Sensor Andrew: Large-scale campus-wide sensing and actuation

A. Rowe
M. E. Bergés
G. Bhatia
E. Goldman
R. Rajkumar
J. H. Garrett, Jr.
J. M. F. Moura
L. Soibelman

In this paper, we present Sensor Andrew, an infrastructure for Internet-scale sensing and actuation across a wide range of heterogeneous devices designed to facilitate application development. The goal of Sensor Andrew is to enable a variety of ubiquitous large-scale monitoring and control applications in a way that is extensible, easy to use, and secure while maintaining privacy. To illustrate the requirements of Sensor Andrew, as well as the capabilities and limitations of the system, we outline one such application in which multiple classes of energy sensors are combined with environmental sensors to not only monitor energy usage but also identify energy waste within buildings.

Introduction

In recent times, there has been a considerable increase in the number of sensors and actuators being embedded in the environment, electronic devices, and home appliances. Most modern buildings, for example, are equipped with transducers that allow one- or two-way communication of information such as electrical power-meter data, environmental sensors, and actuators that are part of building control and automation systems. The richness of sensors and actuators will only continue to increase with the convenience of technologies such as wireless sensor networks, and many real-world problems benefit from this type of distributed monitoring approach [1].

One application that has been of considerable interest and can utilize these sources of data involves the conservation of energy in buildings. Energy use by buildings in the United States alone, residential and commercial, accounts for a total of approximately 30 quadrillion British thermal units every year [2], which is close to 30% of all the primary energy consumed in the country. Optimizing the consumption to reduce that number has been a primary target of many energy-saving campaigns. Effective solutions to this problem require the fusion of data from many different sources, which is typically a difficult task. A single unifying infrastructure that enables the users to make use of the information from multiple different data streams is lacking.

Digital Object Identifier: 10.1147/JRD.2010.2089662

To meet these demands, we have built Sensor Andrew, an infrastructure to support large-scale sensing and actuation across Carnegie Mellon University. We envision a broad set of applications, including infrastructure monitoring, first-responder support, quality-of-life applications for the disabled, water distribution monitoring, building power monitoring and control, social networking, and biometric systems for campus security. Sensing devices that are used range from cameras and battery-operated sensor nodes to energy-monitoring devices wired into building power supplies. Supporting multiple applications and heterogeneous devices requires a standardized communication medium capable of scaling to tens of thousands of sources.

The Sensor Andrew infrastructure provides the following five services: 1) uniform access to heterogeneous devices, 2) sharing of transducers across applications, 3) scaling to many devices, 4) integration of many currently isolated subsystems; and 5) security and privacy protection for the data that is shared. Uniform access to devices is achieved using self-describing data objects defined by a transducer schema. Sharing transducer information across applications is achieved through a publish–subscribe software layer. Scalability is achieved through the use of encapsulated addressing: Each device in the system is addressed with a unique name, server address, and namespace attribute. Integration of subsystems is possible because of standardized communication mechanisms with software adapters, providing the last link of translation to the different communication protocols used by each subsystem. Finally, security and privacy are achieved through encryption, key management, access control, and policies. With these key services available, Sensor Andrew enables researchers to easily utilize an Internet-scale resource of sensors and actuators in a manner that is extensible, easy-to-use, and secure while maintaining privacy.

The core Sensor Andrew architecture is built around the Extensible Messaging and Presence Protocol (XMPP) [3], in which transducers are modeled as event nodes in a push-based publish–subscribe architecture. Sensor Andrew allows for easy integration of new sensors, as well as support for legacy systems. A data handler provides registration, discovery, and data-logging facilities for each device. These tools allow easy and controlled access to transducer information, which streamlines application development, promotes transducer reuse, and allows for interapplication collaboration.

We use building energy conservation as an application to motivate a system such as Sensor Andrew because this application requires cooperation between multiple sensing subsystems. Our specific application targets energy reduction in buildings by automatically identifying particular instances of waste. We made use of sensor readings from a collection of different plug-level (between the appliance and the outlet) and electric panel power sensors and actuators, environmental sensors, and motion detectors. We present two techniques that are based on statistical patterns and correlation between sensors as a starting point for a generalized waste-detection framework. The first technique correlates electrical load usage with occupancy sensors to establish a relationship that can be used to identify when loads are running in unoccupied areas of a building. The second technique monitors individual appliances through time in an attempt to recognize anomalous behavior. By using Sensor Andrew, this application can now be easily scaled to cover an entire city with easy-to-configure access control, security, and data management.

Related work

Multiple efforts have been made with respect to interfacing sensors with existing Internet Protocol (IP) infrastructures. In [4], the authors state that much of the work concerning sensor systems has resulted in "vertically built" designs in which individual components are only compatible with one another and not with other systems. The authors put forth the challenge of making a protocol for sensor networks that can unify communication in the same way that IP was able to support the Internet. Due to the resource-constrained, tightly coupled, and highly optimized nature of sensor networks, this problem continues to exist. Our objective in this work is not to standardize low-level sensor networking communication, but instead to enable unified tools and interoperability across multiple deployments. Some researchers have also tried to integrate sensor networks with IP version 6 (IPv6) [5, 6]. IPv6 solves many of the problems of addressing and sending/receiving data packets to and from individual sensor devices. It allows standard tools such as *traceroute* and *ping* to be used for network diagnostics. However, IPv6 does not solve many of the higher level challenges associated with managing and providing applications with the data. In this paper, we provide a framework that runs on top of existing network protocols such as IPv6 and IPv4 and addresses access control, registration, discovery, event logging, and management of transducer devices beyond a single subnet.

Multiple projects have created architectures that are associated with many of the goals of the Sensor Andrew project—by enabling transducer reuse and collaborative Internet-scale sensing. The Internet-scale Resource-Intensive Sensor Network Service (IrisNet) [7] project uses a distributed database-centric architecture that facilitates the storage, processing, and retrieval of transducer information. It was primarily intended for Internet-connected desktop personal computers and inexpensive commodity off-the-shelf sensors such as webcams. In contrast, the Sensor Andrew project focuses on the management of a wide range of devices, including resource-constrained transducers, which may not have direct Internet connectivity. It also provides presence notification essential for supporting mobile devices. In Global Sensor Networks (GSNs) [8], the authors propose a service-oriented architecture in which sensors can be queried using commands in the style of Structured Query Language and using web services. A peer-to-peer architecture is used for efficiently indexing data. Unlike Sensor Andrew, the GSN is largely focused on single-user applications with little support for access control. Simple Sensor Syndication [9] creates a publish–subscribe model for sensor data using Really Simple Syndication feeds. A small python server located at a sensor gateway preprocesses sensor data and can be configured to publish particular events of interest. Again, the system does not address issues of access control or resource management that are central to the design of Sensor Andrew. The Arch Rock [10] edge server provides a web front end for mote-based devices in which data can be retrieved via web pages and services based on the Simple Object Access Protocol. While they provide a very capable front end for a single network, their system does not address issues in data sharing or discovery at a larger scale. In [11], the authors present a system for making sensor data shareable and searchable over the Internet. The system is primarily designed for searching sensor data. It does not focus on management of sensors between different users with different access control rights. The same is true for the system presented in [12]. Sensor Andrew provides device presence information and is more suitable for real-time communication to support actuation.

Multiple research groups have worked on collaborative sensing services such as SenseWeb [13] from Microsoft Research, SensorWeb [14], and Sensorpedia [15]. These systems are targeted toward visualizing and sharing data with end users. They currently appear to have little support for managing actuators. The aim of Sensor Andrew is not only to collect sensor data but also to support control of environments and to enable interaction between software agents. These projects complement Sensor Andrew and could be integrated to aid in navigation and visualizing events.

Sensor Model Language (SensorML) [16] and Transducer Markup Language [17] are emerging standards that essentially create a common description of sensors and actuators, as well as the systems that generate the data. In the future, we hope to integrate these models with Sensor Andrew, but at the time of development, the schemata were overly complex for our applications, and there was little existing support software available to aid in adoption. In contrast with existing models, the Sensor Andrew schema is biased toward the streaming of sensor and actuator data rather than describing the details of the sensed phenomena.

Despite all of these challenges, domain-specific solutions continue to be developed and deployed at an ever-growing rate. In power distribution systems, for example, there is currently a large effort to integrate various sensing and actuation components to better manage the electric grid. Electric utilities provide their customers with monthly reports of their consumption, but automated meter reading systems are starting to provide more frequent updates. Additionally, there is a growing number of commercially available power meters that can provide traditional power metrics in real time at varying degrees of details such as the main electrical feed of the building, the circuit panels, the individual circuits, or individual outlets. The costs of the latter solutions range from hundreds to tens of thousands of dollars, with their cost increasing with the level of detail obtained.

Commercial efforts are underway to add sensing devices in homes to provide users with energy-usage feedback. Google PowerMeter [18] is a software package that has an interface with smart metering technology to display household energy usage. Companies such as Tendril, Inc. [19], AlertMe [20], and Trilliant [21] are taking a more proactive approach by offering monitoring devices that home owners can install themselves to monitor energy usage. These efforts exclusively focus on electrical appliances, and most do not provide actuation capabilities or access to other diverse sensors. There are, however, some projects that do address the sensor fusion and actuation strategies for energy management [22, 23]. Additionally, there is a small number of research [24] and commercial [25] projects on nonintrusive load monitoring (NILM), a technique that utilizes signal processing and statistics-based algorithms for processing the power data for the entire house to infer the consumption of individual appliances.

The building industry has been working for a number of years toward standard communication protocols and data formats to simplify the exchange of information between monitoring and control equipment in commercial and residential buildings [26, 27]. Multiple attempts [28, 29] have been made to translate communications between different systems to facilitate cross-domain interactions. These systems are narrowly focused on supporting control and automation interactions and were never intended to operate with the number of users or scale of Sensor Andrew.

Paper organization

The remainder of this paper is organized as follows. The following section describes the architecture requirements, design goals, and tradeoffs associated with the Sensor Andrew infrastructure. The section that follows describes the architecture we chose for Sensor Andrew and where it is located in the design tradeoff space. We then discuss the implementation of various system components. Later, we present a section that describes how these components are used in a building energy-management application, and finally, the last section summarizes our contributions and discusses our future plans.

Sensor Andrew architecture and design goals

We adopted the following design goals for Sensor Andrew:

1. *Ubiquitous large-scale monitoring and control*—The sensing infrastructure should exist at a significantly large scale to encourage the development of new and innovative applications. The infrastructure should support both sensing and actuation.
2. *Ease of management, configuration, and use*—The ease of use of the system needs to be considered for both managing the infrastructure and providing simple ways for application developers to form interfaces with their own and other subsystems. A process for registering and discovering transducers relevant to the user's project should also be included.
3. *Scalability and extensibility*—The ability to support a large number of devices and users is of paramount importance. Extensions made by a particular project to satisfy a new requirement should ideally benefit the entire community.
4. *Built-in security and privacy*—The system should support security and privacy considerations, including encryption, key management, access control, and account/user management. Various aspects of privacy should be governed through well-defined policies.
5. *True infrastructure sharing*—One of the most unique contributions of Sensor Andrew is the notion of multiple heterogeneous applications and devices that can utilize

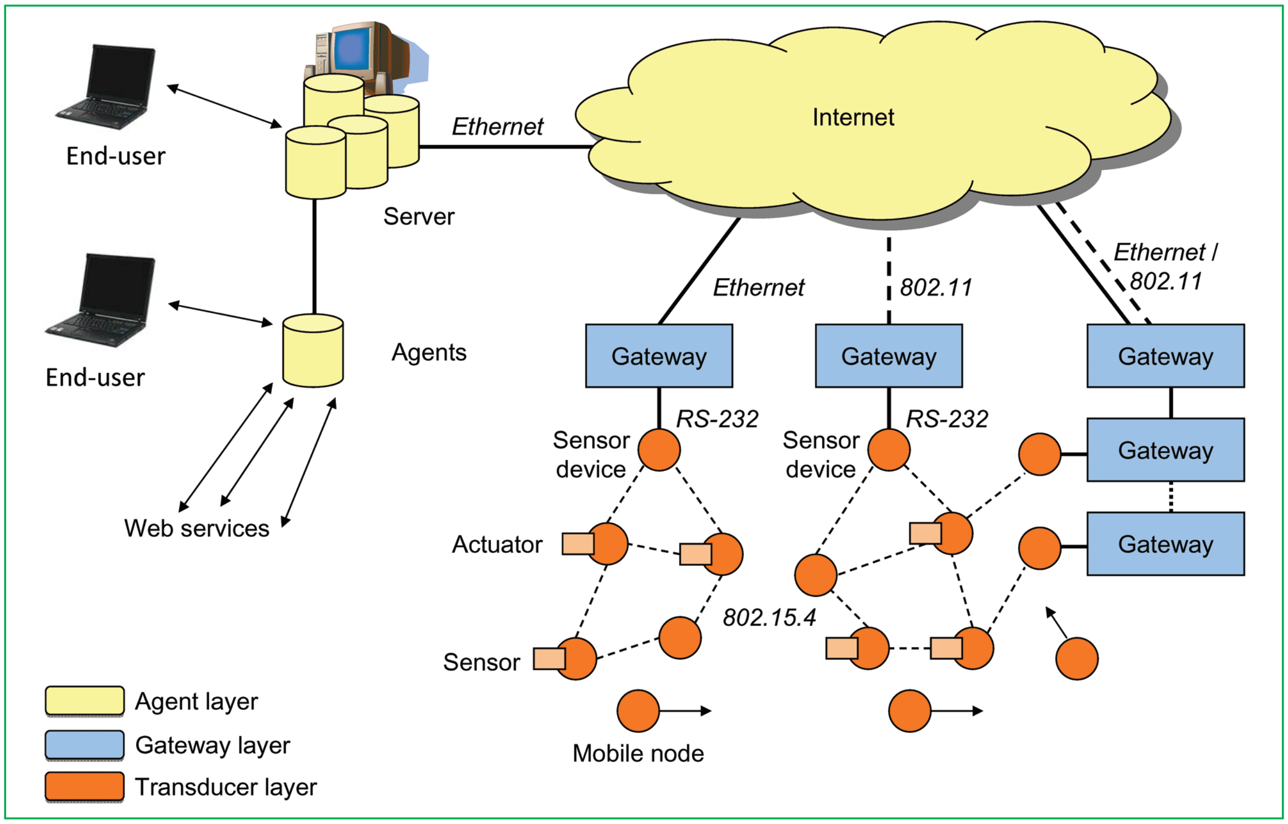

Figure 1

Three-tiered architecture of Sensor Andrew. Dashed and solid lines represent wireless and wired connections, respectively. Actuators and sensors are both transducers and can be attached to sensor devices or mobile nodes.

one another's services. The architecture must easily integrate information from multiple applications, creating additional value to all users as new types of applications are envisioned.

6. *Evolvability*—The architecture must be capable of evaluating different computation paradigms. It must also allow for rapid prototyping in large quantities to demonstrate practical usage and utility. It also needs to be able to change over time. Being able to evolve with and support these changes will be important for incorporating unforeseen and innovative applications in the future.
7. *Robustness*—The system should be robust and be able to reconfigure itself.

Given these challenging goals, it would be infeasible to design every component of Sensor Andrew without relying on previous technologies. We utilize existing technologies wherever possible and innovate as necessary. In this section and the next, we discuss how each of these goals was met by describing the different components of the system.

Figure 1 illustrates the classic three-tiered architecture we adopted with a front-end agent layer, a gateway layer, and a transducer layer. The servers and gateways operate as part of the campus network, whereas the transducer layer may communicate over a variety of different bus or network protocols. What follows is a description of each of the elements of this architecture.

Communication
An infrastructure such as Sensor Andrew has many communication requirements, namely, standard messaging formats, extensible message types, point-to-point and multicast messaging, support for data tracking or event logging, security, privacy, access control, and Internet-scale performance. To meet these requirements, we chose to make use of XMPP, an open Internet protocol inspired by Extensible Markup Language (XML), traditionally used for online chat communications and supported by industry leaders such as IBM, Google, and Apple.

Much in the same way a domain can run its own email server, addressing in XMPP is defined first with a client identification [referred to as a Jabber identification (JID)] followed by a domain name and then a namespace. Entities using XMPP are classified as clients and servers. For example, `gw_x@sensor.xxx.xxx.edu/refrigerator` identifies a particular gateway node's address (`gw_x`) at an XMPP server address (`sensor.xxx.xxx.edu`) with its namespace specified as `refrigerator`. An XMPP server with the correct access permissions can pass a local client's requests to another XMPP server, which, in turn, can pass the request to the destination client. The addition of namespaces appended to the addresses allows for the creation of multiple views.

XMPP supports publish–subscribe messaging, where JID clients can send and receive messages through *event nodes*. Event nodes are addressable data channels that allow clients to publish and subscribe to event feeds. Nodes may also maintain a history of events, provide meta-information about the event feed, and contain access control lists (ACLs). This push model of communication provides a powerful mechanism for distributing sensor data to any interested application or user.

XMPP satisfies our initial requirements in five ways. First, it provides a standard scalable messaging and presence protocol with user/group authorization, authentication, and access control. Since XMPP is already an Internet standard, we can make use of commercially available servers that are maintained by the open-source community. Second, the addressing and messaging scheme of XMPP is optimized for short messages with point-to-point and broadcast capabilities. The addressing scheme is not bound to a physical network location, making it ideal for mobile devices. Third, XMPP provides a publish–subscribe functionality for pushing (e.g., transmitting) sensor data. This is an ideal model for mass distribution of data. Fourth, XMPP provides organized event messages with an internal database for storing transaction records. Finally, XMPP can utilize clustering or replications to meet large-scale demands and provide primary-backup fault tolerance.

Building upon XMPP as a transport layer, we still require various additional components in the system. First, we need schemas to describe a wide range of transducers. Next, we need a set of adapters to facilitate communication between low-level hardware and XMPP. We also require a set of software agents and libraries to streamline development and help manage all of the data streams. In the following section, we describe each of these components in more detail.

Transducer layer

Elements of the transducer layer are end-point sensors or actuator devices with little or no processing power, which have the ability to measure or change some physical characteristic of the environment. These devices typically require custom software (often running on a microcontroller interface) to read general-purpose input/output pins, analog-to-digital converters, serial messages, wireless sensor nodes, or other similar low-level devices and protocols. The goal of the transducer layer is to pass information to an Internet-connected device that is part of the gateway layer. In our building energy optimization application, for example, the most critical transducers include motion sensors, current and voltage transformers, and temperature, light-intensity, and audio-level sensors. **Figure 2** shows these transducers and the devices that host them.

The heterogeneous nature of Sensor Andrew requires supporting vastly different types of transducer devices. To this point, we have mostly described low-data-rate devices such as wireless sensor nodes. Sensor Andrew must also support high-data-rate devices such as video-streaming systems. For devices with high bandwidth requirements, XMPP offers a hands-off mechanism for establishing a secure link between two clients. In these cases, there is no transaction logging, but devices still benefit from addressing, access control, encryption, and authentication.

Gateway layer

The gateway layer consists of devices that have access to the Internet. As described in the next section, these devices run a full XMPP client with associated software *adapters* for any attached transducers. Gateways are responsible for running adapters that format the transducer layer information so that it can be passed onto the server layer. They also have the ability to create and manage the event nodes to which they publish data. Information is passed from the transducer layer to the gateway layer using low-level protocols. Messages are then passed between the gateway layer and the server layer using XMPP with the Sensor Andrew transducer schema. Devices at the server layer can then subscribe to event nodes to receive data once it is published.

For example, a gateway in a classic wireless sensor network, such as the one depicted in Figure 2, would be part of the gateway layer and would publish sensor values for each sensor node. In contrast, the sensor nodes would be considered part of the transducer layer. In general, the device on the Internet that does the publishing would be considered part of the gateway layer.

Adapters publish to unique XMPP event nodes for each device. **Figure 3** shows an overview of the XMPP publish–subscribe system. Event nodes can be hierarchically organized so that subscribers can be notified when particular related groups of nodes produce data.

Agent layer

The agent layer consists of the XMPP servers along with various client applications called *agents*. The purpose of the

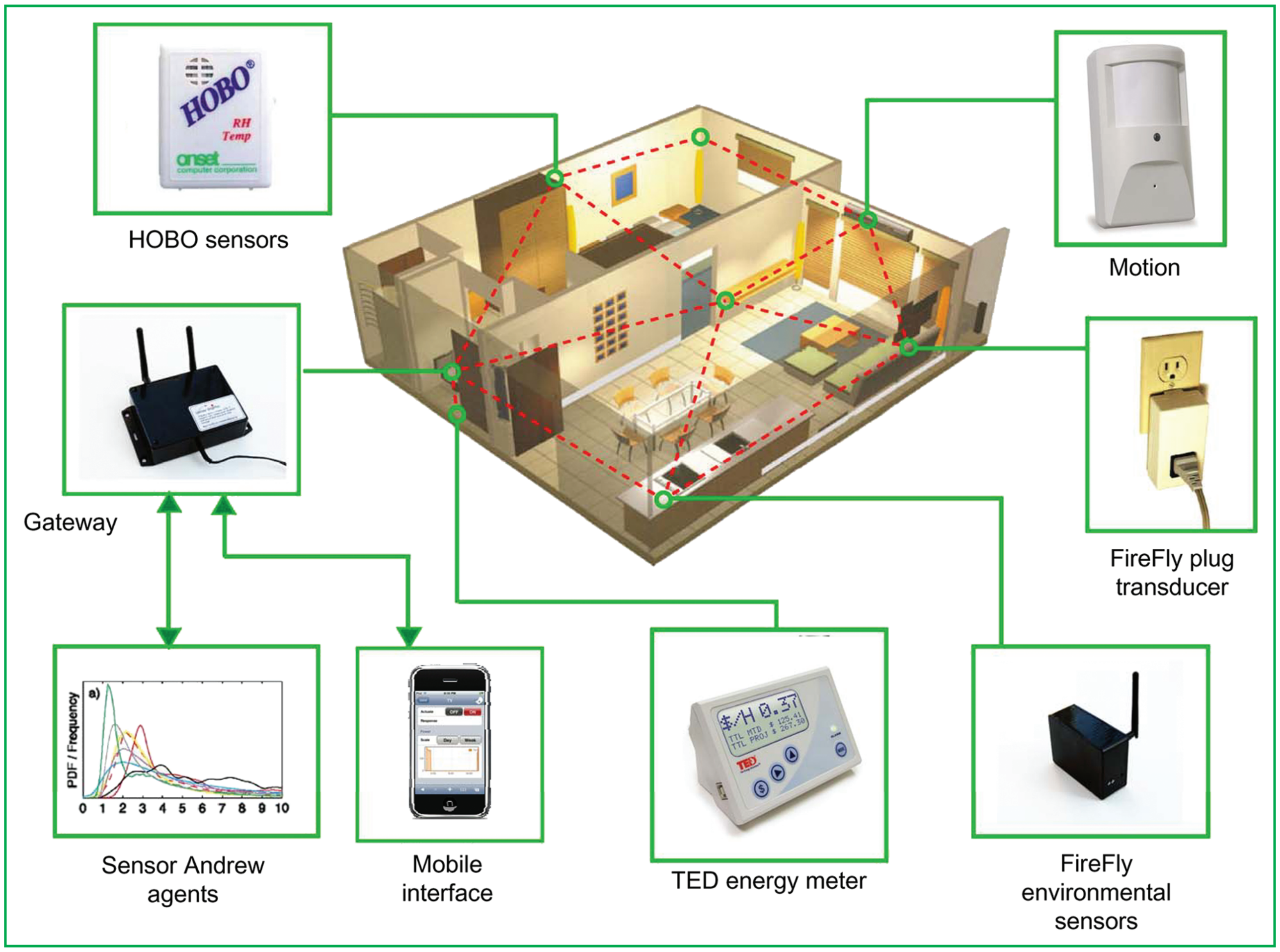

Figure 2

Components of our building energy optimization application. Here, TED refers to a simple but accurate energy monitor that allows users to observe electricity usage throughout the house in real time. The red dashed lines indicate the wireless mesh network formed by all the sensor nodes.

server layer is to provide a simple means for applications running on desktop-class machines to communicate with each other. Applications can subscribe to event nodes and publish their own *meta-events*. These meta-events can then be "consumed" and used by applications creating increasingly richer data streams.

Actuation support
Actuation in Sensor Andrew must be implemented with a consideration toward security and resource sharing; hence, it takes place as a split-phase operation with an action signal followed by a completion callback. First, gateway devices that support actuators are required to subscribe to their respective actuator event nodes. An agent can publish an actuation request to the event node, which is then translated by the adapter of the gateway into a native command for the actuator. Once the actuator operation has completed its transaction, the local adapter is responsible for publishing updated state information back to the event node. An agent could subscribe to this event node to confirm the requested transaction. The low-level actuation transaction occurs at the transducer and gateway layers, but the authentication, permission, and messaging associated with the actuation are handled at the server layer.

System components
Sensor Andrew is designed to operate using a standard XMPP server that supports messaging and event publishing. In our current deployment, we use the Openfire [30] server from Ignite Real-Time Software. Access control for users is provided through ACLs associated with users, groups, and event nodes. Each event node has whitelists and blacklists describing which users can read or write data. The details of how the server manages access control are outside the

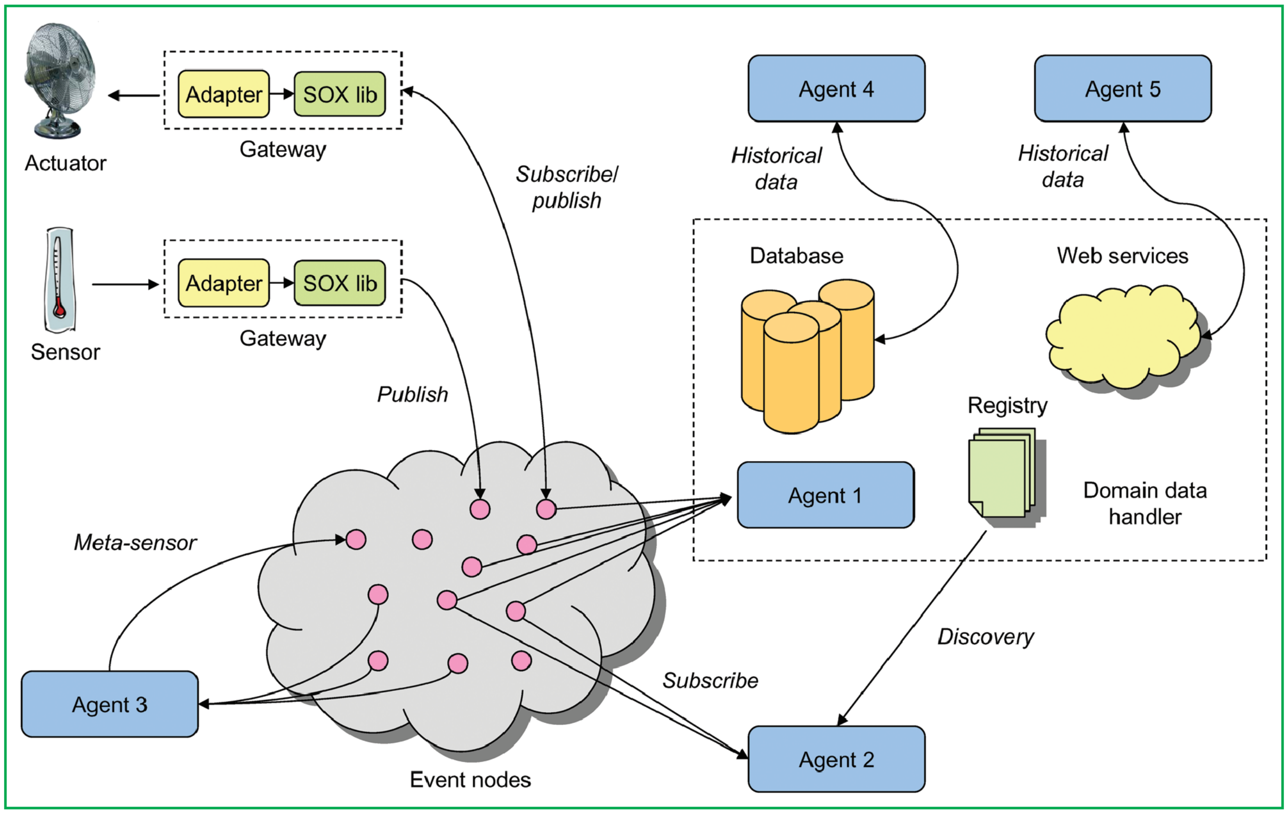

Figure 3

XMPP publish–subscribe transactions to support collection of sensor networking data. (SOX lib: Sensor over XMPP library.)

scope of this paper, but the interested reader can find more information about access control in the XEP-0074 extension of XMPP, where XEP stands for XMPP Extension Protocol.

We developed the Sensor Over XMPP (SOX) library as a layer on top of XMPP that provides a set of common tools and a uniform interface for all Sensor Andrew applications. Various command-line tools wrap different application programming interface calls for simple use on any UNIX**-based computer. To support a variety of hardware platforms and operating systems, we have implemented the SOX library in C, .NET, LabVIEW**, Java**, and Python.

Adapters utilize the SOX library to convert transducer data into SOX-compatible messages. These interfaces, as previously explained, run on the gateway layer collecting and formatting information from the transducer layer. This library facilitates the development process for new adapters. Currently, we support more than a dozen commercial and research-grade devices such as The Energy Detective (TED) energy meter, HOBO** environmental sensors, FireFly 802.15.4, and wireless sensor nodes. All of the sensors and actuators used in our energy application have a corresponding software adapter that leverages our SOX library.

Data handler

The data handler is a web application that oversees all read/write activities on the transducer registry and transducer value archive. At the core, the data handler contains the registry and archive schema, business rules, and read/write functions. A web interface was constructed to allow browsing, editing, and creating transducer and device metadata records in the registry. The registry interface will guide the users through the data-entry process to facilitate participation in the Sensor Andrew network. It also allows users to browse and search existing transducers and archived data. The data handler inherits the permission model of the XMPP server by requiring users to log in with a valid JID, which it then uses to authenticate them with the XMPP server.

A database system supports the data handler by implementing the transducer registry and data value archive schema that defines the relationships among the numerous kinds of metadata that Sensor Andrew supports. The details of this schema are beyond the scope of this paper. For more information on the data handler, please refer to our technical report [31].

Security and privacy

Given the physical nature of the information collected and exchanged throughout the Sensor Andrew system, one has to be naturally concerned about security and privacy issues. Our privacy mechanisms range from technological solutions, such as encryption, to informal policies, such as proper information distribution to the users and labeling of devices. The principal investigators of each project using Sensor Andrew is required to be compliant with a project-wide privacy policy and the university Institutional Review Board requirements. Devices placed in public areas must clearly display what information they are capturing and where further information about the project can be located.

All server-layer communication takes place over XMPP using Transport Layer Security connections. All client applications are also required to authenticate with a username and password. Any guest access to the network is automatically restricted by the ACLs to anonymous data that could not be used to identify individuals. Whenever possible, security is used within individual transducer layer subnets. For example, our example wireless sensor networking deployment uses encryption for all infrastructure communications.

User workflow

In order for users to access Sensor Andrew, they must register to obtain a username and password. This login is the XMPP JID that is used to track the creation of event nodes and their associated permissions. Users can immediately log in to the main web interface and search for existing publicly available sensors. The system supports browsing through the use of a hierarchical map interface or by transducer attributes as described by the meta-information of each device. The data handler provides plotting of historical data over user-defined time periods for quick previewing and the ability to export data files for local processing. To add a transducer into the system, the user must first describe the device through the web interface of the data handler, and in return, an event node and a set of database keys are generated for each transducer. At this time, the user can also specify the username that the transducer adapter will use to publish data (this could also be the original username if desired). Once a device has been registered, the data handler will automatically subscribe to the new event node and begin logging data. The user must add the provided event-node JIDs and database keys into the configuration file for the specific SOX transducer adapter. Once the adapter starts, it will connect to the main XMPP server and begin to publish data to the configured event nodes.

SOX-specific enhancements

Anytime a system leverages existing components, technical and design incompatibilities will likely arise. We now summarize a few notable enhancements to the core Sensor Andrew technologies that were required to streamline the interconnection of each system layer. To utilize XMPP, we developed the SOX library and data schema required to represent and transfer sensor data. This included an extensive set of adapters (often running on embedded platforms) required for interfacing with transducers. Modification of the XMPP server was required to add group permissions to publish–subscribe event nodes since, by default, access control only applies to users. We are suggesting this ACL addition along with our schema as part of an XMPP extension protocol to the XMPP community. Finally, we provide an extensive set of core SOX agents for registration, discovery, logging, and viewing of sensor data. Access to these tools allows new users to quickly store and visualize data before building more complex analytical tools.

Building energy optimization

We now turn to a specific application to illustrate the advantages of utilizing an infrastructure such as Sensor Andrew when developing applications that require access to multiple transducers.

To optimize the energy usage within a building, it is first necessary to identify which appliances consume the most energy and then highlight possible areas that users could intervene. Monitoring electrical loads can be achieved using energy meters at various points in a building's electrical distribution tree. The more difficult challenge is to identify energy *waste* within the system to provide users with suggestions for optimization. Examples of energy waste include unused appliances being left on, windows or doors left open allowing hot or cool air to be lost, appliances slowly degrading in performance over time, and so-called phantom loads associated with inefficient electronic wakeup circuits. Many of these sources of waste are unique to one particular environment; however, we present two techniques based on statistical patterns and correlation between sensors as a starting point for a generalized waste-detection framework. The first technique correlates electrical load usage with occupancy sensors to establish a relationship that can identify when loads are left running in unoccupied areas of a building. The second technique monitors individual appliances over time in an attempt to recognize anomalous behavior. For example, if the refrigerator door is left ajar, the refrigerator begins to consume more energy than usual. Less obvious anomaly examples include situations in which devices such as printers or flat-panel televisions are locked in error states that cause them to continue consuming energy even though they appear to be off. Both of the management techniques presented here share the goal of providing users with meaningful information about where energy may be wasted as opposed to simply displaying energy plots over time.

The first step toward deploying our energy management system was to outfit two apartments with a set of sensors and actuators to capture relevant information regarding the occupancy status and power consumption of the different areas in the living space. Figure 2 shows a diagram of the primary hardware components used to support our application. At the main circuit panel in each house, a NILM system [32] was used to measure the total electricity consumption of the space. The NILM system analyzes power transients generated when appliances turn on and off to identify devices that are consuming energy. Each time an appliance changes state, a unique voltage and current signature can be used as a fingerprint to match against a prerecorded set of appliances. This allows the NILM system to make estimates, solely from the circuit breaker, as to what types of appliances are currently running and how much power they consume. The NILM system runs on a desktop computer with a National Instruments analog data-acquisition card and sensors attached to the breaker panel of the home. We provide a LabVIEW Sensor Andrew adapter that publishes the total power information and the virtual-appliance information to the NILM event node.

In addition to panel-energy instrumentation, we deployed a set of plug-level energy meters [33] as part of a FireFly wireless sensor network. The plug-level sensors can directly sense and actuate individual electrical outlets providing control, as well as "ground truth" to help validate the NILM estimations. Here, *ground truth* refers to unequivocal measurements that can be used for various comparisons. As part of the wireless sensor network, we also deployed environmental sensors in each room that would periodically report temperature, light, acceleration, and sound level to Sensor Andrew. To detect occupancy, we deployed commercial passive infrared motion detectors in a configuration such that they covered the main areas of each room. A modified Linux**-based 802.11 wireless router was running the Sensor Andrew adapters to publish data on behalf of each sensor. The gateway subscribes to any event nodes that might need to service actuation requests from the server layer. In total, each apartment was equipped with a NILM computer, a router, approximately six environmental sensors, six motion sensors, and ten plug meters. All communication from these devices is collected (as part of the transducer layer) and eventually published to Sensor Andrew event nodes using the SOX adapters.

Occupancy-correlated energy usage

We collected three months of data from two off-campus apartments using the Sensor Andrew data-handler database agent. This could have been also collected using a user-defined agent that subscribes to each event node of interest. During the initial deployment setup, the users labeled appliances associated with each energy meter and named the various sensors with human-readable descriptions through a web interface. A correlation matrix with correlation values for all possible pairings between signals from motion detectors and metered electrical appliances in the apartment was obtained. If the electrical usage was significantly correlated with a particular motion sensor, it was considered a good indication that the appliance is physically colocated with the device. Devices such as overhead lights that are typically used while there is motion in the room tend to be better correlated. Appliances such as washers or dryers will have a high correlation during initial activation, but after the user leaves the room, they continue to run without motion. The correlation between signals from each motion detector and each electrical appliance automatically identifies these associations over time without requiring the user to explicitly configure them. **Table 1** shows an example of the correlation coefficient matrix for one of the test apartments. Highlighted values have low p values for testing the hypothesis of no correlation and, hence, indicated a strong linkage between the sensors. In this case, we are interested in both positive and negative correlations. The bottom two rows list the best overall correlated sensor and the best correlated motion sensor. Once the mapping between appliance usage and motion data is found, a Sensor Andrew agent subscribes to all energy streams correlated with a motion detector and triggers an alert anytime an appliance is being used for more than 15 minutes without any accompanying motion. **Figure 4(a)** shows two clusters of appliances and highlights (in light green) those sections detected as waste.

Energy anomaly detection

In addition to detecting occupancy-inferred energy waste, we developed another agent that monitors the energy data streams to find usage anomalies in daily energy consumption over time. There are multiple possible techniques for detecting anomalies in data streams that perform better or worse depending on how much application-specific information they are provided. In our initial deployment, we wanted to create a simple generalized model that could operate on a wide variety of appliances. The algorithm uses a seven-day moving window to calculate the average energy consumption of each device. An anomaly is then defined as any daily value on which the appliance uses more than twice this weeklong rolling average. **Figures 4(b)** and **4(c)** show two examples of this filter running over a period of 70 days. The black bars denote the energy usage of one day, whereas the horizontal red sticks show the threshold used to detect an anomaly. Any time the black bar exceeds the anomaly threshold, the system alerts the user. Figure 4(b) shows the energy usage of a refrigerator as an example of a device that varies over time (potentially based on room temperature) but should generally not trigger the

Table 1 Correlation coefficients for the different sensor streams in an apartment. (Note: Bold values have a p value $< 10^{-25}$.)

Correlation coefficient	*Bathroom light*	*BR light*	*Bed lamp 1*	*Bed lamp 2*	*TV*	*Media laptop*	*Tall lamp*	*Refrigerator*	*Toaster*	*Laptop*	*LCD monitor*	*Bathroom motion*	*Bedroom motion*	*Living room motion*	*Kitchen motion*
Bathroom light	1.00	**0.15**	0.04	-0.01	0.10	-0.01	0.02	0.06	0.00	**0.13**	**0.11**	**1.00**	**0.15**	**0.19**	**0.13**
Bedroom light	**0.15**	1.00	0.09	0.09	-0.02	-0.06	-0.03	0.00	-0.01	0.03	-0.04	**0.15**	**1.00**	0.08	0.03
Bed lamp 1	0.04	0.09	1.00	0.00	-0.01	0.01	-0.01	-0.02	0.00	0.00	0.05	0.04	0.09	0.00	0.00
Bed lamp 2	-0.01	0.09	0.00	1.00	-0.02	0.06	-0.02	0.00	0.00	-0.01	-0.04	-0.01	0.09	-0.01	-0.01
TV	0.10	-0.02	-0.01	-0.02	1.00	**-0.28**	-0.07	0.02	0.04	**0.15**	**0.22**	0.10	-0.02	**0.26**	**0.15**
Media laptop	-0.01	-0.06	0.01	0.06	**-0.28**	1.00	0.09	0.00	-0.02	-0.05	-0.06	-0.01	-0.06	-0.08	-0.05
Tall lamp	0.02	-0.03	-0.01	-0.02	-0.07	0.09	1.00	0.04	-0.01	0.08	0.08	0.02	-0.03	0.06	0.08
Refrigerator	0.06	0.00	-0.02	0.00	0.02	0.00	0.04	1.00	0.02	0.09	0.03	0.06	0.00	0.03	0.09
Toaster	0.00	-0.01	0.00	0.00	0.04	-0.02	-0.01	0.02	1.00	**0.13**	0.01	0.00	-0.01	-0.01	**0.13**
Laptop	**0.13**	0.03	0.00	-0.01	**0.15**	-0.05	0.08	0.09	**0.13**	1.00	0.07	**0.13**	0.03	**0.34**	**1.00**
LCD monitor	**0.11**	-0.04	0.05	-0.04	**0.22**	-0.06	0.08	0.03	0.01	0.07	1.00	**0.11**	-0.04	0.04	0.07
Bathroom motion	**1.00**	**0.15**	0.04	-0.01	0.10	-0.01	0.02	0.06	0.00	**0.13**	**0.11**	1.00	**0.15**	**0.19**	**0.13**
Bedroom motion	**0.15**	**1.00**	0.09	0.09	-0.02	-0.06	-0.03	0.00	-0.01	0.03	-0.04	**0.15**	1.00	0.08	0.03
Living room motion	**0.19**	0.08	0.00	-0.01	**0.26**	-0.08	0.06	0.03	-0.01	**0.34**	0.04	**0.19**	0.08	1.00	**0.34**
Kitchen motion	**0.13**	0.03	0.00	-0.01	**0.15**	-0.05	0.08	0.09	**0.13**	**1.00**	0.07	**0.13**	0.03	**0.34**	1.00
Best correlation:	Bathroom motion	Media laptop	Bedroom motion	Bedroom motion	Living room audio	Bedroom light	Laptop 1	Bedroom light	Kitchen motion	LCD monitor	Laptop 1	Bathroom light	Bathroom motion	Kitchen motion	Living room motion
Best motion correlation:	Bathroom motion	Bedroom motion	Bedroom motion	Bedroom motion	Living room motion	Living room motion	Living room motion	Kitchen motion	Kitchen motion	Kitchen motion	Bathroom motion	Living room motion	Bathroom motion	Kitchen motion	Living room motion

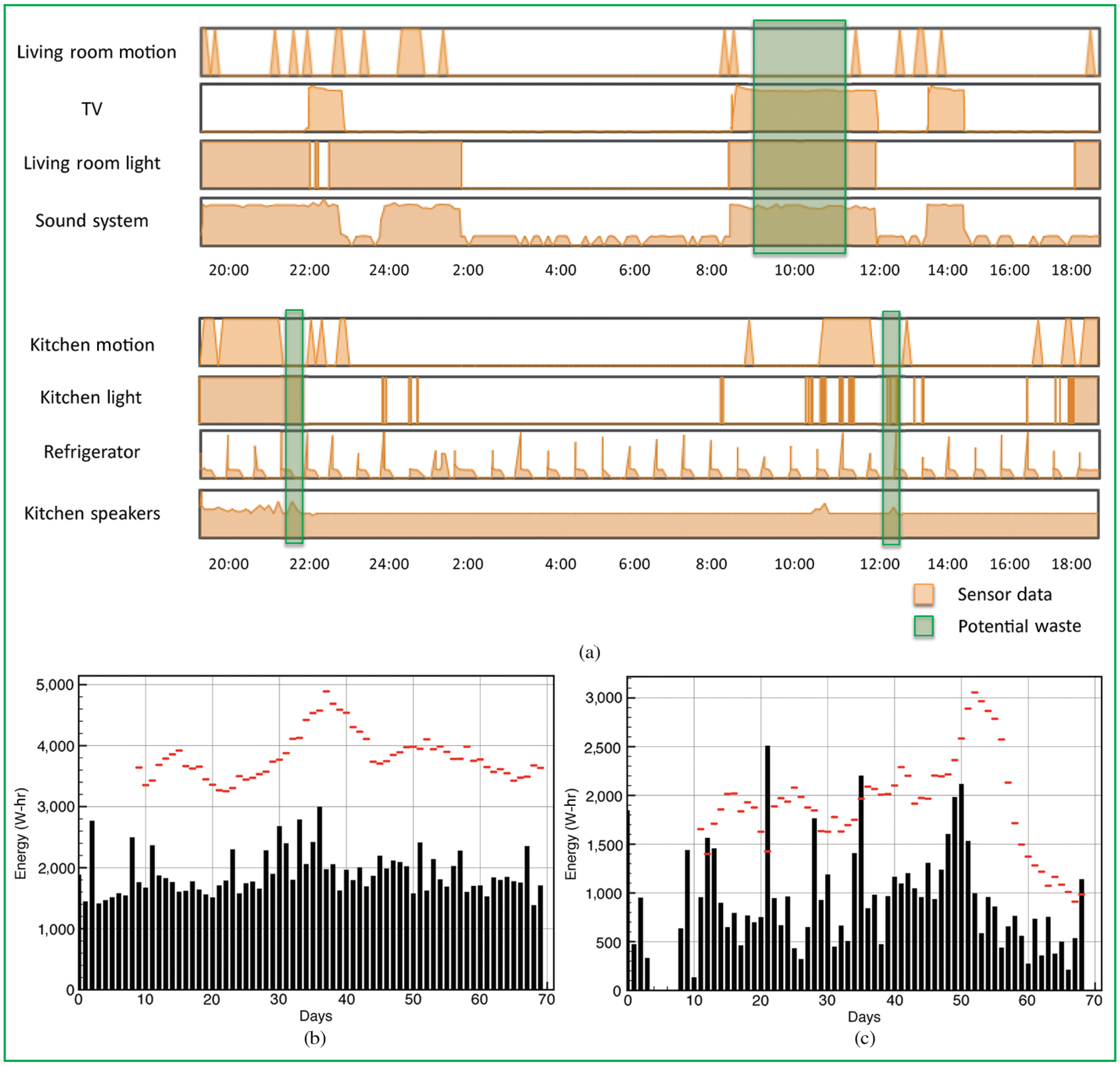

Figure 4

Experimental results. (a) Energy waste identification. (b) Daily consumption with anomaly trigger points for refrigerator. (c) Daily consumption with anomaly trigger points for television. Note that the vertical axis for the graphs in (a) shows the binary values for the motion sensors and active power values (in watts) for the plug-level sensors.

anomaly detector. Figure 4(c) shows television (TV) usage and how, in a few cases, it triggered the anomaly detector. In three out of the four instances when the alert was triggered, it was due to the TV continuing to operate after the user had left the room or fallen asleep. The first triggering was a false alarm due to erratic viewing patterns and lack of historical values for the moving average filter. In the future, more complex detectors could prevent these situations.

Limitations

As shown in the previous sections, Sensor Andrew provides an easy-to-use framework for developers to access and share large numbers of sensor data streams. However, there are a few limitations that warrant further investigation. Our requirement of transaction logging, for example, forces all messages to pass through a server even if the action could be completed with a point-to-point message. To help alleviate server bottlenecks, a unique server can be used for a single domain of interest. For high-speed sensor data, there is currently no database-logging capability. A transaction record exists, but the bandwidth of streaming data would be too much burden on the main message server.

With respect to privacy, our current permission model does not support fine-grained access control over individual record elements. The model only provides access control at the event-node level. This makes it difficult to provide a user permissions to read one sensor value, but not another on the same device. This would only be possible if each individual sensor was modeled by its own event node.

Another current weakness in the architecture that we are trying to address is the ability to capture meta-information about virtual sensors. We would like a way to exactly capture an operation performed on a data stream and trace its relationships back to the source sensor set. For example, dew point is computed using information on humidity and temperature. If there is a dew point sensor, one should be able to extract exactly what operations were performed in each sensor to compose the final value. This becomes extremely difficult when the relationships are not simple linear functions. For example, room occupancy can be composed with any number of heuristics.

The potentially vast volume of information could eventually become a performance bottleneck. Our initial tests indicate that XMPP is compute limited by Transport Layer Security. The current solution is to distribute the servers to avoid individual bottlenecks. However, it could also be possible to increase performance by optimizing the underlying XML messages by converting them into a more compact binary form. This could have a significant impact on processing speed at the cost of obfuscating our packet format. For now, since most transducer packets are small, XML is sufficient and easy to work with. Finally, the time taken to establish an initial XMPP connection is significant compared with just sending a transducer packet, which indicates that it is better to maintain an open session rather than periodically opening and closing it.

Conclusions and future work

In this paper, we presented a multidisciplinary campus-wide scalable sensor network called Sensor Andrew that is designed to host a heterogeneous mix of sensing and low-power applications. We presented the requirements, goals, and design tradeoffs associated with such a large-scale system. Specifically, the goals of Sensor Andrew are to support ubiquitous large-scale monitoring, operation, and control of infrastructure in a way that is extensible and easy to use and that provides security while maintaining privacy. Our architecture provides a complete communication framework allowing new projects to be easily integrated with existing projects to extend overall capabilities. A three-tiered architecture allows for ease of management and facilitates security and privacy controls. Open-source software, customized and integrated with our extensions, enables seamless and scalable communications across layers.

We highlighted the features of Sensor Andrew through a building energy-waste identification application. Although our main goal was to present the application to demonstrate the flexibility, security, and data management facilities of Sensor Andrew, the preliminary waste-detection results show promise. In one of the test apartments, the system detected that lights and computers were left unattended for as much as 25% of the time they were in use.

As future work, we plan to enhance the proposed architecture by streamlining the interface for registering, configuring, and querying sensors through web services. We also plan to continue the development of tools and applications exploring real-world sensing and actuation scenarios. Efforts are already underway to build applications that integrate sensor data with Building Information Models [34], as well as with the Building Interior Data Space Model [35], to provide better models for exploring sensor metadata. We are also investigating integration with SensorML and continuing support for new applications.

Acknowledgments

This work would not be possible without the help and support of many Sensor Andrew project members. In particular, we thank Burcu Akinci, H. Scott Matthews, Bruno Sinopoli, Priya Narasimhan, Matthew Sanfilippo, Anu Pradhan, Pine Liu, Karthik Lakshmanan, Max Buevich, and Saurabh Taneja. This work was funded in part by the Pennsylvania Infrastructure Technology Alliance. Sensor Andrew is a project of the Center for Sensed Critical Infrastructure Research (CenSCIR) at the Institute for Complex Engineering Systems (ICES), Carnegie Mellon University.

References

1. C. Chen-Ritzo, C. Harrison, J. Parasczak, and F. Parr, “Instrumenting the planet,” *IBM J. Res. & Dev.*, vol. 53, no. 3, pp. 1–16, May 2009, Paper 1.
2. *Annual Energy Review*, Energy Inf. Admin., Washington, DC, 2008.
3. XMPP Standards Foundation, *XMPP Standards Foundation*. [Online]. Available: http://xmpp.org/
4. D. Culler, P. Dutta, C. T. Ee, R. Fonseca, J. Hui, P. Levis, J. Polastre, S. Shenker, I. Stoica, G. Tolle, and J. Zhao, “Towards a sensor network architecture: Lowering the waistline,” in *Proc. 10th Conf. Hot Topics Oper. Syst.*, Santa Fe, NM, USENIX Association, 2005, vol. 10, pp. 24–24
5. J. Hui, N. Kushalnagar, D. Culler, and G. Montenegro, “Transmission of IPv6 Packets over IEEE 802.15.4 Networks,” in *Internet Engineering Task Force RFC 4944*. [Online]. Available: http://tools.ietf.org/html/rfc4944

6. J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan, “Building efficient wireless sensor networks with low-level naming,” *SIGOPS Oper. Syst. Rev.*, vol. 35, pp. 146–159, 2001.
7. J. Campbell, P. B. Gibbons, S. Nath, P. Pillai, S. Seshan, and R. Sukthankar, “IrisNet: An Internet-scale architecture for multimedia sensors,” in *Proc. 13th Annu. ACM Int. Conf. Multimedia*, Hilton, Singapore, 2005, pp. 81–88.
8. K. Aberer, M. Hauswirth, and A. Salehi, “Global sensor networks,” School Comput. Commun. Sci., Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland, 2006.
9. M. Colagrosso, W. Simmons, and M. Graham, “Simple sensor syndication,” in *Proc. 4th Int. Conf. Embedded Netw. Sensor Syst.*, Boulder, CO, 2006, pp. 377–378.
10. Arch Rock Corporation, *Arch Rock*. [Online]. Available: http://www.archrock.com/
11. S. Reddy, G. Chen, B. Fulkerson, S. J. Kim, U. Park, N. Yau, J. Cho, and J. H. Mark Hansen, “Sensor-Internet share and search—Enabling collaboration of citizen scientists,” in *Proc. ACM Workshop Data Sharing Interoperability World-Wide Sensor Web*, Cambridge, MA, 2007, pp. 11–16.
12. N. B. Priyantha, A. Kansal, M. Goraczko, and F. Zhao, “Tiny web services: Design and implementation of interoperable and evolvable sensor networks,” in *Proc. 6th ACM Conf. Embedded Netw. Sensor Syst.*, Raleigh, NC, 2008, pp. 253–266.
13. A. Santanche, S. Nath, J. Liu, B. Priyantha, and F. Zhao, “SenseWeb: Browsing the Physical World in Real Time,” in *Proc. 5th ACM/IEEE Int. Conf. IPSN, Demo Abstract*, Nashville, TN, 2006, pp. 547–548.
14. A. Sheth, C. Henson, and S. S. Sahoo, “Semantic sensor web,” *IEEE Internet Comput.*, vol. 12, no. 4, pp. 78–83, Jul./Aug. 2008.
15. Oak Ridge National Laboratory, *Sensorpedia*. [Online]. Available: http://sensorpedia.com/
16. OpenGIS, *Sensor Model Language (SensorML)|OGC*. [Online]. Available: http://www.opengeospatial.org/standards/sensorml
17. Open Geospatial Consortium, Inc., *Transducer Markup Language (TML) | OGC Network*. [Online]. Available: http://www.ogcnetwork.net/infomodels/tml
18. Google, *Google PowerMeter*. [Online]. Available: http://www.google.org/powermeter/
19. *Tendril ≫ Smart Energy for Life | Smart Grid Technology*. [Online]. Available: http://www.tendrilinc.com/
20. *AlertMe—Smart Energy and Monitoring Solutions*. [Online]. Available: http://www.alertme.com/
21. *Trilliant: Empowering the Smart Grid*. [Online]. Available: http://www.trilliantinc.com/
22. X. Jiang, M. V. Ly, J. Taneja, P. Dutta, and D. Culler, “Experiences with a high-fidelity wireless building energy auditing network,” in *Proc. 7th ACM Conf. Embedded Netw. Sensor Syst.*, Berkeley, CA, 2009, pp. 113–126.
23. Y. Kim, T. Schmid, Z. M. Charbiwala, and M. B. Srivastava, “ViridiScope: Design and implementation of a fine grained power monitoring system for homes,” in *Proc. 11th Int. Conf. Ubiquitous Comput.*, 2009, pp. 245–254.
24. S. Shaw, S. Leeb, L. Norford, and R. Cox, “Nonintrusive load monitoring and diagnostics in power systems,” *IEEE Trans. Instrum. Meas.*, vol. 57, no. 7, pp. 1445–1454, Jul. 2008.
25. Enetics Inc., *Enetics Advanced Metering and Analyzers*. [Online]. Available: http://www.enetics.com/
26. ASHRAE SSPC 135, *BACnet Website*. [Online]. Available: http://www.bacnet.org/
27. Organization for the Advancement of Structured Information Standards, *oBIX*. [Online]. Available: http://www.obix.org/
28. W. S. Lee and S. H. Hong, “Implementation of a KNX-ZigBee gateway for home automation,” in *Proc. IEEE 13th Int. Symp. Consum. Electron.*, Kyoto, Japan, 2009, pp. 545–549.
29. M. Neugschwandtner, G. Neugschwandtner, and W. Kastner, “Web services in building automation: Mapping KNX to oBIX,” in *Proc. 5th IEEE Int. Conf. Ind. Informat.*, Vienna, Austria, 2007, pp. 87–92.
30. Jive Software, *Ignite Realtime: Openfire Server*. [Online]. Available: http://www.igniterealtime.org/projects/openfire/
31. A. Rowe, M. Berges, G. Bhatia, E. Goldman, R. Rajkumar, L. Soibelman, J. H. Garrett, and J. M. F. Moura, *Sensor Andrew: Large-Scale Campus-Wide Sensing and Actuation*. Pittsburgh, PA: Carnegie Mellon Univ., 2008.
32. M. Berges, E. Goldman, H. S. Matthews, and L. Soibelman, “Learning systems for electric consumption of buildings,” in *Proc. ASCE Int. Workshop Comput. Civil Eng.*, Austin, TX, 2009.
33. M. Buevich, A. Rowe, M. Berges, and R. Rajkumar, “Sensor Andrew Gateway Agent (SAGA),” in *SAGA*. [Online]. Available: http://www.andrew.cmu.edu/user/agr/projects/saga/
34. G. Lee, R. Sacks, and C. M. Eastman, “Specifying parametric building object behavior (BOB) for a building information modeling system,” *Autom. Construction*, vol. 15, no. 6, pp. 758–776, Nov. 2006.
35. BISDM, *Building Interior Space Data Model*. [Online]. Available: http://bisdm.org/

Received March 2, 2009; accepted for publication April 4, 2010

Anthony Rowe *Electrical and Computer Engineering Department, Carnegie Mellon University, Pittsburgh, PA 15213 USA (agr@ece.cmu.edu).* Mr. Rowe is a Ph.D. candidate in the Electrical and Computer Engineering Department at Carnegie Mellon University. He received a B.S. degree in computer engineering from Carnegie Mellon University in 2003. His research interests include real-time systems, operating systems, wireless sensor networks, lightweight computer vision, and robotics.

Mario E. Bergés *Department of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 USA (marioberges@cmu.edu).* Mr. Bergés is a Ph.D. candidate in the Civil and Environmental Engineering Department at Carnegie Mellon University. He received a B.Sc. degree in 2004 from the Instituto Tecnológico de Santo Domingo in the Dominican Republic and an M.Sc. degree in 2007 from Carnegie Mellon University. His research interests include infrastructure monitoring, building energy management, machine learning for signal processing, and sensor networks.

Gaurav Bhatia *Electrical and Computer Engineering Department, Carnegie Mellon University, Pittsburgh, PA 15213 USA (gauravbhatia@cmu.edu).* Mr. Bhatia received his B.S. and M.S. degrees in electrical and computer engineering from Carnegie Mellon University in 2002 and 2003, respectively. He is a Research Staff Member in the Electrical and Computer Engineering Department at Carnegie Mellon University. His research interests include distributed real-time systems, embedded systems, and model-based design.

Ethan Goldman *Vermont Energy Investment Corporation, Burlington, VT 05401 USA (egoldman@veic.org).* Mr. Goldman received his B.A. degree from Hampshire College in 2001 and his M.S. degree from Carnegie Mellon University in 2009. He is the Measurement and Verification Specialist at Efficiency Vermont, a statewide provider of energy-efficiency services. His research and professional interests include sensor metadata schemas, building energy performance metrics, and energy consumption feedback interfaces.

Ragunathan Rajkumar *Electrical and Computer Engineering Department, Carnegie Mellon University, Pittsburgh, PA 15213 USA (raj@ece.cmu.edu).* Dr. Rajkumar obtained his B.E. (with honors) degree from the University of Madras in 1984 and his M.S. and Ph.D. degrees from Carnegie Mellon University in 1986 and 1989, respectively. He is a Professor in the Departments of Electrical and Computer Engineering and of Computer Science at Carnegie Mellon University. His research interests include all aspects of embedded real-time systems as well as quality-of-service support in operating systems and networking. He was also the primary founder of TimeSys Corporation, a vendor of embedded Linux and Java products.

He has served as chair of several international conferences and has authored a book and more than 90 publications in conferences and journals.

James H. Garrett, Jr. *Department of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 USA (garrett@cmu.edu).* Dr. Garrett received his B.S., M.S., and Ph.D. degrees in civil engineering from Carnegie Mellon University (1982, 1983, and 1986, respectively). He is a professor in the Department of Civil and Environmental Engineering at Carnegie Mellon University. He is also currently the head of the department and co-Director of the Center for Sensed Critical Infrastructure Research. From 2000–2006, Garrett served as the Associate Dean for Academic Affairs in the College of Engineering at Carnegie Mellon. He is a member of the Advanced Infrastructure Systems group in the CEE (Civil and Environmental Engineering) department. His research and teaching interests are oriented toward applications of sensors and sensor systems to civil infrastructure condition assessment; mobile hardware/software systems for field applications; representations and processing strategies to support the usage of engineering codes, standards, and specifications; knowledge-based decision support systems; neural networks for modeling and interpretation problems in civil and environmental engineering.

José M. F. Moura *Electrical and Computer Engineering Department, Carnegie Mellon University, Pittsburgh, PA 15213 USA (moura@ece.cmu.edu).* Dr. Moura holds an M.Sc. and a D.Sc. degree in electrical engineering and computer science from Massachusetts Institute of Technology and an E.E. (electrical engineering) degree from Instituto Superior Técnico (IST, Portugal). He is University Professor of Electrical and Computer Engineering at Carnegie Mellon University and, by courtesy, a Professor of BioMedical Engineering. Dr. Moura's research interests are in statistical signal and image processing, network science, and distributed decision and inference. He is a Fellow of IEEE and AAAS, and a corresponding member of the Academia das Ciencias from Portugal. He was President of the IEEE Signal Processing Society and Editor-in-Chief of the *IEEE Transactions on Signal Processing*.

Lucio Soibelman *Department of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 USA (garrett@cmu.edu).* Dr. Soibelman holds an M.Sc. degree in civil engineering from Universidade Federal do RGS (Rio Grande do Sul) in Porto Alegre, Brazil, and a Ph.D. degree in civil engineering systems from the Massachusetts Institute of Technology. He is a professor in the Department of Civil and Environmental Engineering at Carnegie Mellon University. His research interests include the use of information technology for economic development and construction management, process integration during the development of large-scale engineering systems, data mining, image reasoning, machine learning, and advanced infrastructure systems. He is the current Editor-in-Chief of the American Society of Civil Engineers (ASCE) *Journal of Computing in Civil Engineering* and the current chair of the ASCE Construction Institute Construction Research Council.