Architecture of a Satellite-Based Sensor Network for Environmental Observation

Wei Ye, Fabio Silva, Annette DeSchon and Spundun Bhatt
Information Sciences Institute, University of Southern California

***Abstract*—Sensor webs are a promising technology for future earth science research because of its capability of adaptive observation from a network of in-situ and remote sensors. As important components of sensor webs, in-situ sensor networks have attracted strong attention in recent years. In-situ sensors observe the phenomena being investigated at close proximity, and can be used to calibrate remote sensors. However, despite great technology advancements, there are still many challenges to make sensor networks a turn-key solution for various science applications. We address some of the major challenges by developing a flexible sensor network architecture with a long-term goal to evolve into a sensor web. In this paper, we describe our system architecture and its major components. Our first prototype has been deployed to support an ecological study, and initial results have verified our design principles.**

***Index Terms*—Sensor web, sensor network, satellite communication**

I. Introduction

Sensor webs have been envisioned by the National Aeronautics and Space Administration (NASA) as a powerful future technology for earth science research. Sensor webs enable on-demand, adaptive sensing of a broad array of environmental and ecological phenomena across a wide range of spatial and temporal scales from a heterogeneous suite of sensors, both in-situ and in orbit [1], [2]. In the last few years, there has been significant research activity in designing and developing wireless sensor networks, whose major focus is in-situ sensors that collaboratively perform embedded sensing and communication tasks [3], [4], [5]. Complementing space-based sensors, networks of in-situ sensors become important components of the larger-scale sensor webs.

Despite the rapid advancement in sensor network research, there are still many challenges for scientists to widely adopt this technology. First of all, many applications require that the sensing system be deployed at remote locations without easy Internet access. Second, the system must be flexible to support different sensing capabilities required by different science applications. Third, the system must be robust enough to provide continuous and unattended operation in harsh environments. Finally, intuitive user interfaces and tools are needed for scientists to remotely reconfigure the system and access sensor data.

To address these challenges, we have designed a robust and flexible sensor network with satellite communications, called Sensor Processing and Acquisition Network (SPAN). This paper describes the SPAN architecture and its first prototype that has been deployed to support ecological research. Our SPAN architecture emphasizes a modular and extensible design, such that core building blocks can be reused to develop different scientific observation systems.

We take several approaches to address the major challenges identified above. To support rapid deployment at remote locations, we employ satellite communications as the backhaul to relay in-situ sensor data to a central database. To easily support various science applications, we have developed a unified sensor integration framework that allows streamlined integration of different sensors to the system. Our system therefore supports heterogeneous sets of sensors, from industry-grade products to research-specific prototypes. To ensure robust operation in harsh environments, we have developed mechanisms to monitor system status and recover from potential failures. We also employ extensive data caching to prevent data loss when failures occur.

Our first prototype, instantiating the SPAN architecture, has been deployed at Stunt Ranch in the Santa Monica Mountains. It is being used to support long-term ecological research at the University of California, Los Angeles (UCLA). Initial data collected from this system has verified our design principles. In this paper, we also identify future directions that can translate our design into turn-key solutions for science applications, as well as infusion with other sensor web technologies.

II. System Architecture

In this section, we describe the system architecture of the Sensor Processing and Acquisition Network (SPAN). In the next sections, we will look at the development and deployment of its first prototype.

A. Overview

Figure 1 shows the high-level SPAN architecture, which we envision combining in-situ sensors with space-based sensors. Our current work focuses on in-situ sensors, and our system supports heterogeneous types of sensors, *e.g.,* both wired and wireless, as shown in the figure. The SPAN system is divided into two main parts: the front end and the back end. The front end system consists of all components that are deployed in the field. The SPAN back end, on the other hand, includes all system components deployed in the laboratory, which eventually supply data to the scientists. We employ satellite communication as the wide-area networking (WAN) backhaul to relay data from the front end to the back end, because of its wide availability at remote locations.

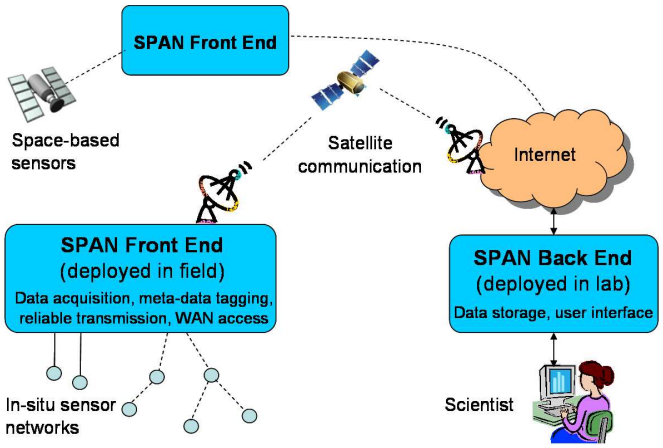


Fig. 1. The SPAN architecture shows potential integration of both in-situ and space-based sensors.

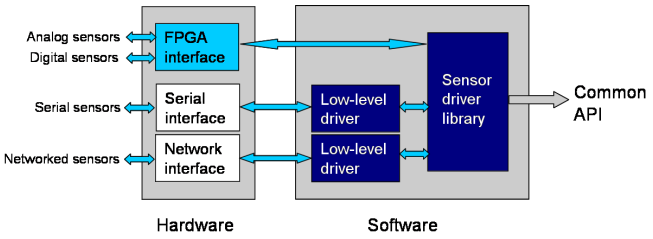


Fig. 2. The unified sensor integration framework hides details of individual sensors, and provide a common API to control sensors.

It is noted that the basic SPAN architecture can be extended to build observation systems with different scales. First, there can be multiple front end systems deployed in the same general area, for example, monitoring a segment along a river. In this case, these multiple front ends can communicate through local wireless networks, such as IEEE 802.11, potentially using directional antennas. This approach allows multiple sites to share a single WAN access point (satellite dish and modem). If multiple front ends are deployed at geographically-distant locations, for example, a shared observation infrastructure deployed by various research groups across several states, our back end is able to handle data feeds from multiple front ends (through multiple satellite links), and bring data to a shared database.

B. SPAN Front End

The major functions in the SPAN front end include sensor management, data acquisition, data and metadata management, reliable data transmission, and WAN access.

a) Sensor management: There are several tasks in sensor management. First, when application requires specific types of sensors, our system will fully *integrate* them. After a sensor is integrated, we provide the capability to *control* the sensor (*e.g.*, enable or disable) and to *configure* its sensing parameters (*e.g.*, sampling rate, raw data or average).

A major challenge in sensor management is the need to support various types of sensors for different science applications. For example, there are simple analog or digital sensors. There are also complex sensors that require the use of an elaborate commanding scheme over a serial port or other digital interfaces. Wireless sensors, such as motes, are even more complicated, because they are distributed at different locations connected with short-range, unreliable wireless communication.

To deal with this challenge, we developed a unified sensor integration framework, as shown in Figure 2. The major objective of the framework is to hide all the details of individual sensors, and provide a common application programming interface (API) to control and configure sensors in a unified fashion. Different sensors are connected to the system through different interfaces. The FPGA interface implements low-level functions required to interface with some analog and simple digital sensors. Some examples include implementation of basic pulse counters, frequency measurements, and simple control options for enabling sensors. It also supports the IEEE 1451 transducer electronic data sheet (TEDS), so that sensors supporting this standard can have their metadata obtained automatically when they are connected into the system. Complex sensors require low-level drivers, which implement the sensor-specific commands required to make measurements, and configure specific sensor parameters. At the higher level, we develop a sensor driver library that provides a common API to control different sensors.

b) Data acquisition: Data acquisition is the actual process of taking sensor readings. It can happen either on-demand or based on regular scheduled intervals. For analog sensors, appropriate analog-to-digital (A/D) conversion is performed with sufficient precision. Most sensors require calibration and unit conversion, which converts raw data to data in meaningful scales. Our system support different types of calibration methods, such as linear, polynomial and binary. If the application requires processed data, for example, average, max or min, the data acquisition component will carry out such processing as well.

c) Data and metadata management: Data management supports coordinated data collection. We create a unified command API that can be used to record data from a wide variety of sensors. It allows the scientists to easily configure or reconfigure the system. It also supports automatic sensor reconfiguration triggered by external sensing events.

Metadata is important to any sensing system, because it provides necessary context for the measured data. Our system manages metadata effectively. Metadata specific to each individual measurement, such as timestamps, is tagged when the measurement is taken and transferred with the sensor data. Sensor specific metadata, such as make, model, serial number, measurement type and unit, *etc.*, are stored in the database and updated only when changes occur.

d) Reliable data transmission: Reliable data transmission is important for science applications. It is required for both local wireless (*e.g.*, motes or IEEE 802.11) and WAN communications. Besides using reliable data transfer proto-

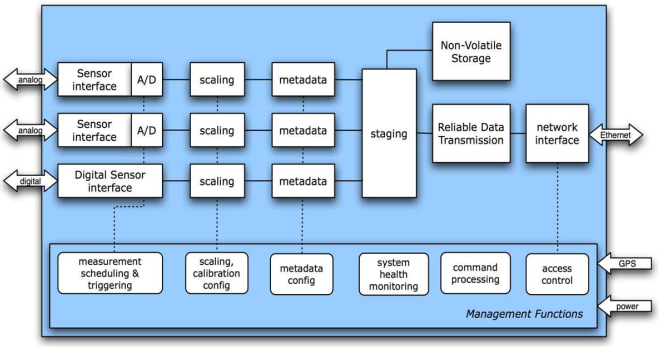


Fig. 3. Major functions in the SPAN front end.

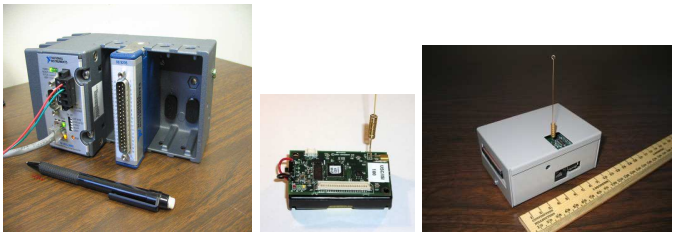


Fig. 4. Left: CompactRIO from National Instruments as the data acquisition platform. Middle: Mote from Crossbow, Inc. as wireless communication platform for distributed sensors. Right: Stargate from Intel as the WAN access controller.

cols, such as TCP or delay/disruption tolerant networking (DTN), we perform extensive local data caching. When sensor data is collected, we provide local storage before transferring it to the back end. Therefore, when a failure occurs during the transmission, data can be recovered from the local cache.

e) WAN access: In the current SPAN architecture, we employ satellite communication as the WAN backhaul method. Due to our modular design, the system can also support other WAN technologies, such as cellular network or WiMAX. The WAN access component performs three important tasks. First, it monitors the availability of the satellite link and potential IP address changes. Long-range wireless communications suffer from low-quality and/or intermittent links, which may largely affect the performance of our system and interrupt continuous data transfer. Second, the WAN access component manages the use of the satellite link. When the front end is powered with batteries or solar power, the WAN access component can duty cycle the satellite modem to conserve energy. If multiple front ends share the same satellite link, the WAN access component coordinates their activity as a base station. Finally, the WAN access component provides basic security measures, such as firewall and access control.

We have described the major functions in the SPAN front end. We summarize their relationship in Figure 3. It should be noted that these functions can be implemented over a distributed set of equipment rather than on a single platform.

C. SPAN Back End

The SPAN front end collects data in the field, and transfers them to the back end in the laboratory through satellite communication. The major functions in the back end include data storage and user interfaces.

a) Data storage: The SPAN system stores all sensor data and related metadata in a database. The database can either be privately owned by a scientist or one that is shared by a scientific community. The SPAN back end system has a data relay component that receives data from the front end and handles the interaction with the database. For example, when the database is busy or otherwise not available, the data relay component can either cache data locally or ask the front end to slow down or pause the data transfer.

b) User interfaces: User interfaces and tools are important for scientists to use the SPAN system, since they are usually not software or networking experts. We provide three types of interfaces. A command interface allows users to control and reconfigure the system remotely. For example, the user can start or stop a sensor, or change its sampling rate. A data interface allows users to easily access sensor data and metadata from the database. Finally, a status interface can be used to display system status information, such as component failures on the front end, or the availability of the satellite link. These interfaces hide most system complexities from the users, and provide them with an intuitive environment to work with the system.

III. Prototype Implementation

The above section describes the architecture of our SPAN system, including both front and back ends, as well as their interaction. Now we describe an instantiation of the architecture with our first prototype.

A. Platform Consideration

The major front end platforms include sensors, the data acquisition platform, and the wireless communication platform. Sensors are largely application specific. However, some sensors, such as weather stations that combine a number of meteorological measurements, are common for many environmental monitoring applications. Most industry sensors have a wire connection with some form of interface, for example, analog, digital, serial, etc. Different sensors also have different sampling rates, accuracy, and resolution, which require a sufficient number of bits in A/D converters. Such large variation of sensor properties requires a versatile data acquisition platform that has rich sensor interfaces and the capability of real-time data sampling over multiple sensors. After reviewing different options, we have selected the CompactRIO from National Instruments [6], as shown in Figure 4 (left).

The CompactRIO (cRIO) is a rugged platform that can survive in harsh environments (rated from -40 to 70°C). It has a reconfigurable chassis that allows users to plug in different modules to interface to different sensors. Some modules support the IEEE 1451.4 transducer electronic data sheet (TEDS). On the software side, the cRIO runs a real-time operating system (RTOS) to meet the requirements of

high-frequency sensing and real-time control tasks. The cRIO has a built-in TCP/IP networking stack, which is ready for networked data collection and transfer. Combined with the RTOS, the graphical programming language, LabVIEW, is used to develop different applications.

Although many industry sensors have a wire connection, the new trend in distributed sensing is to employ wireless communication. This allows much more flexible sensor deployment than what can be achieved with only wired sensors. Wireless sensors form a short-range, multi-hop network that relays data from each sensor to a base station. Since wireless sensors are mostly powered by batteries, energy efficiency is a major consideration when selecting wireless communication platforms. IEEE 802.11-based radios are powerful, but require relatively high power. In comparison, the motes designed by UC Berkeley and Crossbow Inc. are an excellent low-power alternative. Motes are typically microcontroller-based platforms integrated with a low-power radio, such as the IEEE 802.15.4 (ZigBee). An example mote is shown in Figure 4 (middle) [7].

For wide-area networking (WAN) technologies, we have compared different satellite communication choices, such as WildBlue, HughesNet, and the NASA's Geostationary Operational Environmental Satellite (GOES) system. Among them, WildBlue provides the commercial service of Internet over satellites that has the best combination of coverage, bandwidth and price. By installing a small satellite dish and a communication modem, we are ready to connect to the Internet from almost any location within the continental US.

To control the satellite modem and manage the WAN link we employ a Stargate, which is a general-purpose embedded PC designed by Intel. This general-purpose computer, shown in Figure 4 (right), runs the popular Linux operation system, and allows us to easily implement all the functionality of the WAN access point described in Section II-B. These functions are more difficult to implement over the cRIO. Moreover, there is abundant open-source software that can be directly integrated into our system. For example, the DTN reference implementation [8] and the routing software for motes and ad hoc networks are all available from the community.

The back end system is deployed in the lab, so we selected a general-purpose Linux workstation to host different application software. With Linux, we select an open-source, reliable database, MySQL, to store sensor data and metadata. Responding to the scientists' requirements, our first prototype provides the capability for scientists to share their data in a community database, called SensorBase, which is developed by UCLA. SensorBase provides scientists with a friendly front end to the MySQL database, allowing easy database maintenance, table creation, and data visualization and sharing. We have implemented the data relay component as a UNIX daemon process that receives data from the front end and inject it into SensorBase. There are also different tools available for monitoring system status. We choose Nagios due to its comprehensive monitoring capability and intuitive user interface. Nagios can also be configured to send e-mails, or page a scientist, in case system failures are detected.

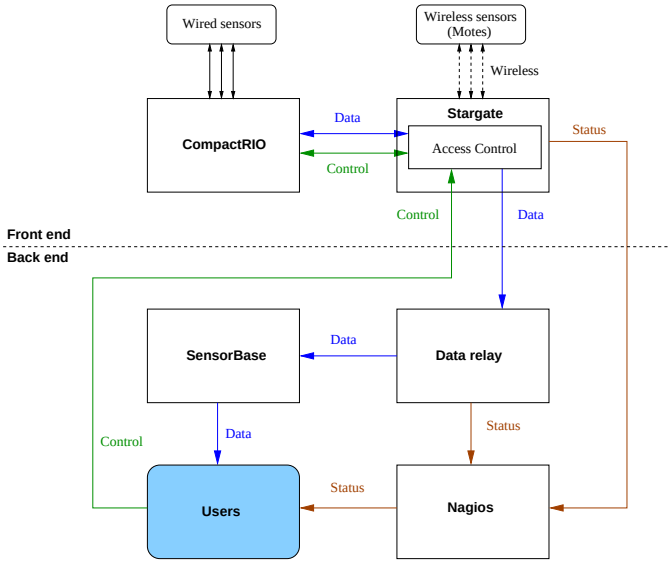


Fig. 5. System integration diagram of our first prototype.

B. System Integration

Figure 5 is a block diagram of our first prototype, showing the relationship of all components after system integration. At the front end, wired sensors are connected to the cRIO, and wireless sensors (motes) relay their data to their base station, the Stargate. The Stargate provides access control and manages the usage of the satellite link. Sensor data is transferred to the data relay component, which then uploads it into MySQL database using the same SensorBase interface that scientists do. In addition to the sensor data flow in the system, there are two types of communication flows in the system. Nagios collects system health information (status) from different components in the system, and provides an intuitive user interface accessible via a web browser. Finally, users can reconfigure the front end through the control channel. This first prototype is completely implemented as shown in the figure.

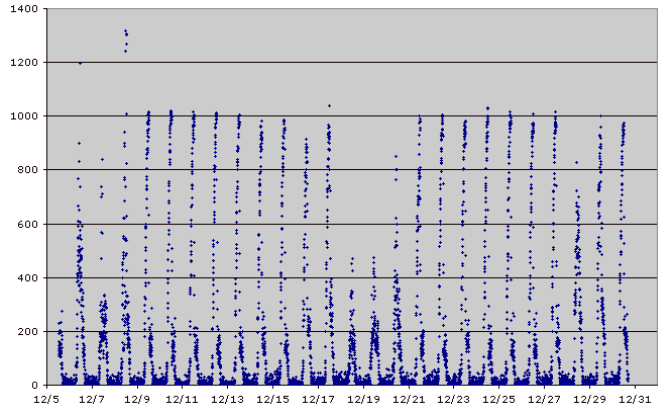
IV. Lessons Learned from Initial Deployment

Figure 6 shows the partial deployment of our first prototype at Stunt Ranch in the Santa Monica Mountains. The WildBlue satellite dish is located near the top of the pole, and is positioned in such a way that it does not affect the nearby sensors. Below it we have temperature, humidity, photosynthetically active radiation (PAR), wind speed, and precipitation sensors. The cRIO, Stargate, and satellite modem are installed in the enclosure box. All components on the pole are powered by line power, which is available on the site.

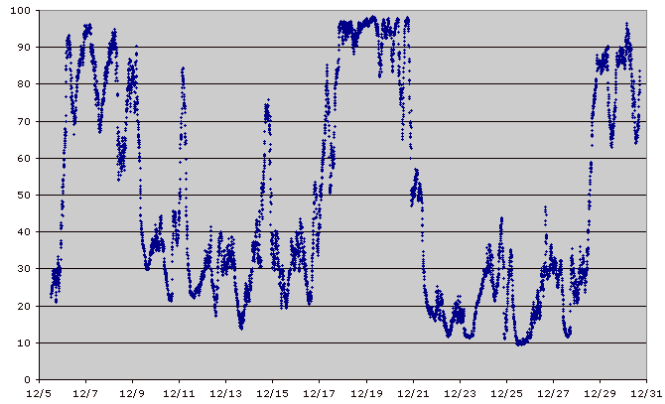
A. Scientific Application: Ecological Study

In the deployment of our SPAN system, scientists are interested in the long-term investigation of the influence of the 2006–07 southern California drought conditions on the water relations of important chaparral shrub and tree species

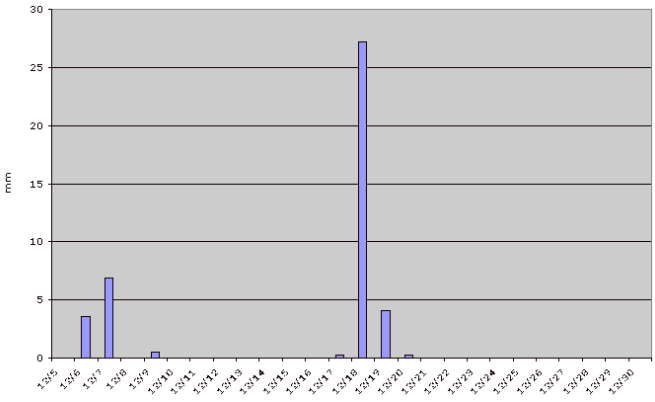
Fig. 6. Our first prototype has been deployed at the Stunt Ranch in the Santa Monica Mountains to support ecological research.



(a) Photosynthetically active radiation (PAR)



(b) Relative humidity



(c) Precipitation

Fig. 7. Example sensor data collected during December 5–31, 2007.

that differ in their depth of rooting. Rainfall over this past hydrologic year in southern California has been less than 25% of normal, making it the driest year on record. In addition to core measurements of air temperature, relative humidity, wind speed, solar radiation, rainfall, and soil moisture, we use sap flow sensors to continuously monitor the flow of water through the xylem system of replicated stems. Sap flow methods have been used to quantify water use by natural vegetation, forest plantations and crop plants, to determine how water uptake by trees influences groundwater discharge, and to determine the effects of atmospheric and other environmental variables on transpiration by individual tree species.

At Stunt Ranch, the sap flow sensors are distributed at several selected plants. The scientists currently selected four species to compare their access to soil moisture with plant water stress. At each plant we use a mote, equipped with a sensor board, featuring 6 A/D channels (with a 24-bit resolution), that connects to several sap flow sensors on that plant. The mote collects data from all sap flow sensors on the same tree and sends the sensor data back to the base station (Stargate) potentially over multiple hops. We use a car battery at each plant to power the sap flow sensors.

B. Initial Sensor Data

Our prototype system has been successfully running at the Stunt Ranch for several months. We have successfully collected data from all sensors deployed. With the satellite communication, we are able to retrieve all sensor data in real time and save it in the database. We have also experienced periods of time when the satellite link is not reliable or completely down. Our system was able to avoid data loss by recovering data from local caches. The prototype demonstrated the effectiveness of our architecture design.

Figure 7 shows an example of sensor data collected from our first deployment at Stunt Ranch. The data was collected from December 5 to December 31, 2007. Figure 7(a) to Figure 7(c) are samples taken from PAR, relative humidity and precipitation sensors, respectively. These figures show

interesting correlation on sensor readings. For example, around December 19, the low solar radiation readings, as shown in Figure 7(a), can be explained by a rain event, as illustrated by Figure 7(c), which was also responsible for the high relative humidity around these days, see Figure 7(b).

On the other hand, we have also discovered a few issues that need to be further addressed. For example, even though there is line power in the field, it is not safe to assume that the power is always available. In our case, we had a few instances where the power was cut for unknown reasons. As a result, the system stopped working, and to make it worse, there was no way to find out remotely, from the back end. To address this issue, we plan to add a backup battery, which allows the system to send emergency notification messages.

V. Conclusions

Sensor webs are a promising technology to support environmental and ecological research, by combining in-situ and space-based sensors, and by providing adaptive reconfiguration capability. As a step towards the sensor web vision, this paper presents the architecture of a sensor network that uses satellite communication to transfer data from remote sensors to the laboratory. Although our current focus is on in-situ sensors, the architecture reflects our long-term goal of combining in-situ sensors with space-based remote sensors. We plan to explore this direction in the future.

To validate our system architecture design, we have implemented our first prototype and deployed it at Stunt Ranch to support ecological research. Initial testing shows promising results, and has validated our design. We plan to further collaborate with the scientists on implementing specific use cases that require event-triggered system reconfiguration.

Acknowledgments

This work is supported by NASA through the ESTO's AIST program under grant number NNX06AE46G as the Satellite Sensornet Gateway (SSG) project. We thank Aaron Falk for his early contribution to this work. We thank Philip Rundel and Eric Graham from the Center for Embedded Networked Sensing (CENS) at UCLA for their collaboration and support on the deployment of our first prototype at Stunt Ranch. We also thank Yeung Lam at UCLA for providing his initial implementation of the data relay daemon in the back end.

References

[1] E. Torres-Martinez, G. Paules, M. Schoeberg, and M. W. Kalb, "Comparing energy-saving MAC protocols for wireless sensor networks," *Acta Astronautica*, vol. 53, no. 4–10, pp. 423–428, 2003.

[2] R. Sherwood, K. Moe, S. Smith, and G. Prescott, "Advances in sensor webs for nasa earth science missions," in *Poster at the American Geophysical Union (AGU) Fall Meeting*, San Francisco, California, USA, Dec. 2007.

[3] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in *Proceedings of the fifth annual ACM/IEEE international conference on Mobile Computing and Networking*, 1999, pp. 263–270.

[4] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," in *Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems*. Cambridge, MA, USA: ACM, Nov. 2000, pp. 93–104.

[5] G. Tolle, J. Polastre, R. Szewczyk, N. Turner, K. Tu, S. Burgess, D. Gay, P. Buonadonna, W. Hong, T. Dawson, and D. Culler, "A macroscope in the redwoods," in *Proceedings of the 3rd ACM SenSys Conference*, San Diego, CA, USA, Nov. 2005.

[6] National Instruments Co., "Data sheet: Ni compactrio specifications," http://decibel.ni.com/content/docs/DOC-1660.

[7] Crossbow Technology Inc., "Mica2 data sheet," http://www.xbow.com/.

[8] Delay-Tolerant Networking Research Group (DTNRG), http://www.dtnrg.org/.