRAE Standard 135-2004) and LonTalk (ANSI/EIA709.1) have done an excellent job solving this problem at the device level. For instance, an

ANSI/EIA709.1 packet is optimized to move from device to device over a simple twisted pair using as little as 9 bytes. In those 9 bytes are contained source and destination address, engineering units, the data value itself and a cyclic redundancy check (CRC). All of this is implemented in a \$4 microcontroller that also handles the device electronics — and the solution scales to  $32385 \times 2^{24}$  devices. This solution also allows for low-cost unshielded two-wire twisted pair wiring further reducing total installation cost.

To do the same thing using XML messaging would require a larger processor in the occupancy sensor because processing XML is more intensive than processing control packets. Extra memory is required to host the XML parser and XML messages, which, being text based, are inherently much larger. Although it may be possible to implement such a system over RS-485 or ANSI/EIA709.2 free topology physical layers, the number of messages per second that could be sent is greatly reduced. A complete control system based on device-to-device XML messaging using today's technology would simply be too expensive.

## XML and BACnet®

By **H. Michael Newman**,  
Chairman, SSPC 135 XML-WG

The use of XML also has been a hot topic within ASHRAE. Guideline Project Committee 20, XML for HVAC&R has been working on ways to establish a common data exchange format for describing the characteristics of HVAC&R equipment and controls along with building performance data using XML. Standing Standards Project Committee 135 (SSPC 135), the BACnet committee, has been developing a set of Web services that will enable access to building automation and control system data via the World Wide Web. Addendum c to ANSI/ASHRAE Standard 135-2004, BACnet, recently was in a 45-day public review.

The purpose of Addendum c is to provide a means to integrate building automation and control systems with other enterprise computing applications. Web services provide for computer-to-computer applications many of the same advantages that the World Wide Web provides for human-to-computer information access. Potential uses of the technology include simplifying access to building energy and performance data for inclusion in spreadsheets and other management reports; accessing equipment run-time data for use by maintenance management systems; allowing tenant access to, and control of, space temperature setpoints; coupling of room scheduling with ventilation and comfort control; and many more.

The new addendum is in two parts. The first proposes an Annex N to BACnet that defines the BACnet Web services interface, BACnet/WS. This interface is intended to be "protocol neutral" in that the defined web services can be used with any underlying protocol including BACnet, Konnex, MODBUS, LON or legacy proprietary protocols.

This has been accomplished by defining an application program interface (API) to read and write the common elements of all building automation and control systems such as values, schedules, trend logs, and alarm information using services such as 'getValue' and 'setValue' that use a simple path such as "/Company HQ/Conference Room A/Space Temperature" to define the intended data source. The proposed standard also provides powerful mechanisms for localization where certain types of data such as time, date and numbers can be formatted according to local custom and language. Text names and descriptions may also be accessed according to the local language. Several manufacturers will display Annex N gateway products at the 2005 AHR Expo, February 7–9.

The second part of the addendum contains an addition to BACnet's Annex H, Combining BACnet Networks with Non-BACnet Networks, which prescribes the gateway mapping specifically to and from BACnet messages.

The combined effect of the BACnet/WS annexes is to provide a set of generic Web services that can potentially interface to any building automation protocol, as well as to describe exactly how this interface would work with underlying BACnet systems. ●

---

To keep costs low and to guarantee device-level interoperability, vendors likely will keep the existing device networking protocols that have been optimized for control at the device level, but XML is a “perfect fit” at the gateway/supervisory level where high-speed Ethernet connections and 32-bit processors are becoming the norm.

### **Integration Not Interoperability**

BACnet and LonMark are accepted worldwide as the two protocols for providing interoperability within building control systems. Will XML finally resolve the “war” between LonMark and BACnet systems or will it be a third protocol?

Most likely, the use of XML and new XML-based standards will not end these differences. Expect that suppliers will continue to use LonMark, BACnet, and other open and proprietary technologies for systems that provide specific functions in buildings. Examples of this would include HVAC control systems that use LonMark or BACnet, lighting control systems based on Modbus, LonMark, or DALI, or fire alarm systems and security systems that are based on proprietary protocols. I expect that the protocol used to tie these together could be based on XML and that when information from these systems needs to be shared with business or enterprise systems that XML will also be used.

In short, XML does not replace or compete with other protocols, rather it provides a bridge that ties them together and provides a ready point for interaction with business systems.

But, XML also has much broader uses in HVAC than only for controls. It can be used for building performance data, modeling, equipment catalog information, electronic commerce, etc.

### **Conclusion**

Today we expect access to systems using only a Web browser. The days of installing a custom application for every system we need to access are all but gone. Tomorrow, I will expect all data to be stored as XML. The days of proprietary formats in the IT industry are fading. The building industry, often considered slow to adopt new IT standards, will adopt XML even more quickly than it adopted Web browsers. New XML-based systems will be driven by customer demand and ubiquitous access to rapid application development (RAD) tools.

Remember, existing automation systems have value that XML-based systems simply do not address. Clearly, the correct use of XML is as an integration layer, sitting between corporate systems and dedicated device networks, or as an integration layer between disparate control systems. Used in such a manner, the HVAC industry, integrators and building owners will surely benefit from XML.

### **References**

1. W3C Transformations (XSLT), Version 1.0, Nov. 1999. [www.w3.org/TR/xslt](http://www.w3.org/TR/xslt).

*Advertisement formerly in this space.*