Web Service based integration of building information systems

Veijo Lappalainen and Kalevi Piira
VTT, Finland

Corresponding email: Veijo.Lappalainen@vtt.fi

SUMMARY

The development of new information technologies and standards like Internet and WWW, building control networks, data communication protocols, wireless sensors, mesh networks, product data technologies etc. enables the development of the intelligent, real-time buildings and related new business models in the marketplace of real estate information systems. One of the most promising integration technology in this context is Web Services technologies. The paper first introduces the starting points as well as needs and future challenges for the development. Basic generic ideas of Web Services technologies are then introduced. Main consideration is thereafter focused to the introduction of standard based building automation Web Services approaches: BACnet Web Services and Open Building Information Exchange (oBIX). The basic ideas and concepts of these two methods will be introduced and discussed.

INTRODUCTION

The operation and maintenance of buildings have become dependent on various building automation and information systems (HVAC control, electricity and lighting control, burglar alarm, access control, CCTV), technical building management systems (energy management, FM, BIM) and financial-administrative systems (accounting, personnel administration, renting, space management). In addition, integration of building systems with enterprise systems is becoming more and more important.

These existing building information systems are typically fragmented and isolated from each other forming “automation islands”, which do not communicate together. A consequence of this can be the different systems contain overlapping information, which creates a risk for data integrity. Also because the lacking communication between different systems, it is common nowadays that the information needed by the integrative applications must be collected up manually from different systems and reports.

One of the most important future challenges is the interfacing and integration of the existing applications and to enable the sharing of the information already existing in different systems. New information technologies and standards like Internet and Web, building control networks, data communication protocols, wireless sensors and mesh networks, product data technologies etc. enable the development of the integrated, intelligent real-time buildings and new ICT based building services. Regarding the systems integration, Web Services technologies have become the most promising integration technology.

Standard based open system architectures including object models, interface and service definitions are needed to enable the subsystem and component development. The most important standards official and industry standards will be discussed.
WEB SERVICES AS A GENERIC INTEGRATION TECHNOLOGY

In recent years Internet has become a basic element of data communication network infrastructure. A big challenge still is the integration the systems to enable automatic data transfer between various applications without the need of manual intervention. One solution is the programmable Internet, which fundamental building block is Web Services.

The idea of Web Services (http://www.w3.org/2002/ws/) is to connect applications (and machines) by the same way as e-mail connects people and Web connects people to information. Web Services are platform independent, specifications are based on existing standards and the easy programming of Web Services may affect for application-to-application (machine-to-machine, B2B) communication in the same way as the easy programming of html affects into the breakthrough of the www.

If the objective is performance or reliability, then there are better technologies than Web Services. If the objective is to integrate different loosely coupled components and services, which are decentralized over Internet, then Web Services is the right technology.

The Web Services architecture is built on the Internet and the XML family of technologies. Main technologies are SOAP (communications protocol), WSDL (service description), UDDI (service discovery) and XML technologies (representation of documents) [5]. These Web Services technologies are language, platform, operating system and hardware independent technologies and they provide a standard means of interoperating between different software applications and decentralizing loosely coupled components and services into the Internet. The main Web Services architecture stacks are illustrated in Figure 1.

Unlike many other approaches to integration, Web Services connect devices simply by agreeing on message format, rather than forcing disparate devices to run identical software. Additional functionality for building secure, reliable, transacted Web Services— are possible through a set of recently published specifications (SOAP messaging formats) known as the Advanced Web Services architecture. The ubiquity of HTTP and the simplicity of SOAP make them an ideal basis for implementing Web Services that can be called from almost any environment. Web Services runtime communication is shown in Figure 2.
WEB SERVICES AS A INTEGRATION TECHNOLOGY OF BUILDING AUTOMATION AND INFORMATION SYSTEMS

Creation of the new Web Services Interfaces into building automation systems has been motivated by recognition that Web Services will be emerging as the predominant technology for the integration of a wide variety of enterprise information systems. The objective has been therefore to define standard means of using Web Services for integration and exchange of facility data from disparate data sources, including building control networks, with a variety of business enterprise applications.

The standardization efforts for the application of XML/Web Services to Building Automation started under the umbrella of CABA (http://www.caba.org) in April 2003, when the first meeting of the CABA XML Guideline Committee was held. In that time the different interest groups like BACnet, LON, Modbus, etc. societies worked together. At the end of 2003 it became clear that rather than a lightweight guideline an official or industrial standard was needed. Also it became clear CABA was not the right umbrella organization to bring the standard into the practice.

At that time the name oBIX (Open Building Information Xchange) was given to CABA’s XML/Web Services Guideline committee and discussion started about which standardization organization oBIX should be joined. There were finally two main alternatives: OASIS and ASHRAE. LonWorks and more business oriented members were targeting to IT like standard and were supporting OASIS. BACnet and HVAC oriented members wanted that the new standard should be HVAC and building domain originated and they were promoting ASHRAE, hosting also BACnet standardization.

The discussions took about six months without reaching consensus. The opinions were uncompromising and the consequence was that the CABA’s oBIX committee joined to OASIS but without BACnet society. The BACnet opposition started their own XML/WS standardization process with a new work item in ASHRAE SSPC 135 (Standing Standard Project Committee), in its XML working group (XML-WG).

BACnet/WS standardization progressed in the beginning quite quickly and the draft BACnet/WS standard was published for the first public review in October 2004. The second draft of the standard was published for the public review in March 2006. The final version of BACnet/WS standard [1] was approved by the ASHRAE Standards Committee and by the
ASHRAE Board of Directors in September 2006; and by the American National Standards Institute (ANSI) in October 2006.

The standardization work within OASIS oBIX committee did not start as fast as within BACnet/WS committee. However, several draft versions of the standard were published in Web. The first public review version was published in July 2006. After resolving the review comments OBI X TC approved the first version of the standard in December 2006 [4]. The objective is to get the standard approved by OASIS in early 2007.

BUILDING AUTOMATION WEB SERVICE TECHNOLOGIES

BACnet Web Services

The BACnet/WS standard 135-2004c-1 Adding BACnet/WS Web Services Interface [1], addendum c to the BACnet standard is logically in two parts:

- Annex N - BACnet/WS Web Services interface (normative) to the BACnet standard
- New Clause H.6 - Using BACnet with the BACnet/WS Web Services Interface to Annex H of the BACnet standard (Combining BACnet with non-BACnet networks)

This interface proposed in Annex N is intended to be "protocol neutral" so that the defined Web Services can be used with any underlying protocol including BACnet, LonWorks, Konnex, MODBUS 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" to define the intended data source. An example of such a path would be: "/ABC HQ/Conference Room A/Space Temperature".

The standard 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 using the local language.

The second part of the addendum, Clause H.6, 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 interfac ed to any building automation protocol as well as to describe exactly how this interface would work with underlying BACnet systems.

Implementations of the services shall conform to the Web Services Interoperability Organization (WS-I) Basic Profile, which specifies the use of Simple Object Access Protocol (SOAP), over Hypertext Transfer Protocol (HTTP), and encodes the data for transport using Extensible Markup Language (XML), using the datatypes and canonical encodings defined by the World Wide Web Consortium XML Schema.

Data Model

BACnet/WS interface specification establishes a minimal set of requirements for the structuring and association of data exchanged with a BACnet/WS server.
Any node may have a value. The possible types for a node's value are limited to the primitive datatypes "String", "Integer", "Multistate", "Boolean", "Real", "Date", "Time", "DateTime", and "Duration". Nodes that have a value may also have other attributes related to that value, such as minimum, writable, etc.

An attribute is a single aspect or quality of a node, such as its value or its writability. Every node exposes a collection of attributes. Some attributes are required for all nodes, and some are conditionally required based on the value of other attributes. Some are localizable and may return different values based on an option in a service request.

A path is a character string that is used to identify a node or an attribute of a node. The hierarchy of nodes is reflected in a path as a hierarchy of identifiers arranged as a delimited series, similar to the arrangement of identifiers in URL for the WWW. A path like "/East Wing/AHU #5/Discharge Temp" identifies a node, and a path like "/East Wing/ AHU #5/Discharge Temp:InAlarm" identifies the InAlarm attribute of that node.

To allow for an arbitrary number of logical arrangements of nodes, a single node may logically appear to be in more than one place in the hierarchy through the use of a reference node. Reference nodes may be used to build alternate logical arrangements of nodes since the children of a reference node may differ from that of its referent node.

One of the basic concepts of the data model is Node representing the object of the model. The types of the Node are (the allowable values for the corresponding attribute): Unknown, System, Network, Device, Functional, Organizational, Area, Equipment, Point, Collection, Property and Other.

BACnet/WS includes the following Services: getValue, getValues, getArray, getArraySize, setValue, setValues Service, getHistoryPeriodic, getDefaultLocale, and getSupportedLocales.

Mapping BACnet to BACnet/WS

BACnet/WS interface is intended to be "protocol neutral" and the defined Web Services can be used with any underlying protocol. When applying BACnet/WS standard the underlying protocol has to be mapped to BACnet/WS concept. Table 1 shows an example of the mapping when the underlying protocol is BACnet. In the table BACnet/WS values and value related attributes have been mapped to BACnet Object properties.

<table>
<thead>
<tr>
<th>BACnet Object Type</th>
<th>Value</th>
<th>ValueType</th>
<th>Units</th>
<th>Maximum</th>
<th>Minimum</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accumulator</td>
<td>Present_Value</td>
<td>&quot;Integer&quot;</td>
<td>Units</td>
<td>Max_Pres_Value</td>
<td>Min_Pres_Value</td>
<td></td>
</tr>
<tr>
<td>Analog Input</td>
<td>Present_Value</td>
<td>&quot;Real&quot;</td>
<td>Units</td>
<td>Max_Pres_Value</td>
<td>Min_Pres_Value</td>
<td></td>
</tr>
<tr>
<td>Analog Output</td>
<td>Present_Value</td>
<td>&quot;Real&quot;</td>
<td>Units</td>
<td>Max_Pres_Value</td>
<td>Min_Pres_Value</td>
<td></td>
</tr>
<tr>
<td>Analog Value</td>
<td>Present_Value</td>
<td>&quot;Real&quot;</td>
<td>Units</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Averaging</td>
<td>(varies)</td>
<td>&quot;Real&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Binary {Input</td>
<td>Present_Value</td>
<td>&quot;Boolean&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Binary Output</td>
<td>Present_Value</td>
<td>&quot;Boolean&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Binary Value</td>
<td>Present_Value</td>
<td>&quot;Boolean&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Calendar</td>
<td>Present_Value</td>
<td>&quot;Boolean&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Present_Value</td>
<td>&quot;Integer&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Device</td>
<td>System_Status</td>
<td>&quot;Multistate&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Event Enrollment</td>
<td>Event-State</td>
<td>&quot;Multistate&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Life Safety Point</td>
<td>Present_Value</td>
<td>&quot;Multistate&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BACnet Object Type</td>
<td>Value</td>
<td>ValueType</td>
<td>Units</td>
<td>Maximum</td>
<td>Minimum</td>
<td>Resolution</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------</td>
<td>-----------</td>
<td>-------</td>
<td>---------</td>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>Life Safety Zone</td>
<td>Present_Value</td>
<td>&quot;Multistate&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Loop</td>
<td>Present_Value</td>
<td>&quot;Real&quot;</td>
<td>Output_Units</td>
<td>Maximum_Output</td>
<td>Minimum_Output</td>
<td></td>
</tr>
<tr>
<td>Multistate Input</td>
<td>Present_Value</td>
<td>&quot;Multistate&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multistate Output</td>
<td>Present_Value</td>
<td>&quot;Multistate&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multistate Value</td>
<td>Present_Value</td>
<td>&quot;Multistate&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Converter</td>
<td>Present_Value</td>
<td>&quot;Real&quot;</td>
<td>Units</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schedule</td>
<td>Present_Value</td>
<td>(varies)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Open Building Information Exchange – oBIX**

The basic references for oBIX are the oBIX specification[4] and the official oBIX committee Website [5].

The core of the oBIX standard includes discovery, data access, historical trends and alarming services for HVAC, security, energy management and lighting. The services are autonomous and have a loosely coupled relationship to a central discovery service.

The data service allows clients to read and write oBIX set of primitive data types. This data could be real-time control points such as actuators and sensors, or it could be vendor properties. In addition to explicit client request for data values, the data service supports data subscription in two ways client initiated polling and server initiated callbacks.

Historical data is needed for reporting and analysis within a system for optimizing performance, efficiency and user experience. The oBIX historical data service will be used to query trend data that has been collected by the server as time-stamped value samples.

The alarm service will provide a normalized representation of an alarm event record. It will have the ability to query alarms that have occurred on the server, the ability to subscribe for new alarms that are generated and the ability to acknowledge existing alarms.

**oBIX architecture**

*The oBIX architecture is presented in Figure 3. It is based on the following principles*

- **Object Model**: a concise object model used to define all oBIX information.
- **XML Syntax**: a simple XML syntax for expressing the object model.
- **URIs**: URIs are used to identify information within the object model.
- **REST**: a small set of verbs is used to access objects via their URIs and transfer their state via XML.
- **Contracts**: a template model for expressing new oBIX “types”.
- **Extendibility**: providing for consistent extendibility using only these concepts.

![oBIX Architecture](image-url)
oBIX object model

The oBIX specification is based on a small, fixed set of object types. These object types map one to one to an XML element type. The oBIX object model is summarized in the illustration of Figure 4.

The root abstraction in oBIX data model is the obj element object. Every XML element in oBIX is a derivative of the obj element. Any obj element or its derivatives can contain other obj elements.

oBIX roadmap

A common view within oBIX community is that oBIX 1.0 is just the starting point towards more comprehensive applications and integration with enterprise systems. There are already preliminary ideas how oBIX 1.0 should be developed further. As an illustration a roadmap towards oBIX 2.0 is presented in Figure 5 [6].

Other Web Services approaches – OPC-UA

In addition to BACnet/WS and oBIX maybe the most important other Web Service approach in building industry is OPC Web Services. Its main application domain is industrial automa-
tion, but it is applicable to the building automation as well. However, because it is not build-
ing domain specific it won’t be handled in more detail here. OPC Web Services are based on
the new OPC Universal Architecture, OPC-UA [7], [8].

A big change compared to the earlier versions of OPC that users will notice right away is that
OPC-UA is not based on Microsoft COM/DCOM. The whole world, including Microsoft, is
embracing open, internet-based communication standards. OPC has done the same by basing
UA on TCP/IP, HTTP, SOAP, and XML. Not being tied solely to Microsoft platforms makes
OPC-UA highly scalable and applicable to different HW platforms.

DISCUSSION

A major problem in today’s building information systems is that the building information sys-
tems are fragmented and isolated forming “automation islands”, which don’t communicate
together. The systems can contain overlapping information for their applications because of
lack of communication. Integration of the data of different sources to produce reliable real-
time management information is very difficult. Systems are typically proprietary without
compatibility in core information. These create also dependence from the system vendors.

Web Service technologies described in the paper enable the standard based solution to all
these problems. From the viewpoint of technological development having competing ap-
proaches has positive influence. Negative is that there will still remains automation islands.
In addition, Web Services technology without application area service specification (stand-
ards) is not very valuable. It means that it is important to develop new Web Services inter-
faces into building automation systems.

ACKNOWLEDGEMENT

The results of this paper are based mainly on the work done in co-operation with Fatman Oy
within Eureka E!3083 AutoSpace project.

REFERENCES

   for Building Automation and Control Networks. 135-2004c-1. Adding BACnet/WS Web Services
   Interface. Approved by the ASHRAE Standards Committee on September 29, 2006; by
   the ASHRAE Board of Directors on September 29, 2006; and by the American National Stan-
2. BACnet Website: http://www.bacnet.org/
4. oBIX 1.0. Committee Specification 01, 5 December 2006. OASIS Open Building Information
   Exchange (oBIX) Technical Committee.
5. oBIX Website of: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=obix
6. Toby Considine. oBIX: Are we there yet? Presentation slides. FMOC Summer Meeting 2005
   http://www.nibs.org/FMOC/71305/7_oBIX%20_FM_OC.pdf