TECHNOLOGY ENABLERS FOR NEXT-GENERATION ECONOMIC BUILDING MONITORING SYSTEMS

JAMES SWEENEY JR.
SENIOR RESEARCH ASSOCIATE

CHARLES H. CULP, P.E., PH.D.
ASSOCIATE DIRECTOR

ENERGY SYSTEMS LAB
TEXAS A&M UNIVERSITY
COLLEGE STATION, TEXAS

ABSTRACT

The measurement of a building's electrical and thermal consumption provides the necessary data used to increase energy efficiency. Measurements typically range from hourly to monthly. Monthly data can be used to determine if savings are being maintained. Hourly data provides added detail for diagnostics. Currently, the cost of a complete monitoring system deters use in buildings under 30,000 to 50,000 square feet. Buildings can be optimized using techniques like Continuous Commissioning^{SM} (CC^{SM}) and experience a reduction in consumption ranging from 10% to 40% [1]. Using hourly data has proven to be very effective in maintaining the initial level of savings over an extended period of time [1]. The Energy Systems Laboratory at Texas A&M University (TAMU) has applied CC^{SM} to over 100 buildings and obtained an average savings of 22 percent. Currently, whole-building and sub-metering relies on conventional dial-up based data logging systems. Development of a next-generation data acquisition system is essential to achieve a lower cost for building energy monitoring and analysis. The next-generation system discussed in this paper is a complete redesign. It will be Internet-enabled and secure; take advantage of current advances in smarter sensors, use embedded micro-controllers and mixed signal processors and use Java and XML.

INTRODUCTION

On-going metering and monitoring is an essential part of the CC^{SM} process [1]. Monitoring a building's energy and environmental conditions generally requires the measurement of the building's energy usage and demand, thermal consumption, lighting, temperature, relative humidity and sensitive gases such as carbon dioxide. Hourly intervals are recommended for monitoring, at a minimum.

The material and installation costs of current implementations may run from $5,000 to $20,000 for an installed system. Electronic interfacing and configuration of these systems is also problematic and error prone [2]. Current systems utilize dialup technology, often yielding weekly data with associated long distance charges.

This article discusses building monitoring components and protocols and technologies that will increase the measurement system's integration, control and maintenance.

SYSTEM COMPONENTS

The building monitoring system components may be divided into four subsystems: 1) sensor/transducer, 2) data collection and control, 3) local building server, and 4) master building server. In practice these subsystems are quite distinct, distributed and from multiple vendors. Table 1 lists the subsystems, their functions, their components, inputs and outputs.

Sensors:

Physical variables like electrical current, electrical power, and temperature change the output signal of sensors. Sensor performance must be specified. Generally, items like signal output, environmental ranges of operation, accuracy, linearity and repeatability must be carefully specified. Sensor outputs can be voltage or current signals, which are typically associated with "dumb" or analog sensors. "Smart" sensors usually output digital values in which the sensor electronics may perform calibrations and scaling functions internally. Other sensors like totalizers, can output pulses, which need to be counted and totalized. This next generation system should be designed to accept a range of sensors.

Data Collection and Control

This component is usually called the "logger". The logger controls collection of sensor data and communications. Next generation devices will allow increased data

Subsystem	***Function(s)***	***Components***	***Input***	***Ouputs***
Sensor/Transducer	Convert a physical variable to an electrical signal and possibly a digital value	Sensors of various types and “intelligence”	Monitored physical parameter	Analog and digital signals
Data Collection and Control	Collect data and instrument status, performs local configuration control and communicates externally	Data loggers	Sensor signal(s)	Data communications and control signals
Local Building Server	Aggregate and analyze local building data, upload data to master building server, pass or generate control information to the logger	Small to mid-computer system with a relational database system (RDBMS) and application specific software	Data streams from data collection devices Configuration data from other servers	Database upload to master building system Control data to the logger
Master Building Server	Aggregate building data, manage measurement devices (loggers), analyze data for quality, savings, and unknowns Control network of local servers and loggers	Large computer system with RDBMS and online analytical processing (OLAP) systems	Data streams from data collection devices and/or database uploads	Control signals and reporting

Table 1. Building Monitoring System Components.

checking, validation and analysis to be done at the point of collection.

Local Building Server

The local building server aggregates and analyzes data streams from the data collection and control subsystem for direct action or pre-processing for the master building server. This server also generates or passes control data to the local logger. Local building servers may be installed to handle certain sections of a large building or integrate the whole building. In some cases the local building server may be built into the logger.

Master Building Server

This system primarily handles data from local building servers. Local building servers may be aggregated in geographic or application specific context for analysis. Primary duties include aggregating building data, managing measurement devices (loggers), and analyzing data for quality, diagnostics, and savings. Modern EMCS may encompass all of the subsystems. Simple monitoring systems may only include sensors, and data collection and control subsystems.

In order for these subsystems to interoperate, they must be networked with standard communication protocols. The Internet provides an open communication protocol, Transmission Control Protocol/Internet Protocol (TCP/IP).

COMMUNICATIONS AND INTERFACES

The Internet, supported by TCP/IP, has become the standard for computer-to-computer communications. The Open Systems Interconnect Reference Model (OSI) describes TCP/IP. OSI contains seven layers that define the functions of data communications protocols [3]. They are the physical, data link, network, transport, session, presentation and application layers.

TCP/IP contains four layers, three of which are from the OSI model, plus an additional layer. OSI application, transport, and network layers describe TCP/IP. An additional layer, the Internet layer, is also defined (Table 2). This layer is above the network layer and is the heart off IP [3]. IP is responsible for packet delivery. TCP resides in the transport layer and is responsible for reliable data delivery service with end-to-end error detection and correction [3]. Secure networking of data and control is a principal challenge in distributed building monitoring. TCP/IP has emerged as the standard for Internet enabled data loggers and transducers. TCP/IP is a more reliable and faster communication vehicle compared to traditional dialup communications.

Stack Layer	*Description*
4. Application Layer	Application layer and processes use the transport layer protocols to deliver data
3. Transport Layer	Provides data delivery in the form of the transmission control protocol and the user datagram protocol
2. Internet Layer	Handles routing and transmission of data to the transport layer
1. Network Layer	Routines for accessing physical networks

Table 2. TCP/IP Stack.

TCP/IP and Security

TCP/IP protocol is a packet-switched, digital communications technology designed to reliably transport data files among various computers around the world. Building monitoring and control data is generally security sensitive and the Internet is insecure. Establishing a Virtual Private Network (VPN) for a building monitoring system is a means to secure the information. A VPN provides a secure point-to-point tunnel through the public Internet. These tunnels are created and destroyed on demand. The “tunneling” of data is the repackaging of data from one network to another [4].

The repackaging of data occurs at OSI layers two through five. Link-level (data-link and network) encryption prevents network traffic analysis and attacks. Protocol-level (network and transport) encryption encrypts only data, facilitating traffic analysis. IPSecure (IPSec) is a secure network protocol that acts at the network, protecting and authenticating IP packets between participating IPSec devices, such as routers [5].

The data packets may also be encrypted with sophisticated algorithms in the application layer (seven). Secure Sockets Layer (SSL) protocol is a means to provide privacy and reliability between two communicating applications. SSL operates in the application layer of the OSI model. This protocol enables a server and a client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before the application protocol transmits or receives its first byte of data. [6]. This allows another application to sit on top of SSL. Communications between data collection subsystems and building servers may be secured this way. VPNs also include bandwidth management. Packets may be tagged with priority and time-sensitivity information, allowing traffic to be routed based on its delivery priority [4]. Priority and time-sensitivity packet information is crucial for the building monitoring system to expeditiously transmit and receive data through the building's local network. Standard and open protocols like TCP/IP, IPSec and SSL provide base communication technologies to build the monitoring system on.

Smart Sensors Networking

A “smart” sensor is a sensor with some processing of the physical value sensed and usually has the processed value digitized. A "smart" sensor interface standard has recently become a reality with the emergence of the Institute of Electronics and Electrical Engineers (IEEE) 1451.

IEEE 1451 defines an interface for the connection of sensors and transducers to microprocessors, control and field networks, and data acquisition and instrumentation systems that are network independent [7]. Although the building engineer may not be interested in knowing the details of the protocol standards, it is important to realize that sensors complying with this standard are interchangeable yielding simpler installations and maintenance.

IEEE 1451 Details

The standard is divided into four main parts:

1) transducer to microprocessor communication interface (IEEE 1451.2),
2) networked smart transducer model (IEEE 1451.1),
3) multi-drop distributed system for interfacing smart transducers (IEEE 1451.3 -proposed), and
4) mixed-mode transducer interface (IEEE 1451.4 -proposed).

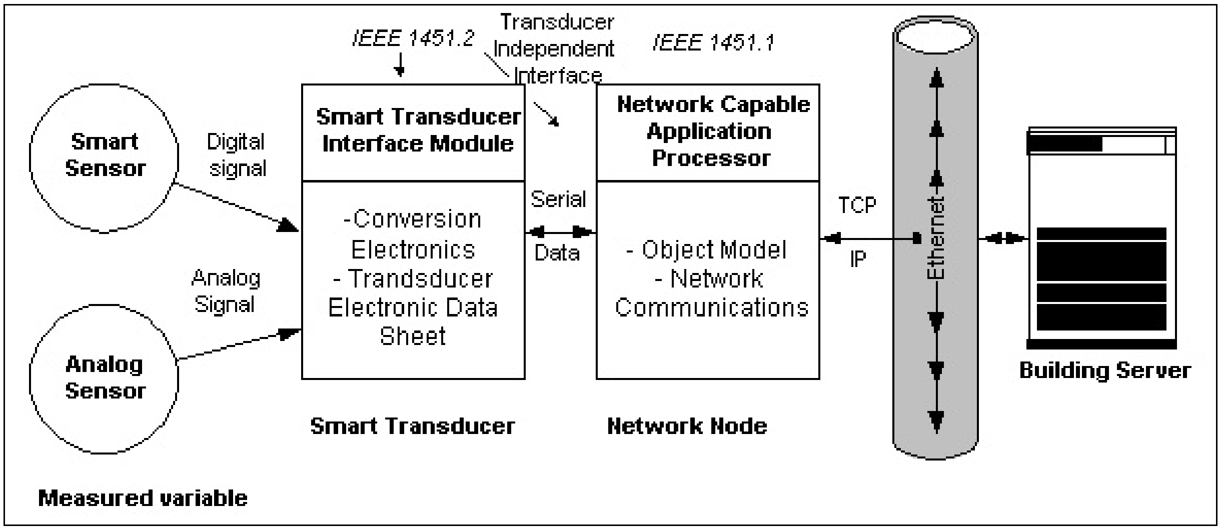

Figure 1. Building Monitoring System with IEEE 1451.

IEEE 1451.1 defines the Transducer Electronic Datasheet (TEDS) and the Smart Transducer Interface Module (STIM). Every transducer that has a TEDS in non-volatile memory contains the type, attributes, operation and calibration of the transducer. This table is stored in the STIM. The mandatory data is 178 bytes [7].

Only two of the eight TEDS fields are required and must remain with the STIM for its lifetime. The remaining six fields are optional [8]. This capability will allow built in diagnostics, which will reduce install costs and on-going maintenance.

The STIM is a networked and intelligent transducer node that supports up to 255 sensors or actuators of various signal mixes. The STIM is connected to a network node called Network Capable Application Processor (NCAP), shown in Figure 2. The STIM transparently communicates with the network via the Transducer Independent Interface (TII), which links the STIM to the NCAP [8].

IEEE 1451 Relevance

The network smart transducer model defines a common object model for encapsulating the transducers interoperability at the application layer. IEEE 1451 provides methods that support network-neutral communication with both publisher-subscribe and client-server mechanisms [9]. This facilitates building monitoring data streams to the NCAP where the data may be stored, pre-processed, and/or pushed-pulled, depending on the application requirements (Figure 1).

Open standards in networking from the Internet to the sensor network interface are enabling secure, distributed building monitoring. The data collection and control subsystem may contain a NCAP, enabling distributed monitoring and control of IEEE 1451.2 enabled sensors and actuators (Figure 1). Secure data transmission from the data collection and control subsystem is provided by IPSec and/or SSL over the building local Ethernet. VPNs may facilitate secure data transmission from the data collection and control subsystem to the master building server.

EMBEDDED SYSTEMS & INTERNET APPLIANCES

Low cost embedded designs, systems on a chip (SoC), and Internet appliances (IA) are paving the way to low-cost hardware implementations that includes the logger and local server functionality. Embedded devices are found in consumer electronics, computer peripherals, and control systems in automobiles, aircraft and other industrial applications. Their operating systems and application programs are combined on the same device. Eight, sixteen and even thirty-two bit microcomputers are found in embedded applications. High performance microprocessors have become today’s embedded controllers [10].

Hardware is divided into subsystems that are built around the micro-controller. The integration of volatile and non-volatile memories into microprocessors has permitted local storage of measured parameters without significantly increasing cost [11].

The integration of microprocessors, communication subsystems and volatile memories is widespread. New classes of embedded devices are evolving. SoCs are emerging as effective application designs, which are more complex than the traditional application specific integrated circuits of the past. Analog Devices' ADuC812 data converter is one such device. This device is a general-purpose data acquisition system on a single chip. It contains an eight-channel analog to digital and eight-channel digital to analog subsystem, serial communication facilities and an 8051 compatible micro-controller with volatile and non-volatile memories. This device has been demonstrated in the IEEE 1451 context [8]. In this application, the ADuC812 was implemented as a STIM. If the ADuC812 is embedded with a network adapter and stack, the STIM and NCAP could feasibly be combined into one system. This will make feasible an open, network-accessible data collection system.

SoC and high power microcomputers are also enabling the development of low cost single board computers (SBC). These systems are complete X86 compatible machines that have graphic, serial, parallel, and network subsystems.

SoC may reduce overall system cost because fewer components are necessary to get the same functionality. Benefits of a single chip system are not only power, weight and size but the price may be lower in high volume [10]. By combining the SBC with small footprint embedded operating systems (Linux, WinCE, BeIA), micro-relational databases and a Java Virtual Machine, complex applications may be run at the embedded level. SoCs are at the core of a new class of Internet enabled devices called Internet Appliances (IA). Commercial integrators are offering SoCs targeted at the IA market combining industry standard microprocessors, embedded programmable logic and memory [4].

SYSTEM PROGRAMMING

Building applications must have a strong framework that facilitates reusable components, operating system independence, and network operation. Sun Microsystems's programming paradigm, Java, accomplishes these. Applications must be developed fast to increase time to market. Sophisticated rapid application development tools such as an integrated development environment are available today for Java.

JAVA and Object-Oriented Programming

The Java programming language is an object-oriented that may be interpreted or compiled. Quite often programs written in Java are converted into byte codes, which are the portable machine language of the Java Virtual Machine (JVM) and interpreted on demand.

Portability.

Java was designed to be network oriented. In buildings, monitoring networks are composed of several systems with a variety of CPU and operating system architectures. A Java application can execute anywhere on the network. The Java compiler generates an architecture-independent object file format called byte code that can run on many processors. The machine specifics are handled by the architecture specific JVM. Java has been ported to various embedded systems from such low level devices as Dallas Semiconductor's Tiny Internet Interface to X86-based IA design such as National Semiconductor's Geode.

Distributed and Secure.

Java was built from the ground up with security in mind. Any code may be run with restricted permissions that prevent it from doing harm on the host system [12]. Defense occurs in access control of the memory space. For restricted programs, the JVM does not allow direct access to memory of the supporting system. The JVM also goes through a process known as bytecode verification whenever it loads an untrusted class [12]. This ensures that the JVM is protected from crashes and attacks. Java provides classes and interfaces for authentication and cryptography. These pieces of security allow Java to verify that any data is authentic and not modified in transit.

Java provides dynamic networking capability over multiple OS platforms. Java's design is powerful because it facilitates the clean definition of interfaces and makes it possible to provide reusable software, as Java objects are portable.

Java is a distributed application-programming paradigm that facilitates network oriented application communications across the building monitoring system. Data exchanged between these subsystems must be standardized and open.

XML

The World Wide Web Consortium's eXtensible Markup Language is changing the landscape of inter-application and systems data interchange by offering a framework of "self-describing" data. The self-describing nature of XML provides a common data format for data context.

XML is a text-based data description language that facilitates interoperability using an open standard. This markup language for data simplifies communications between applications and platforms. As HTML has enabled the display of information on the web, XML enables data interchange between systems. The structure and specifics of data are not lost in XML encoded data as with application specific and proprietary formats.

XML may be used at all levels of communication across the building monitoring system. Low-level communication from the data collection and control subsystems may encode data streams in XML to simplify upstream processing.

Figure 2 is an example of a Site Energy Report in a simple XML document. XML documents are built with various types of markup. Elements are the most common form of markup [13]. Elements are delimited by angle brackets pairs (<>, </>). In this example, the *</site-energy-report>* element contains many sub-elements. All elements within the *</site-energy-report>* support the same syntax whereby the data in between the angle brackets pairs is the content. The site-energy-measurement element contains attributes that are assigned specific values. The "entry" sub-element of the site-energy-measurement element has attributes: timestamp, position, description, units and value. Each of these attributes is assigned a numeric or character value.

This text file may be parsed and processed by another application. An example process would be to parse the site-energy-report and upload the data to a database or report generating system.

```
ENERGY XML EXAMPLE:
<site-energy-report>
<date>2001-04-10</date>
<time>08:00</time>
<site-information>
        <name-1>Evan's Library</name-1>
        <name-2>Texas A University</name-2>
        <site-number>492</site-number>
        <location>
           <address></address>
           <city>College Station</city>
           <state>Texas</state>
           <country>USA</country>
        </location>
        <use>Library facility</use>
        <area>square footage of the
        building</area>
        <description>This building was built in
        X. It has point of interest A,
        etc</description>
</site-information>
<site-energy-measurement>
        <entry timestamp="2001-04-02 00:00"
        position="1" description="Whole
        Building Electric" units="kWh"
        value="1200.01" />
        <entry timestamp="2001-04-02 00:00"
        position="2" description="Hot Water
        Flow Meter" units="Gal/h"
        value="2546.21" />
</site-energy-measurement>
</site-energy-report>
```

Figure 2. Energy XML Example.

SUMMARY

Next-generation networked monitoring systems will revolutionize future building monitoring systems in terms of better integration, control, and maintenance. Data collection and control subsystems may be networked through the Internet, enabling an open and powerful distributed monitoring system. Building monitoring systems developers may use the IEEE 1451 protocol with secure Internet communications to build an open, object-based system.

The technology discussed herein represents methods to improve monitoring in buildings. Today, sequential data files are obtained using dialup technology. Secure Internet technology allows flexible secured access to data without dialup costs and other limitations. Java and XML software technology provides a platform independent rapid and extensible development environment for building monitoring systems. XML technology facilitates extensible data interchange and usability by employing data descriptions. These descriptions provide context for multiple applications to share common data across the Internet.

REFERNCES

1. Kats, G.H., et al*, Energy Efficiency as a Commodity: The Emergence of an Efficiency Secondary Market for Savings in Commercial Buildings*, U.S. Department of Energy Protocol, April 1996.

2. Orihara, A., et al, An Autonomous "Decentralized System Platform under Multi-vendor Environments in Building Automation", *IEEE Proceedings*. ISADS. pp. 409-415, 1997.

3. Hunt, C., *TCP/IP Network Administration*, 2rd. Ed., O'Reilly and Associates, Inc. California, 1998.

4. Becker, S., "Unlocking the gate to mobile Web Access", *Portable Design*, pp.33– 48, April 2001.

5. Internet Engineering Task Force, March 2001*, IEFT IPSec Homepage,* Available: http://www.ietf.org/html.charters/ipsec-charter.html, (Read 04-15-2001).

6. Freier, A.O., March 1996. *The SSL Protocol Version 3.0*, Available: http://home.netscape.com/eng/ssl3/ssl-toc.html. (Read 4/10/2001).

7. Lee, K., "IEEE 1451: A Standard in Support of Smart Transducer Networking. IEEE Instrument and Measurement Technology Conference", *2000 IMTC IEEE Proceedings*, Vol. 2. pp. 525-528.

8. O'Mara,B., Conway,P., "Designing a IEEE 1451.2–Compliant Transducer", *Senors*. Vol 17 No. 8, August 2000.

9. Manders, E. J., Barford. L.A., "Diagnosis of a Continuous Dynamic System from Distributed Measurements", *2000 IEEE Proceedings*.

10. Karakehayov, Z.,Christensen, K.S., Winther, O., "*Embedded Systems Design with 8051 Microcontrollers – Hardware and Software*", Marcel Dekker, Inc. USA, 1999.

11. Nagaraju,M., Kumar,T.P. "*Networked Electronic Emergy Meters with Power Quality Analysis*", Signion Systems Ltd., Hyderabad, India, pp 45-57

12. Flanagan, D., *Java in a Nutshell*, 3rd. Ed., O'Reilly and Associates, Inc. California, 1999.

13. Walsh, N., October 1998, *What is XML? at Xml.com Homepage*, Available: http://www.xml.com/pub/a/98/10/guide2.html , (Read 4/16/2001).