Wireless sensor seismic response monitoring system implemented on top of NEESgrid

John-Michael Wong*, Jan Goethals, Bozidar Stojadinovic
University of California, Berkeley, Dept. of Civil and Environmental Engineering, 760 Davis Hall, Berkeley CA, USA 94720-1710

ABSTRACT

Emerging wireless sensor technology presents a tremendous opportunity for developing low-cost monitoring systems for civil infrastructure. A set of wireless MICA2 mote accelerometers were deployed on a reinforced concrete bridge column to collect data during a shaking table earthquake simulation test. The data from the test were ingested into a data and metadata system for testing the usability of the metadata description for data mining and visualization. The metadata system used the NEESgrid metadata services interface for backend storage. The results of the data acquisition and ingestion were used to determine the feasibility of using a comprehensive sensor, data storage, and data retrieval system for health monitoring.

Keywords: civil engineering, wireless sensor, mote, mica2, infrastructure, seismic, monitoring, TinyOS, earthquake

1. INTRODUCTION

A framework for creating a structural monitoring data repository was created for acquiring, storing, classifying, and displaying sensor data. The data acquisition system implemented as part of that framework uses wireless sensors based on the MICA2 platform. The details of the sensor system and an earthquake simulator shake table test are described in this paper.

The MICA2 platform[1] was chosen for evaluation because of its low-cost and low-power consumption components. Several other projects have used this sensor platform for structural health monitoring applications[2]. The sensors use MEMS accelerometers manufactured by Crossbow Technology, Inc. The sensors are equipped with an onboard microprocessor running the TinyOS operating system[3]. The processor allows the user to program intelligence into the sensor. The sensors use a self-organizing ad-hoc network to communicate data back to the base station.

A special application was written for the TinyOS platform specifically for acquiring data in a laboratory testing environment. This control program was uploaded into the flash memory of each sensor. A base station mote was connected directly to the data acquisition computer to communicate with the accelerometers. Commands to start and stop data recording, increase and decrease radio transmission power, and to stream buffered data to the base station were sent to the motes for testing control.

Six different tests were run on the test specimen. An elastic range test, yield level test, design level test, maximum level test, a 7% test, and a 70% test were performed. Failure of the specimen occurred after the 70% test. White noise tests were performed before each test in all three directions.

The results of the experiment showed that the radio interference and subsequent packet loss were the largest sources of error in the acquisition system. Resolution problems also posed a significant source of error. With one mote, the amount of packet loss is negligible. However, with two different motes streaming data to the base station simultaneously, packet loss increased to 25%. Packet loss with three motes increased to nearly 40%. Because of the high amount of packet loss, the data storage and retrieval system could not be used effectively for all the sensors in real time. However, near real-time data might be collected accurately if each sensor's data is transmitted individually to the base station from the on-board buffer, or if better transmission methods are developed.

* jmwong@ce.berkeley.edu; dunamis.ce.berkeley.edu

Health Monitoring and Smart Nondestructive Evaluation of Structural and Biological Systems IV, edited by Tribikram Kundu, Proc. of SPIE Vol. 5768 (SPIE, Bellingham, WA, 2005) 0277-786X/05/$15 • doi: 10.1117/12.599860

2. SENSOR DESCRIPTION

2.1 Mica 2 platform sensor

The wireless sensors used were based on the MICA2 platform designed by UC Berkeley researchers and commercialized by Crossbow Technology, Inc. The MICA2 platform uses low-power and low-cost components that allow for prolonged, wireless, and intelligent sensing. Each MICA2 module contains a microcontroller and radio (Fig. 1). The radio handles communication between the sensor and base station. The onboard microprocessor is programmed using TinyOS-based code. Although many different sensor boards can be attached to the module, only the two-axis MEMS accelerometer embedded on the Crossbow MTS310 sensor board was investigated in this experiment.

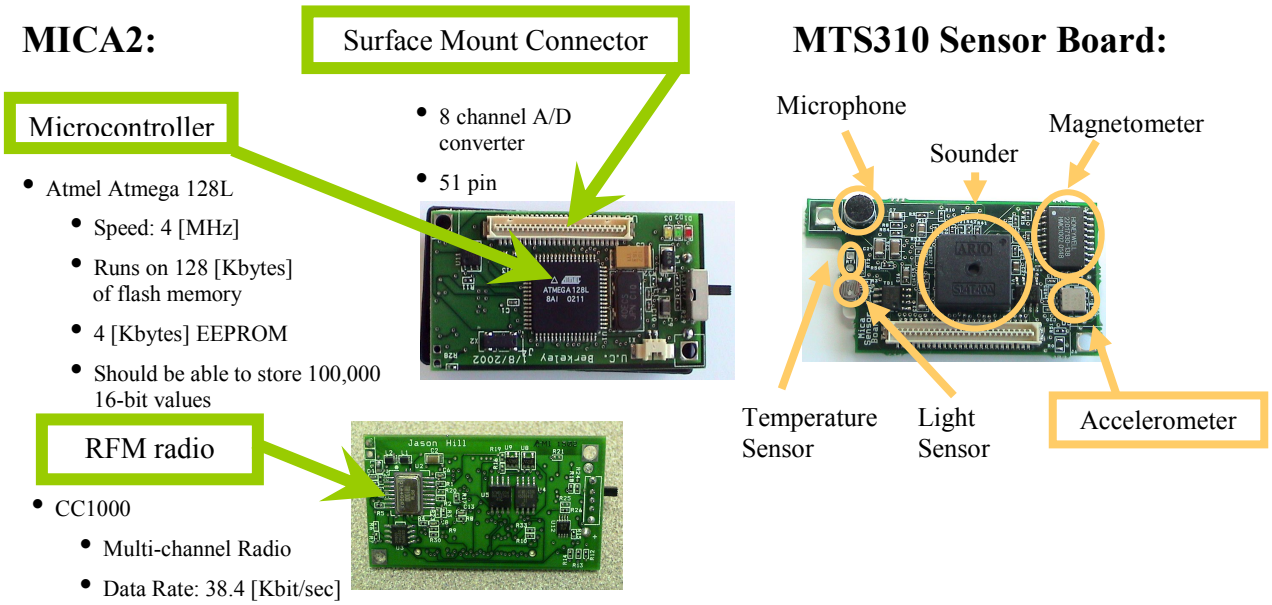

Fig. 1. MICA2 components and MTS310 sensor board

The MICA2 board uses a Chipcon CC1000 radio transceiver. This radio provides the option of operating on a center frequency of 916 MHz, 433 MHz, or 315 MHz. For this project, the radios were all configured to use the 433 MHz frequency. Multiple channels can be programmed in each frequency band. The radio provides a maximum data rate of 38400 bits/sec. The outdoor transmission range is estimated to be about 1000 ft. for the 433 MHz range and about 500 ft for the 916 MHz range.

The MICA2 components are controlled by the Atmel Atmega 128L, a low-power, 4 MHz microprocessor. The microprocessor runs the TinyOS operating system that allows the user to program computational tasks on a power and performance constrained processor. The TinyOS operating system, libraries, and applications are written in nesC, a specially developed language with a C-like syntax[4]. TinyOS is an event-driven operating system where executables are built from modular components linked by well-defined, bidirectional programming interfaces.

2.2 Accelerometer

For this project, the software applications and data acquisition routines were written specifically to work with the Analog Devices ADXL202JE accelerometer[5] built into the Crossbow MTS310 sensor board. The accelerometer is a MEMS 2-axis, ±2 g device (Fig. 2). It was chosen for this application partially for its very low current draw less than 1 mA. The accelerometer provides an analog signal which is passed through an on-board analog to digital converter. Although the characteristics of this accelerometer do not provide the quality necessary for detailed structural monitoring, it was sufficient enough demonstrate the feasibility of using these kinds of sensors for structural health monitoring applications.

Model Number	ADXL202JE
Range	±2 g (1 g ≈ 9.8 m/s^2)
Resolution	0.002 g (RMS)
Sensitivity	167 mV/g ±17%
Offset	2.5V ±0.4V

Fig. 2. Accelerometer specifications

3. DATA ACQUISITION SYSTEM

3.1 Sensor communication system

Data collected by the accelerometers is transmitted to a MICA2 base station (Fig. 3). The base station is connected to a computer via an RS-232 serial port. The base station radio enables commands to be sent to the MICA2 sensors from the control computer. TinyOS applications running on the base station computer issue commands and handle incoming events. For this project, a conventional laptop computer running Windows XP and the Cygwin operating environment was used as the base station computer. Together, a single MICA base station, computer, and the MICA2 sensor boards comprise the hardware components for the wireless data acquisition system. A more sophisticated system could involve more interface boards handling multiple MICA2 networks running on different frequency bands.

Fig. 3. MICA2 base station mote

The software components for the data acquisition system involve components written in nesC for the TinyOS platform as well as Java and MATLAB applications. There are three major components in this software system:

1. TinyOS programming to control the operation of the sensor board. This involves control of the LEDs, accelerometer, memory, and radio through the onboard microcontroller;
2. A Java application running on the base station computer that reads and writes data packets through the serial port and forwards them through the base station radio.
3. Java and MATLAB programs to construct the command packets that give instructions to the sensor boards, and for parsing and storing the incoming data packets from the accelerometers.

MicaControl is the name of the TinyOS application that runs on the onboard microcontroller. It is configured to listen for command packets from the radio and performs an action based on the content of the command packet. It responds to eight different commands. Two of the commands control the state of the sensor board LEDs, two commands control radio power output, and four commands control the recording and transmission of accelerometer data.

The SerialForwarder is the component that sends packets from the computer through to the base station module radio (Fig. 4). It is part of the TinyOS core Java programs. SerialForwarder works by listening for connections on a specified port which applications can connect to over the Internet. It forwards the packets between the network socket and the serial port with the base station MICA2 module. This makes it possible to interact with the MICA2 sensor network to be accessed remotely. Multiple clients can connect simultaneously since the SerialForwarder sends a copy of each message from the sensor network to each client.

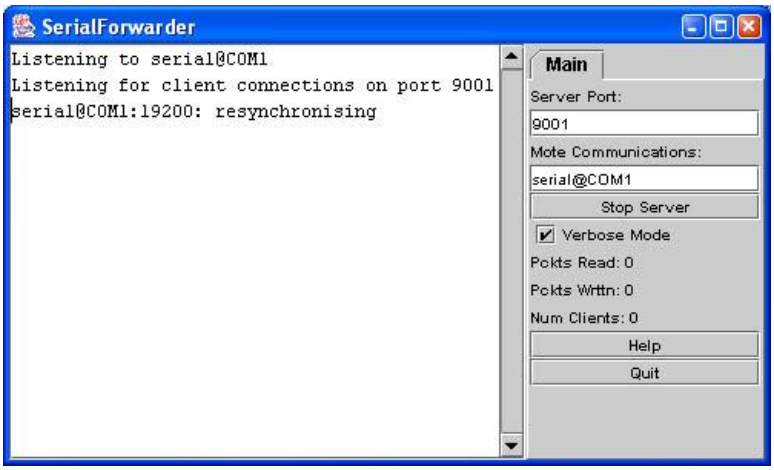

Fig. 4. SerialForwarder application window

A MATLAB program which connects to the SerialForwarder was used initially to test sending command packets, receiving data packets, and to perform basic processing on the incoming data. After that, more sophisticated Java applications were written to perform similar functions. The broadcast packet injector (BCastInject.java) provided in the core TinyOS distribution was modified to send command packets that can be understood by MicaControl. The class DataCollect.java parses incoming packets and stores the sensor readings in a standard format for storage and retrieval.

The TinyOS Oscilloscope application was modified to interface with MicaControl for graphing accelerometer readings (Fig. 5). The GUI window allows the user the send commands to the sensors and view and save data as it streams in. Text fields are available to input the number of samples to be collected by the accelerometer and the sampling rate. The "Start Sensing" button sends the appropriate command packet to tell the sensor to record data in its internal buffer and then transmit it over the radio.

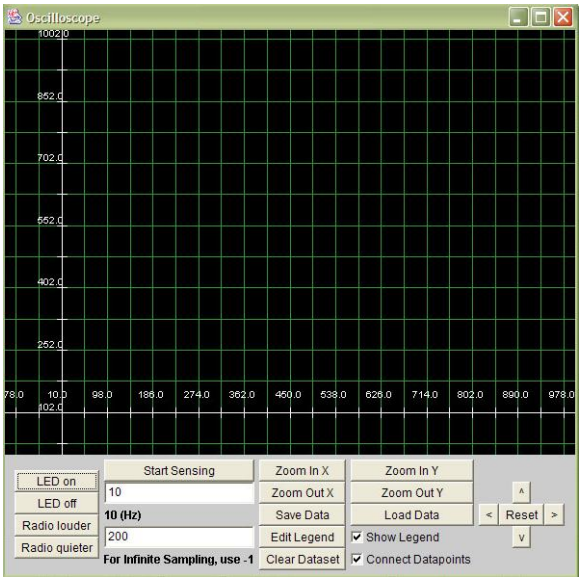

Fig. 5. Oscilloscope application

3.2 Integration with NEESgrid-based metadata storage

Once acceleration data is collected, it is stored in a plain ASCII text file of comma separated values. A header on each file records the date and time when the file was saved, the number of samples collected, and the channel number of the

sensor. After the data is stored in a file, the metadata for the sensor data can be generated and ingested into the metadata repository using the NEESgrid metadata service (NMDS) as the backend storage system.

Six different classes of metadata about the structure need to be entered into the monitoring data storage and visualization system. Descriptions of the structural alerts for tracking threshold values and overall structural state need to be defined. Information about the data files collected by the sensors needs to be input telling the locations and content of the data files. Timestamps of events when data was collected indicate how to group the data sets by time. General information about the structure like its name, location, dimensions, and photographs go in as static data. Locations are defined which describe where sensors are placed on the structure. Lastly, information about the sensors is input assigning each sensor a unique name, a location in the structure, and characteristic properties.

Once the metadata description about the structure and the monitoring data is ingested into the visualization and storage system, then it automatically generates web pages for browsing and searching the data (Fig. 6). Information can be browsed in a variety of different ways. For example, the user can look for sensor data by specific location in the structure, by triggered alerts based on threshold monitoring values, by sensor name, or by time. The system is able to query data in a variety of different ways based on the structure of the data model.

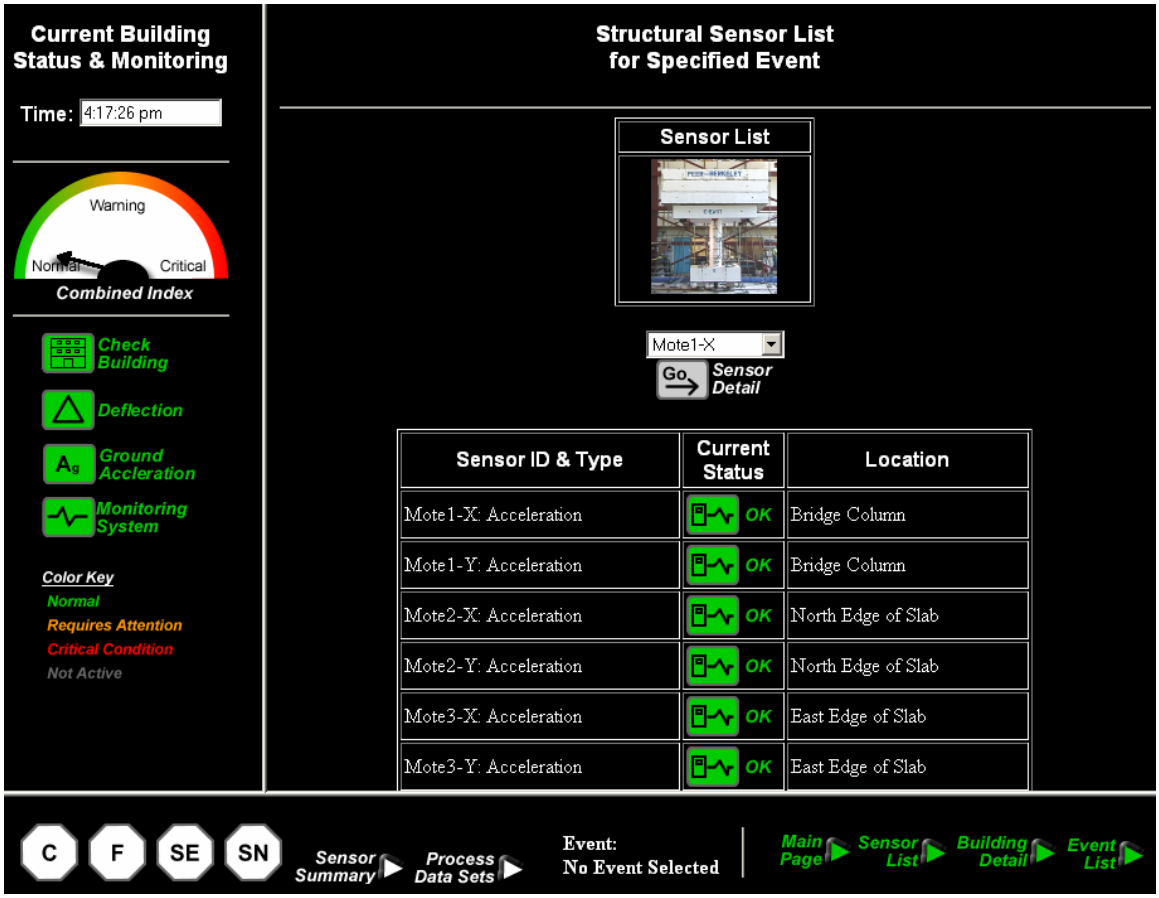

Fig. 6. Monitoring web interface

4. EARTHQUAKE SIMULATOR TESTING

A shaking table test of a reinforced concrete bridge column was used to test the operation of the wireless data acquisition system with the MICA2 sensors. The column is a 1/3-scale model of a typical spiral reinforced concrete column used to construct new highway bridges in California. The column was subjected to bi-directional horizontal and vertical ground motion excitation in several stages with varying intensity.

4.1 Equipment setup

One MICA2 module acted as the base station connected to a serial port. Three other MICA2 modules connected to Crossbow MTS310 sensor boards were used for the data collection. Two of the sensors were placed on the bridge deck part of the specimen and the third sensor was placed on the column (Fig. 7). The base station was connected to a PC laptop running the TinyOS control applications. The base station and laptop were placed within 30 ft of the specimen.

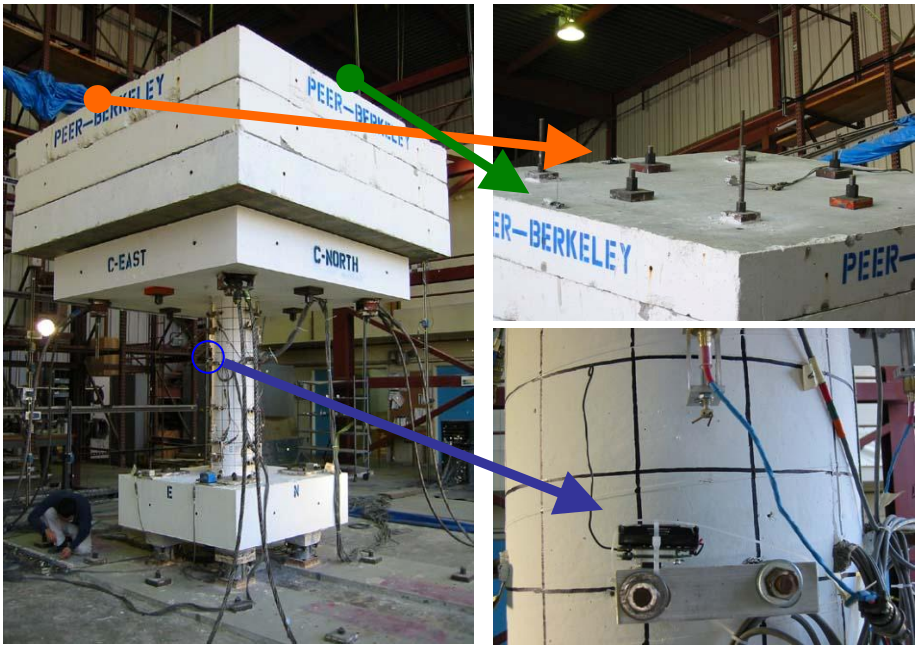

Fig. 7. Sensor placement on test specimen

Before the sensors were installed on the specimen they were first programmed with the MicaControl program to respond properly to the command packets. Each sensor board was connected to the computer then the nesC source code for the microprocessor application was complied and stored in the onboard flash memory. The sensors were each assigned a unique channel id number during their programming. The base station MICA2 unit was programmed with the TOSBase program which allows the mote to receive radio signals and forward them onto the serial port, and also to broadcast serial port signals over the radio. TOSBase is part of the core TinyOS distribution.

4.2 Data collection

Before the shake table test, each of the motes was run through a series of calibration tests. The calibration tests determined signal properties needed to obtain the correct readings from the sensor digital output. After that, the earthquake test stages were run on the specimen: an elastic range test, yield level test, design level test, maximum level test, and a final test round at 7% at 70%. White noise tests in the x, y, and z directions were conducted before each shaking table test. Problems with noise and packet loss prevented meaningful data from being collected during some of the white noise tests in the yield level stage and the x-direction white noise test in the design level phase. The shake table tests with their data time stamps are listed in Fig. 8.

Calibration	
Calibration of Motes 1, 2, 3	June 10, 2004
Elastic Range Testing	
White noise in X-direction	June 14, 2004, 17:10
White noise in Y-direction	June 14, 2004, 17:11
White noise in Z-direction	June 14, 2004, 17:13
Elastic test	June 14, 2004, 17:38
Yield Level Testing	
White noise in X-direction	No data
White noise in Y-direction	No data
White noise in Z-direction	No data
Yield test	June 15, 2004, 10:35
Design Level Test	
White noise in X-direction	No data
White noise in Y-direction	June 15, 2004, 11:27
White noise in Z-direction	June 15, 2004, 11:30
Design test	June 15, 2004, 11:51
Maximum Level Test	
White noise in X-direction	June 15, 2004, 13:59
White noise in Y-direction	June 15, 2004, 14:00
White noise in Z-direction	June 15, 2004, 14:02
Max level test	June 15, 2004, 14:07
Final Tests	
White noise in X-direction	June 15, 2004, 15:35
White noise in Y-direction	June 15, 2004, 15:36
White noise in Z-direction	June 15, 2004, 15:37
7% test	June 15, 2004, 15:43
70% test – [specimen failure]	June 15, 2004, 15:47

Fig. 8. Shake table test sequence

Possible packet loss issues were examined by running a series of radio transmission tests. This was necessary to diagnose the effect of radio irregularities when running the MICA2 system[6]. The rate of packet loss varied significantly depending on the transmission range, sampling rate, and the number of sensors transmitting simultaneously. The number of lost packets increased with a longer transmission range. However, within a 250 ft. radius, distance did not contribute significantly to packet loss.

The number of lost packets increased with higher sampling rates. The percentage of packet reception was nearly 100% at less than 50 Hz, but dropped to 50% at 100 Hz (Fig. 9). Although the sensors and onboard accelerometers are capable of very high internal sampling rates, the reliability of data transmission decreases the ability to use those higher rates for data streaming. It is possible that inefficiencies in the data acquisition software which uses the TinyOS building blocks might also contribute to the packet loss effect.

The rate of packet loss increased as more sensor motes communicated simultaneously. The hardware and TinyOS applications are designed to allow the motes to communicate at the same time. However, the actual performance shows a significant increase in packet loss with multiple motes (Fig. 10). The packet loss with two different motes transmitting data simultaneously was around 25%. Three motes transmitting under the same conditions yielded a packet loss of nearly 40%.

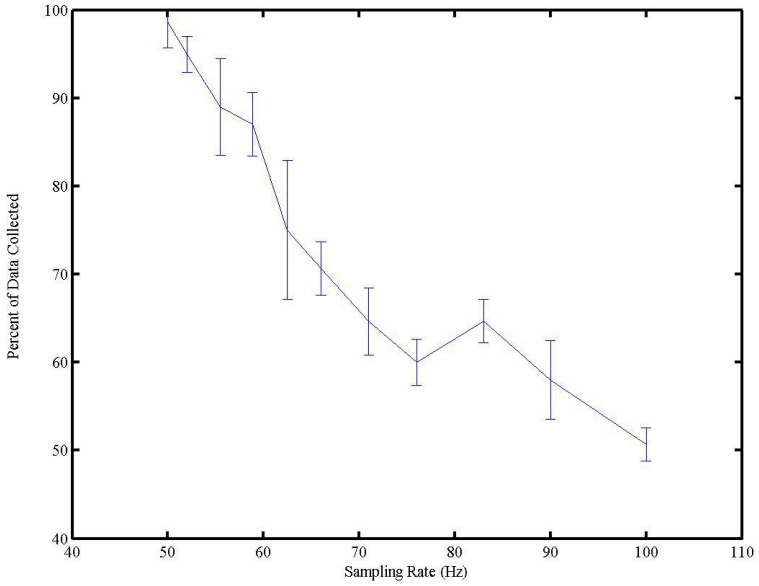

Fig. 9. Percentage of data collected at different sampling rates

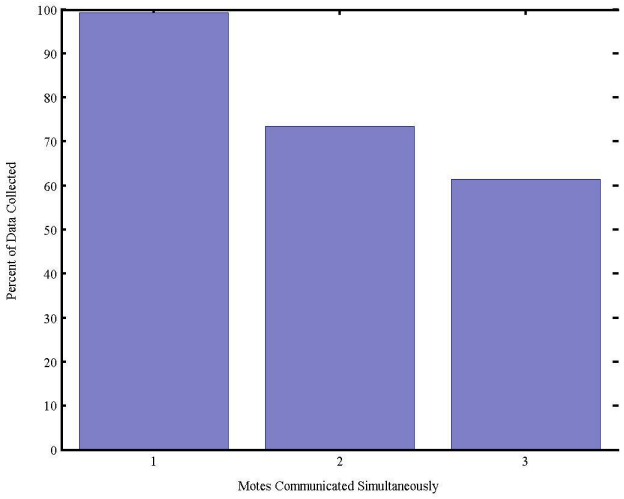

Fig. 10. The effect of multiple motes communicating simultaneously

The sensors were each powered using two standard AA size batteries for a 3V input. The batteries had to be changed for each day of testing. Using the same batteries for more than two days would have resulted in a drop in the power source voltage causing radio transmission problems leading to packet loss. The MICA2 motes eventually just shutdown when there is insufficient voltage from the batteries. One set of batteries was sufficient to power the motes during the four hour testing duration on the second day. For any long-term structural monitoring, other methods of powering the motes would need to be implemented. It is simply not feasible to keep changing the batteries due to their short life time.

4.3 Data storage and visualization

The data collected from the acceleration sensors was input into MATLAB for analysis and visualization as well as ingested into the metadata-based storage system for access through the web interface. The data files were saved in a standard comma separated values file and uploaded on a web server to allow access over the network. Metadata was written for each data file from each test in the NEESML format[7]. An example entry from the Elastic level test is shown in Fig. 11. This process can be easily automated by generating the NEESML files programmatically, or by writing Java classes that can input the information directly into the repository.

```
<data id="Elastic.1X" name="Elastic.1X" container="nees:rootContainer">
        <name string="Elastic.1X"/>
        <producedBy string="Mote1-X"/>
        <dataUnitsX string="sec"/>
        <dataUnitsY string="cm/s^2"/>
        <dataFile   string="http://dunamis.ce/ucbdata/Elastic 1X.csv"/>
        <timeStamp  string="2004-06-14T17:38"/>
        <dataType   string="Acceleration"/>
        <locatedAt  string="Column"/>
</data>
```

Slot Name	Description
name	Name of the data set
producedBy	Name of the sensor that recorded the data
dataUnitsX	Units for the x-values
dataUnitsY	Units for the y-values
dataFile	URL of the data file
timeStamp	Time when data was collected
dataType	Kind of data stored in file
locatedAt	Location on specimen where data was taken

Fig. 11. Metadata entry for Mote #1 X-axis data during Elastic Level test

Once the data and metadata are uploaded into the system, the data can be automatically visualized and browsed through the web interface (Fig. 12). Metadata for each sensor was input into the data storage and visualization system. Each mote was treated as having two different sensors (one accelerometer in the x and y axis). The sensor metadata information contained the name of the sensor, the names of the data sets it collected, the type of data collected, the location of the sensor, the sensor manufacturer, and model number. Limited information from the manufacturer's specification sheet was also entered such as resolution, bandwidth, noise, non linearity, sensitivity, and transverse sensitivity. Photographs of the sensor and threshold limit states for alert triggering were added as optional information.

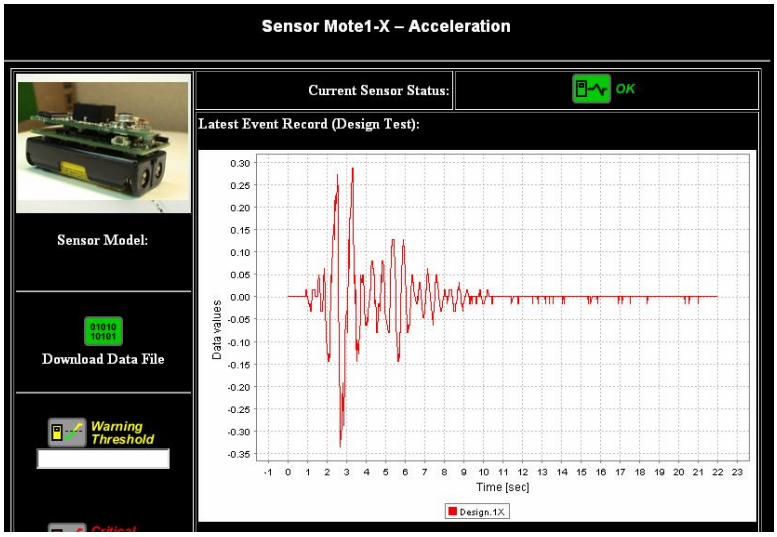

Fig. 12. Web interface acceleration data plot

The time stamps from each of the testing runs were input as event objects in the metadata model. In the laboratory, these events correspond to the shake table runs. In a practical structural monitoring situation, these events would correspond to actual earthquakes experienced by the structure. The data retrieval system correlates the time stamps from the events with the time stamps from the data files to group data from multiple sensors. This makes it easy to find and compare data from multiple sensors during the same event. The events were entered with magnitude values

corresponding to the relative intensity of the different testing sequences. In a practical situation, the actual magnitude of the earthquake could be entered allowing the user to browse for the most intense shaking experienced by the structure.

5. CONCLUSIONS

The results from the shake table test were used to determine the feasibility of using a comprehensive sensor, data storage, data retrieval, and visualization system for a structural monitoring application. The accelerometer data was successfully retrieved using a wireless link. The data was able to be cataloged using a metadata description and ingested into a metadata storage system based on the NEESgrid metadata services API. The data was automatically retrievable for presentation and visualization over a network through a web interface.

The test revealed that although the main components of the health monitoring application are functional, there are still shortcomings that restrict its practical use. The accelerometer and the analog-to-digital converter implemented in the current generation of MICA2 motes do not possess the fidelity required for structural state evaluation. Resolution problems due to quantization significantly affect the low level acceleration readings (Fig. 13). Communication packet loss due to interference in radio transmission is quite pronounced at high sampling rates and with multiple motes attempting to communicate simultaneously. This made the overall time required to download all accelerometer data slow even with only three sensors. Receiving the data from a densely instrumented structure would take much longer. Further improvements in self-organizing wireless networks and newer transmission strategies should help to address this issue.

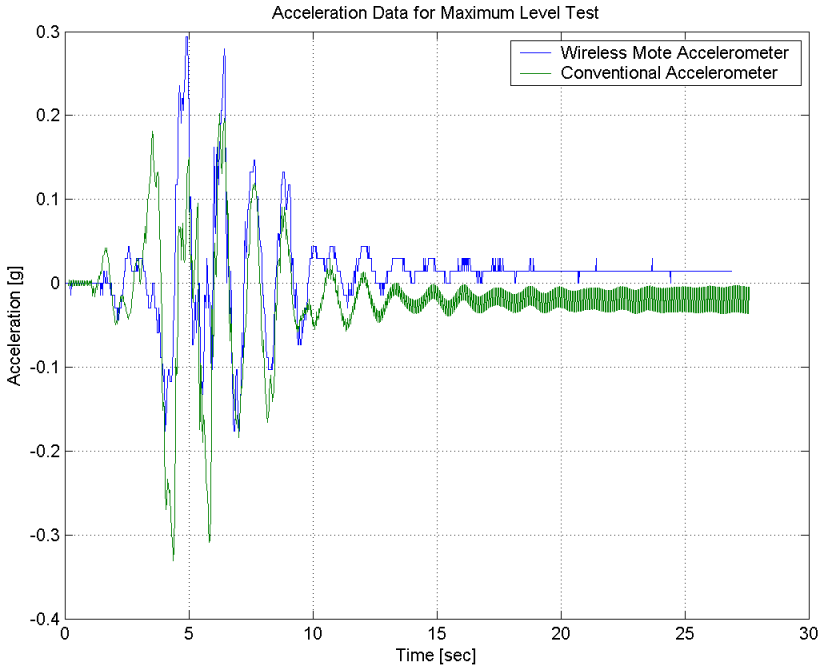

Fig. 13. Acceleration data from wireless and conventional accelerometers

ACKNOWLEDGEMENTS

The authors of this paper gratefully acknowledge the funding and support for this project provided by the Kajima Corporation through the CUREE-Kajima Phase V Joint Research Project. The authors thank Mr. Andres Espinoza and Professor Stephen A. Mahin, who conducted the reinforced concrete column shaking table test, for allowing us to utilize his specimens for the purposes of this project.

REFERENCES

1. J.L. Hill, D.E. Culler, "Mica: A Wireless Platform for Deeply Embedded Networks", *IEEE Micro*, 22, 12-24, IEEE, 2002.
2. S.D. Glaser, "Some real-world applications of wireless sensor nodes", *Smart Structures and Materials 2004*, 5391, 344-355, SPIE, San Diego CA, 2004.
3. J. Hill et al, "System architecture directions for networked sensors", *International Conference on Architectural Support for Programming Languages and Operating Systems*, 93-104, ACM, Cambridge MA, 2000.
4. D. Gay et al, "The nesC language: A holistic approach to network embedded systems", *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation*, 1-11, ACM, 2003.
5. H. Weinberg, "Dual axis, low g, fully integrated accelerometers", *Analog Dialogue,* 33, 23-24, Analog Devices, 1999.
6. G. Zhou, T. He, S. Krishnamurthy, J.A. Stankovic, "Impact of radio irregularity on wireless sensor networks", *Mobisys 2004 – Second International Conference on Mobile Systems, Applications and Services*, 125-138, ACM, Boston MA, 2004.
7. J. Futrelle, "NEESML Reference User Guide", *NEESGrid Technical Report 2003-15*, National Center for Supercomputing Applications, 2003.