Web-based Heterogeneous WSN Integration using Pervasive Communication

Mihaela Cardei, Anthony Marcus, Ionut Cardei, Timur Tavtilov
Department of Computer and Electrical Engineering and Computer Science
Florida Atlantic University
Boca Raton, FL 33431, USA
E-mail: {mihaela@cse., amarcu10@, icardei@cse., ttavtilo@}fau.edu

Abstract

This paper presents a REST-compliant service oriented architecture of a web-based heterogeneous wireless sensor network monitoring system that has applicability in remote patient monitoring in healthcare. As smartphone and web-page technologies are ubiquitous nowadays, our architecture uses smartphone as a gateway between the data collected and the Internet. The system uses a heterogeneous WSN consisting of environmental sensors, medical sensors, and smartphone's internal sensors. To validate the system, a prototype experiment of the system is implemented using Android smartphone platform, Crossbow Micaz motes, and Alive Technology hart and activity monitor sensor.

1. Introduction

Wireless sensor networks (WSNs) provide rapid, untethered access to information and computing, eliminating the barriers of distance, time, and location for many applications in healthcare, weather monitoring, infrastructure security, environmental and habitat monitoring, traffic control, and many more.

One of the most promising application of WSNs technology is in healthcare. Remote health monitoring, or telemedicine, is an emerging key area of research that integrates wireless communications, sensing, and healthcare. As the healthcare costs are rising, life expectancy is increasing, and the world population is aging [7], there has been a need to monitor patients and residents out of hospitals, in their own environment, and during emergency situations. Strategically placing a number of wireless sensors on the human body creates a wireless body area network (WBAN) that can monitor various vital signs that can provide real-time feedback to the user and medical personnel.

Usually, sensors are placed on the human body as tiny patches or incorporated in the clothes or shoes, allowing ubiquitous health monitoring for extended periods of time. Such sensors can measure relevant physiological parameters such as heart rate, blood pressure, body and skin temperature, oxygen saturation, respiration rate, electrocardiogram, etc. In-home and nursing home pervasive sensor networks may assist residents, patients, and their caregivers by providing continuous medical monitoring, medical data access, memory enhancement, control of home appliances, and emergency communication. Monitoring patients at home drastically reduces the medical expenses. One such example is monitoring of patients after a surgery, during recovery, or monitoring of patients with certain medical condition.

Various works for remote healthcare monitoring using sensor networks have been proposed recently. Some research works focus on continuous medical monitoring of degenerative diseases such as Alzheimer, Parkinson, or similar cognitive disorders [12, 10]. Other projects such as Code Blue at Harvard [11], Scalable Medical Alert Response Technology (SMART) [14], and Medical Emergency Detection in Sensor Networks (MEDiSN) [9] extends WSNs for emergency applications, hospitals, or disaster recovery.

Another use of WSN technology is smart home-care applications, such as SmartDrawer [2] that monitor the medication of chronically ill patients, and ALARM-NET [15] that uses environmental sensors (e.g. air quality, light, temperature) to measure environment parameters and patient motion to detect prolonged periods of inactivity. An architecture for a wide telemedicine network for continuous patient monitoring is proposed in [13]. Patients equipped with WBANs consisting of sensors such as ECG, EMG sensor for muscle activity, EEG, blood pressure, motion sensors, and breathing sensors are continuously monitored, and medical data is transmitted to a medical server on the Internet for processing, storage, and access by the healthcare professionals.

In this paper we propose a system that can be used to monitor patients remotely. The system uses a heterogeneous WSN consisting of medical sensors, environmental sensors, and smartphone's internal sensors. The framework proposed is extensible and new types of sensors can be added as needed. As smartphone and web-based technologies are ubiquitous nowadays, our architecture uses smartphone as a gateway between the collected data and the Internet, used both in data acquisition and data consumption (e.g. accessing the data). Web-based technologies and services represent fundamental mechanisms in remote data storage and access. To provide a high degree of interoperability and abstraction, we propose a REST-compliant Service Oriented Architecture (SOA) that uses Web Services (WS). Then, we implement

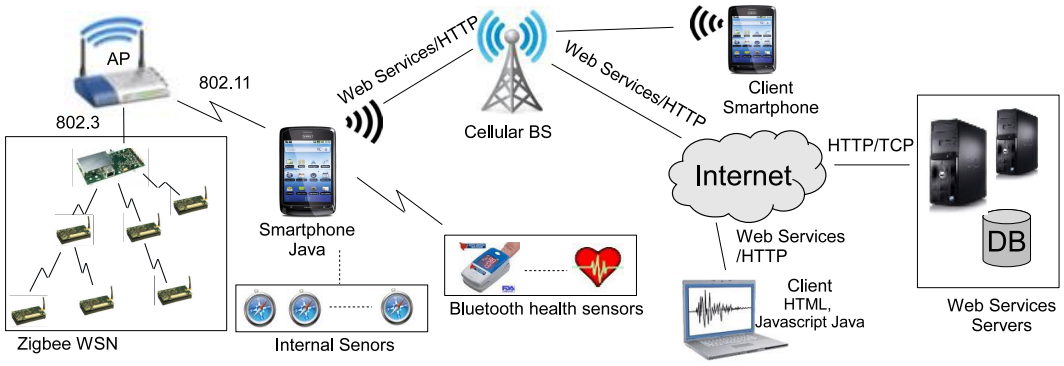

Figure 1. SOA architecture

a prototype experiment of the system using Android smartphone platform, Crossbow Micaz motes [3], and Alive Technology hart and activity monitor sensor [1].

2 Service Oriented Architecture for Patient Monitoring

In this section we present the architecture for the in-home patient monitoring system. We derive the following requirements for the system:

- The system must be extensible to allow integration of various sensors. Various patients need to be monitored with different sensors depending on their medical condition. In addition, environmental sensors can be used to measure different in-home parameters such as light, temperature, etc.
- The system must be scalable with the number of patients and number of users. A caregiver (e.g. nurse, doctor, etc.) is expected to supervise a number of patients. In addition, a patient can be monitored by family members and friends.
- The patient to be monitored may be inside a building or external surroundings, and he can be fixed or mobile.
- The monitored data must be made available to the authorized users using monitoring web-based applications and as a data store that can be queried.
- The users can be local or remote, connected to the Internet, fixed or mobile.
- The system must use the available technology.
- The cost of the system must be relatively small, otherwise the system may become too expensive to implement and operate.
- Only authorized users must be able to access the data and data transmission must be secure.

We propose a Service Oriented Architecture (SOA) that is REST-compliant and uses Web Services (WS) to implement the desired operations. Representational State Transfer (REST) is a software architectural style based on the work of Roy Fielding [6].

With REST, each data item is handled as a resource, and each resource is an atomic data unit. A number of methods are implemented for each resource. To provide a high level of abstraction and interoperability, each method is implemented as a RESTful web service.

The W3C [8] defines a Web Service (WS) as "a software system designed to support interoperable machine-to-machine interaction over a network". The W3C has brought a complete protocol framework, denoted $WS-*$ specifications or in general as SOAP web services. Note that RESTful web services are proposed to be more lightweight and avoid the overhead from XML format in the W3C $WS-*$ specifications. Web communication is implemented using HTTP and the structure of the request/reply messages can be expressed using JavaScript Object Notation (JSON) format.

The architecture of the system is presented in Figure 1. This is a web-based WSN monitoring system that implements two main functions, data acquisition and data consumption. Smartphones have become an ubiquitous technology and can be used as a gateway to collect sensor measurements. Data can be processed locally at the smartphone, and then sent using the cellular network (e.g. 3G/4G) to the Internet, and from here to the WS servers where data is stored in a database for future use. Data transmission can be done using HTTP (HTTPS) over the TCP protocol.

Users that consume sensor data are the caregivers (nurse, doctor, etc.), relatives and friends that must be able to access patient data. For simplicity, we will refer to them as caregivers for the rest of the discussion. They run client software applications on PCs, laptops, PDAs, or smartphones to connect to the server on the Internet to access patient data. Clients access data by calling the web services and rely on standard web protocols, such as HTTPS over TCP.

A wide variety of commercially available sensors can be used by applications. One such category is formed by medical sensors such as ECG, EMG sensor for muscle activity, EEG, blood pressure monitors, motion sensors, pulse oxymeters, and breathing sensors. The medical sensors associated to a patient form a Wireless Body Area Network (WBAN). Medical sensors usually transmit data to the smartphone using Bluetooth.

The second sensor category are the ZigBee environmental sensors, such as Micaz motes [3] from Crossbow. As an example, a mote equipped with the MTS310 multi-sensor

board can sense temperature, humidity, barometric pressure, sound, and ambient light. Micaz motes transmit data using the ZigBee protocol stack, where physical and MAC layer follow the specifications of the IEEE 802.15.4 standard.

The third sensor category is formed by the smartphone's internal sensors, such as 3D accelerometer, digital camera, GPS, and compass. The accelerometer sensor can be used to infer the current body position and actions, for instance if a patient is walking and how much energy is spent.

The sensor measurements are transmitted to the smartphone and from there to the WS server, as specified previously. What measurements and how often they are transmitted depends on the application.

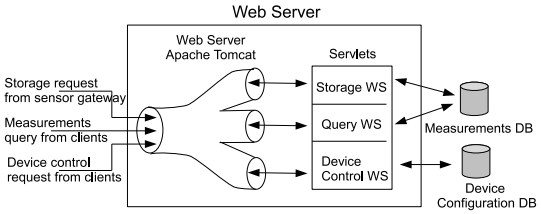

Figure 2. Web services dispatch

Figure 2 shows the WS dispatch at the web server. There are three types of requests coming to the web server:

- Storage requests from sensor gateways (smartphones): these are messages containing sensor data, which will be processed by the *Storage WS* servlet and then stored in the *Measurements DB*.
- Measurement queries from clients: these are queries coming from caregivers' clients authorized to access sensor data. These queries are solved by the *Query WS* servlet, data being retrieved from the *Measurements DB*.
- Device control requests from clients: these are requests from caregivers' clients or from sensor gateways. The caregivers' requests are different commands for the sensors. For example there may be a request to start/stop the ECG sensor readings. A sensor gateway periodically queries the web server for any new commands for the sensors that it manages. These requests are processed by the *Device Control WS* servlet and stored in the *Device Configuration DB*.

Figure 3 illustrates the smartphone system. The smartphone acts as a gateway for the patient sensors. External sensors can communicate with the smartphone using Wi-Fi, Bluetooth, or ZigBee interfaces. In addition, smartphone's internal sensors (accelerometer, GPS, compass, etc.) may be part of the monitoring system. For example, the communication with the Alive Technologies Heart and Activity Monitor Sensor [1] is done using the Bluetooth interface, and the communication with ZigBee motes is done using the ZigBee interface. Data from the sensors is processed locally by the smartphone and then is transmitted to the Internet web server using a cellular data network.

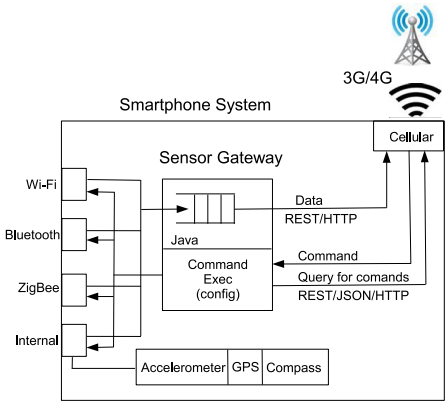

Figure 3. Smartphone system

There are two ways in which sensor data can be delivered to consumers: push and pull techniques. Using the push technique, the client registers on the server to receive new data updates. When the data arrives from the sensor gateway, the server will send ('push') them to clients that registered. This mechanism is also called the publish/subscriber model. In the pull technique, the client periodically interrogates the web server for new data. If available, these data will be returned to the client. Our system in this paper uses the pull technique. Web clients send queries to the server identifying the nodes, sensors, and specific time intervals. To efficiently utilize resources when sensor data is not needed, a sensor gateway uses a similar pull technique for turning on/off sensors by sending queries for configuration commands to the server. Commands for sensor activation are sent by sensor control clients, such as providers, that need to access patient sensor data and turn the sensor is off. Activation commands contain a timeout period to prevent sensors from being left active indefinitely. The server solves any configuration request conflicts from multiple clients. This protocol is described in Figure 5.

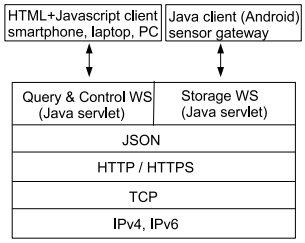

Figure 4. Protocol stack

Figure 4 presents the protocol stack. The two main communication flows are (i) between the Java client running on the sensor gateway and the Java servlet implementing the storage WS on the web server, and (ii) between the HTML/Javascript client running on the caregiver's smarthone (or laptop, PC, etc.) and the Java servlet implementing the query and control of the WSs on the web server.

JSON is used to serialize and transmit structured data over a network connection. Data transmission is accomplished using HTTP/HTTPS protocol with TCP for a reliable commu-

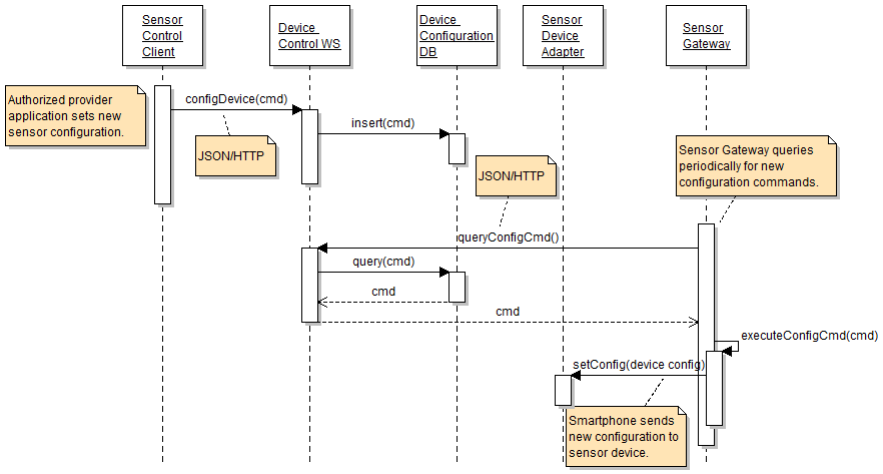

Figure 5. Setting new sensor configurations

nication. At the network layer, IP/IPv6 are used.

Next, we present the sequence diagrams for the three main types of operations supported by our system: (i) setting new sensor configurations using the *Device Configuration DB*, (ii) report sensor measurements to the *Measurements DB*, and (iii) client queries the *Measurements DB* for sensor data.

Figure 5 shows the sequence diagram for setting new sensor configurations. Requests are being sent by the sensor control client to the Device Control WS implemented on the web server, and from here are stored in the Device Control DB. Sensor gateways periodically query the Device Control DB for new sensor configuration requests. This query is sent through the Device Control WS on the web server. The web service returns any configuration requests from control clients. They are parsed by the sensor gateway to basic sensor configuration commands that are sent to the corresponding sensor devices using Device Adapters. A Device Adapter is a software component on the Sensor Gateway that knows to talk with internal and external sensor devices connected wirelessly. They implement a uniform interface and simplify integration with new sensors.

Upon startup, when a new sensor is connected, and periodically thereafter, the Sensor Gateway sends to the Device Control WS a registration message that contains the gateway ID and the configuration of all sensors under control and their status (e.g. active, sampling rate, resolution, energy reserve). The system's battery level is considered to be another sensor that can be used for managing sensing schedules efficiently. The Device Control service maintains these information in the Device Configuration DB. Periodically, each Sensor Gateway sends a query for new pending configuration commands to the Device Control WS using the query operation. All accumulated applicable commands are then returned to the gateway, where they are parsed and executed. Such commands can turn on/off sensors for periods of time, change their parameters (e.g. sampling rate), or change the gateway's operation mode, such as turning it off during the night.

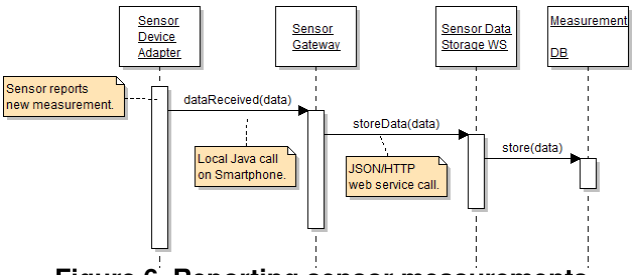

Figure 6. Reporting sensor measurements

The sequence diagram in Figure 6 describes the operations involved in sending the sensor measurements from sensors to the web server. First, sensor data are sent from the device itself to a sensor Device Adapter. The Adapter forwards the to the Sensor Gateway using the *dataReceived()* operation. There, measurements are filtered, timestamped if necessary, and buffered for transmission. From the buffer, the sensor gateway (smartphone) transmits the data to the Sensor Data Storage WS on the web server using a secure HTTPS connection. The Sensor Data Storage servlet stores the incoming data to the Measurements DB.

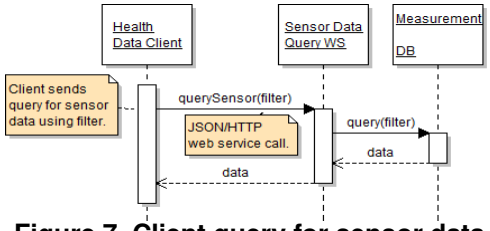

Figure 7. Client query for sensor data

Figure 7 illustrates the sequence of operations necessary for the caregivers' clients to obtain sensor data from the web server. These clients run in a browser or on a smartphone, laptop. The Health Data Client (e.g. caregiver application) sends the query to the Query WS running on the web server. The web service queries the Measurements DB according to the query filter that identifies the node, sensor, time interval, and other attributes. The sensor data is sent then back to the client with an JSON/HTTPS reply.

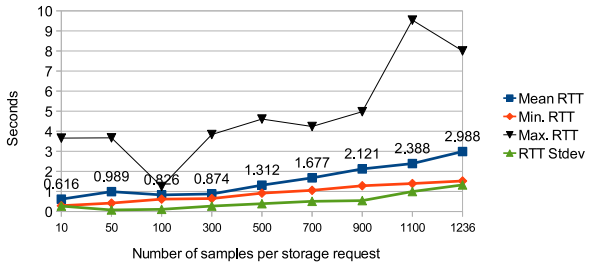

Figure 8. Data storage web service round trip time depending on the number of sensor samples per message

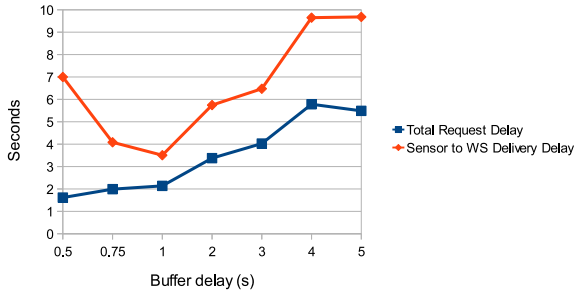

Figure 9. The total web service call RTT depending on the buffering delay

An important aspect of any medical monitoring application is taking care of the security considerations. According to the legislation, only authorized users must have access to the medical information. In addition, data transmission must be encrypted and authenticated, which can be accomplished using HTTPS. Designing the security aspects is not one of the objectives of this paper. This would require a threat analysis [4, 5] and providing mechanisms for each possible threat.

3 Experiment Description and Performance Measurements

In this section we describe the experimental setup and provide more details on the architecture implementation.

The Sensor Gateway software has been tested on HTC G1 and Eris Android phones. The G1 phone has 192 MB RAM and uses the 3G HSPA GSM cellular link. The Eris has 288 MB RAM, a 588 MHz CPU, and runs on a EV-DO rev.A 3G CDMA cellular connection with a maximum nominal data rate of 1.8 Mbps for uplink and 3.1 Mbps for downlink. Both phones have a Bluetooth 2.0 interface used to connect wirelessly to nearby sensors.

The environmental sensors nodes are Crossbow [3] MPR2400CA processor-radio boards (motes) connected to MTS310 sensor boards (having a microphone, temperature, light, magnetic, and acceleration sensors). The motes send measurements via a Zigbee radio to a Base Station mote plugged into a Crossbow MIB600 programming board. This forwards packets using an Ethernet-WiFi bridge to the Sensor Gateway listening to a TCP port on its WiFi interface (Figure 1). The Bluetooth sensor is the Alive Heart and Activity Monitor [1] sampling the heart ECG signal at 300 Hz and 3D acceleration at 75 Hz. The sample resolution is 8 bits. The total number of samples per second originating from this health sensor is 525.

The Sensor Gateway software is implemented in Java on the Android platform, running on the GNU/Linux operating system. Measurements from all internal and external sensors are packed in JSON format and sent to the Storage web service using HTTP. Each measurement is annotated with the node ID, sensor ID, and a timestamp.

The web services are implemented in Java EE and run on the Apache Tomcat 6 application server. Tomcat is an open source implementation of the Java Servlet and JavaServer Pages technologies and is basically a container for running servlet code implementing web services. The three servlets rely on the open source MySql relational database for persistent data storage and retrieval.

We evaluated several application Quality of Service (QoS) metrics. In the first experiment we measured the latency of transmitting the data from the external Crossbow sensors to the Sensor Gateway. The average round-trip-time (RTT) was 58.2 ms, with a standard deviation of 12.36 ms. The packet size sent from the motes has 33 bytes. A constant 9.1 ms overhead from the RTT is due from the serial connection between the WSN Base Station mote and the MIB600 device that bridges the Zigbee WSN with the Ethernet LAN. This serial line has a data rate of 57600 bps that could become a congestion bottleneck for a Base Station receiving packets at a high rate.

The delay test was run over an idle Ethernet LAN and with one mote to minimize background traffic. Most overhead in the RTT can be attributed to the 8 bit Atmel ATmega128 CPU running at maximum 8 MHz.

In the second set of experiments we measured the RTT for web service storage requests carrying sensor data over the 3G CDMA cellular link. Each measurement value is annotated with a timestamp (ms since epoch), node ID, and sensor ID. The measurements are encoded to a UTF-8 JSON object for transmission using HTTP. The number of individual values sent with one request was varied between 10 and 1236 and each test ran for 10 minutes.

The sample average, minimum, maximum, and the standard deviation are shown in Figure 8. The corresponding request size varies from 353 bytes (for 10 values) to 39577 bytes (for 1236 values). On average, each additional value added to the storage request increases the JSON string by 32 bytes. The web service response delay is explained as follows. The uplink cellular connection was measured to have a highly variable TCP data rate, even during a short time interval. The FCC Internet bandwidth test reported uplink data

rates ranging from 251 kbps to 415 kbps (mean: 298 kbps) and ping times between 179 ms and 245 ms (mean: 212 ms) during 3 minutes of testing right after running the web service storage experiment. The ping delay with the cellular link is on average 3 times higher than with an ADSL Internet connection. To mitigate the high delay and low application-level uplink data rate a practical solution is to pipeline multiple storage requests using two or more concurrent worker threads. Using this method, the Sensor Gateway and the cellular link were able to sustain the full 525 samples/s storage load from the Bluetooth heart monitor.

The third set of experiments helped us evaluate and choose the buffering delay for heart ECG and 3D acceleration sensor data delivery. Incoming data packets from the Bluetooth sensor are buffered for a time called buffer delay by the Device Adapter thread listening on the Bluetooth interface. After the buffer delay expires, accumulated data packets are packed to a single entry and added to a data queue shared with the two worker threads responsible with the transmission to the storage web service. Each worker thread blocks on the queue, waiting for a new entry from the Device Adapter. When a new entry is available, a worker thread takes all measurements from the queue entry data packets and encodes them to one JSON string, together with timestamps, sensor, and node IDs. We measured the average JSON encoding time and it is about 0.3 ms per sensor value. The encoding time plus the request RTT are then measured and reported together as the Total Request Delay. This, and the average total delivery delay from sensor to web service are shown in the chart from Figure 9. This total delivery delay includes the buffer delay, the queue waiting time, encoding delay, and the web service call RTT. The optimal buffering delay is 1 second.

We notice that the total delivery delay has a minimum for a Buffer Delay of 1 second. This optimal value depends on the network performance, on the incoming data flow volume, on the CPU encoding performance, and on the number of worker threads. Reducing the buffer delay increases the average queue length and the total number of web service calls, adding more overhead from ping delays. Conversely, increasing the buffer delay reduces the number of separate calls for the same sensor data stream, reducing the overhead from network latency, but adding delay from buffering. A local minimum for the total sensor delivery delay is achievable and recommended for improving application QoS.

As an alternative to the JSON/HTTP web service data delivery method we will consider as future work a binary format that takes less encoding CPU overhead and uses less memory, and that we expected to be more efficient.

4 Conclusions

In this paper we propose a REST-compliant web-based SOA that can be used to monitor patients remotely. To better asses patients' health and environmental conditions, the system uses a heterogeneous WSNs consisting of medical, environmental, and smartphone internal sensors. To validate the system, we implemented a prototype using Android smartphone platform, Crossbow Micaz motes, and Alive Technology hart and activity monitor sensor.

Acknowledgments

This work was supported in part by NSF grant CCF 0545488.

References

[1] Alive Technologies Hart and Activity Monitor, http://www.alivetec.com/products.htm, as of July 2011.

[2] E. Becker, V. Metsis, R. Arora, J. Vinjumur, Y. Xu, and F. Makedon, SmartDrawer: RFID-based smart medicine drawer for assistive environments, *Intl. Conf. on Pervasive Technologies Related to Assistive Environments*, June 2009.

[3] Crossbow Technology, http://www.xbow.com:81/Products /wproductsoverview.aspx, last accessed July 2011.

[4] K. Divi, M. R. Kanjee, and L. Hong, Secure architecture for healthcare Wireless Sensor Networks, *Intl. Conf. on Information Assurance and Security*, Aug. 2010.

[5] E. B. Fernandez and M. M. Larrondo-Petrie, Designing secure SCADA systems using security patterns, *Hawaii Conf. on Systems Science*, Jan. 2010.

[6] R. Fielding, Architectural Styles and the Design of Network based Software Architectures, University of California, Irvine, Department of Informatics, DISSERTATION, 2000.

[7] Y. Hao and R. Foster, Wireless Body Sensor Networks for Health Monitoring Applications, *Phys. Meas.*, Vol. 29, pp. R27-R56, Nov. 2008.

[8] H. Hass and A. Brown, Web Services Glossary, Feb. 2004, available: http://www.w3.org/TR/ws-gloss/.

[9] J. Ko, R. Musaloiu-E, J. H. Lim, Y. Chen, A. Terzis, T. Gao, W. Destler, and L. Selavo, Demo Abstract: MEDISN: Medical Emergency Detection in Sensor Networks, *ACM Conference on Embedded Networked Sensor Systems*, 2008.

[10] LACE: Laboratory for Assisted Cognition Environments, http://www.cs.rochester.edu/u/kautz/ac/, as of July 2011.

[11] K. Lorincz, D. Malan, T. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, and S. Moulton, Sensor networks for emergency response: challenges and opportunities, *IEEE Pervasive Computing*, pp. 16-23, 2004.

[12] J. Modayil, R. Levinson, C. Harman, D. Halper, and H. Kautz, Integrating Sensing and Cueing for More Effective Activity Reminders, *AAAI Fall 2008 Symposium on AI in Eldercare: New Solutions to Old Problems*, Nov. 2008.

[13] C. Otto, A. Milenkovic, C. Sanders, and E. Javanov, System Architecture of a Wireless Body Area Sensor Network for Ubiquitous Health Monitoring, *Journal of Mobile Multimedia*, Vol. 1. No. 4, pp. 307-326, 2006.

[14] J. Waterman, D. Curtis, M. Goraczko, E. Shih, P. Sarin, E. Pino, L. Ohno-Machado, R. Greenes, J. Guttag, and T. Stair, Demonstration of SMART Technology, *Washington DC: AMIA 2005 Annual Symposium*, 2005.

[15] A. Wood, G. Virone, T. Doan, Q. Cao, L. Selavo, Y. Wu, L. Fang, Z. He, S. Lin, and J. Stankovic, ALARM-NET: Wireless Sensor Networks for Assisted-Living and Residential Monitoring, *Technical Report University of Virginia Computer Science Department*, 2006.