Flexible smart sensor framework for autonomous structural health monitoring

Jennifer A. Rice[1]*, Kirill Mechitov[2], Sung-Han Sim[2], Tomonori Nagayama[3], Shinae Jang[2], Robin Kim[2], Billie F. Spencer, Jr.[2], Gul Agha[2] and Yozo Fujino[3]

[1]*Department of Civil and Environmental Engineering, Texas Tech University, Lubbock, TX, USA*
[2]*Department of Civil and Environmental Engineering, University of Illinois at Urbana-Champaign, 205 N. Mathews Avenue, Urbana, IL 61801, USA*
[3]*Department of Civil Engineering, University of Tokyo 7-3-1, Hongo, Bunkyo-ku, Tokyo 113-8656, Japan*

(Received November 13, 2009, Accepted February 18, 2010)

Abstract. Wireless smart sensors enable new approaches to improve structural health monitoring (SHM) practices through the use of distributed data processing. Such an approach is scalable to the large number of sensor nodes required for high-fidelity modal analysis and damage detection. While much of the technology associated with smart sensors has been available for nearly a decade, there have been limited numbers of full-scale implementations due to the lack of critical hardware and software elements. This research develops a flexible wireless smart sensor framework for full-scale, autonomous SHM that integrates the necessary software and hardware while addressing key implementation requirements. The Imote2 smart sensor platform is employed, providing the computation and communication resources that support demanding sensor network applications such as SHM of civil infrastructure. A multi-metric Imote2 sensor board with onboard signal processing specifically designed for SHM applications has been designed and validated. The framework software is based on a service-oriented architecture that is modular, reusable and extensible, thus allowing engineers to more readily realize the potential of smart sensor technology. Flexible network management software combines a sleep/wake cycle for enhanced power efficiency with threshold detection for triggering network wide operations such as synchronized sensing or decentralized modal analysis. The framework developed in this research has been validated on a full-scale a cable-stayed bridge in South Korea.

Keywords: smart sensor network; structural health monitoring; full-scale bridge monitoring; service-oriented architecture; Imote2.

1. Introduction

Civil infrastructure is the foundation of our society and has a widespread impact on the quality of our daily lives. Monitoring the safety and functionality of the world's buildings, bridges and lifeline systems, is critical to improving maintenance practices, minimizing the cost associated with repair and ultimately improving public safety. Structural health monitoring (SHM) provides the means for capturing structural response and assessing structural condition for a variety of purposes. For example, the information from an SHM system can be used to fine-tune idealized structural models, thereby allowing more accurate prediction of the response due to extreme loading conditions, such

*Corresponding Author, Assistant Professor, E-mail: Jennifer.Rice@ttu.edu

as an earthquake. SHM also can be used to characterize loads in situ, which can allow the detection of unusual loading conditions as well as validate the structure's design. In addition, real-time monitoring systems can measure the response of a structure before, during and after a natural or man-made disaster, that can be used in damage detection algorithms to assess the post-event condition of a structure.

Gaining a clear understanding of structural behavior to allow a reasonable assessment of its as-built condition requires high-fidelity sensor data to build accurate models (Nagayama *et al.* 2007). In addition, potentially problematic structural changes, such as corrosion, cracking, buckling, fracture, etc., all occur locally within a structure. It is expected that sensors must be in close proximity to the damage to capture the resulting change in response while sensors further from the damage are unlikely to observe measurable changes. To achieve an effective monitoring system that is capable of generating informative structural models and detecting critical structural changes, a dense array of sensors is required. Due to the cost of deployment and the potential for data inundation, such a dense instrumentation system is not practically realized with traditional structural monitoring technology.

Advances in wireless technology and embedded processing have made much lower-cost wireless smart sensor networks (WSSNs) an attractive alternative to wired, centralized data acquisition (DAQ) systems. The majority of the work using wireless smart sensors for structural monitoring has focused on using the sensors to emulate traditional wired sensor systems (Arms *et al*. 2004, Pakzad *et al*. 2008, Whelan and Janoyan 2009). As these systems require that all data be sent back to a central DAQ system for further processing, the amount of wireless communication required in the network can become costly in terms of excessive communication times and the associated power it consumes as the network size increases. For example, a wireless sensor network implemented on the Golden Gate Bridge that generated 20 MB of data (1600 seconds of data, sampling at 50 Hz on 64 sensor nodes) took over nine hours to complete the communication of the data back to a central location (Pakzad *et al*. 2008).

WSSNs leverage onboard computational capacity on the wireless sensors to allow data processing to occur within the network, as opposed to at a central location. By implementing data processing techniques, such as modal analysis or damage detection algorithms, in such a distributed manner, the amount of communication that occurs within the network can be reduced, while providing usable information on the structural condition (Nagayama and Spencer 2007). WSSNs employing decentralized computing offer a scalable solution that has the potential to dramatically improve SHM efforts.

In recent years, researchers have made progress toward addressing the inherent roadblocks to realizing SHM application that utilize WSSNs. Nagayama and Spencer (2007) successfully implemented a distributed WSSN SHM system on a laboratory scale truss structure. To date, the hardware and software required for SHM have yet to be integrated to achieve a framework that is suitable for autonomous monitoring and distributed data management on a full-scale structure employing a dense network of sensors.

The objective of this research is to provide an enabling WSSN framework to address issues that have limited their effectiveness for SHM systems. In particular, the development of flexible, multi-metric sensors for use in WSSNs with user-selectable anti-aliasing filters to produce high-quality data appropriate for SHM applications and the development and validation of an enabling, open-source software framework that is modular and adaptable are presented. The integration of these software and hardware components has resulted in a flexible framework that enables autonomous, full-scale implementations of SHM systems.

2. High-fidelity WSSN data acquisition

The successful utilization of WSSNs for structural monitoring relies on their ability to capture data that provides a reasonable representation of the physical response. Some of the inherent characteristics associated with wireless smart sensors have made high-fidelity data acquisition a challenging undertaking. This section presents the development of sensor hardware and data acquisition software designed specifically for a broad range of SHM applications. This sensor hardware interfaces with the Imote2 smart sensor platform, the only commercially available smart sensor platform that is well-suited to the demands of such applications. Until now, the vibration sensors that have been widely available for smart sensor platforms, and in particular the Imote2, have lacked user-selectable anti-aliasing filters, flexibility in the choice of sensing parameters, sample rate accuracy, and temperature correction.

2.1 Smart sensor platform

The smart sensor platform used in this research is the Imote2 (Fig. 1). The Imote2 is built around Intel's low-power X-scale processor (PXA27x). The scalability of the processor speed based on application needs allows for increased performance without a significant increase in overall power consumption. The onboard memory of the Imote2 is another feature that sets it apart from other wireless sensor platforms and allows its use for the high-frequency sampling and computationally intense data processing required for dynamic structural monitoring. It has 256 KB of integrated SRAM, 32 MB of external SDRAM and 32 MB of flash memory (Crossbow Technology 2007a).

TinyOS (www.tinyos.net) is the open-source operating system used on many smart sensors (Levis *et al*. 2005), including the Imote2. TinyOS only supports two types of executions: tasks and hardware event handlers. This concurrency model makes programming in TinyOS complicated, as it requires the use of many small event handlers and does not support real-time scheduling, making control of execution timing difficult. Hardware event handlers can preempt the execution of a task. Nagayama *et al*. (2006) discuss the potential effects of this limitation as it pertains to achieving synchronized sensing in a WSSN and some methods for overcoming it. In particular, an uncertain delay in the start of sensing due to the lack of strict timing control is an issue that must be addressed if synchronized sensing is to be achieved.

The Imote2 provides a flexible platform for a range of sensing applications. The sensors used

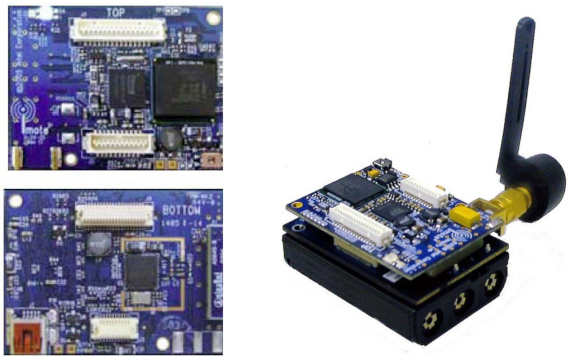

Fig. 1 Imote2 top and bottom view (left) and stacked on battery board with antenna (right)

with the Imote2 are interfaced to the main board via two connectors in a stackable configuration. The Imote2 does not have an onboard analog-to-digital converter (ADC) and therefore is only compatible with digital inputs, specifically I2C (Inter-Integrated Circuit: a two-line serial data bus that supports multiple data channels) and SPI (Serial Peripheral Interface: a four-wire digital bus that typically only supports a single data channel). The Imote2's flexible sensor interface allows its users to tailor sensor boards to their application.

2.2 SHM-A sensor board

Vibration-based SHM requires sensed data that well represents the physical response of the structure both in amplitude and phase. The measurements must have ample resolution to characterize the structural response and must be recorded with a consistent sample rate that is synchronized with other sensed data from the structure. Whether the data is used to perform modal analysis, system identification or vibration-based damage detection, these aspects of the data quality must be met so that reasonable results may be achieved (Nagayama *et al*. 2007). To be used in SHM applications, the sensor hardware that interfaces with the Imote2 must provide such high-fidelity data.

The only commercially available accelerometer sensor board that interfaces with the Imote2 (ITS400C sensor board, Crossbow Technology 2007b) was evaluated to determine its appropriateness for SHM applications by Nagayama *et al*. (2006). The results demonstrated the need for a sensor board with higher resolution and more accurate sampling rates designed specifically for SHM applications. To meet this need, the design of a newly developed Structural Health Monitoring Accelerometer (SHM-A) board is presented with its experimental verification. The SHM-A board provides user-selectable sampling rates and anti-aliasing filters for a broad range of applications. The components of the SHM-A board have been selected to meet the requirements of vibration-based SHM applications, specifically with respect to data quality and the demands of achieving synchronized sensing.

The key component of the SHM-A board is the Quickfilter QF4A512 ADC and signal conditioner with programmable sampling rates and digital filters (Quickfilter Technologies 2007). The SHM-A board interfaces with the Imote2 via SPI I/O and has a three-axis analog accelerometer for vibration measurement. A block diagram of the components of the SHM-A sensor board is given in Fig. 2. Fig. 3 shows three views of the board. The details of the board components will be discussed in subsequent paragraphs.

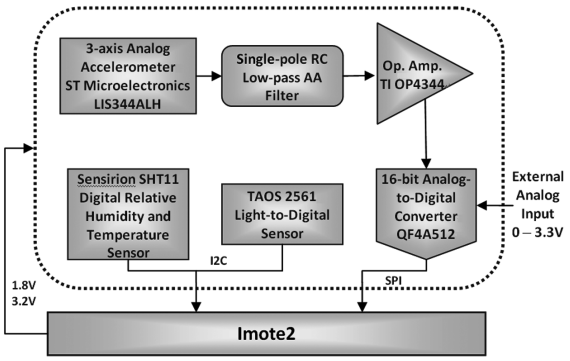

Fig. 2 Block diagram of SHM-A Rev 4.0

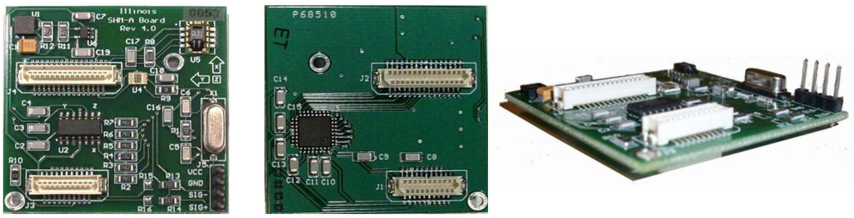

Fig. 3 SHM-A sensor board: top (left), bottom (middle) and perspective (right)

Table 1 LIS344ALH accelerometer specifications (STMicroelectronics 2008)

Parameter	Value
Axes	3
Measurement range	±2 g
Resolution	0.66 V/g
Power supply	2.4 V to 3.6 V
Noise density, x- and y-axes	22 - 28 μg/Hz
Noise density, z-axis	30 - 60 μg/Hz
Temperature range	-40 to 85°C
Supply current	0.85 mA

The ST Microelectronics LIS344ALH capacitive-type MEMS accelerometer (STMicroelectronics 2008), with DC to 1500 Hz measurement range, was chosen for the SHM-A board. This type of accelerometer utilizes the motion of a proof mass to change the distance between internal capacitive plates, resulting in a change of output voltage in response to acceleration. Kurata *et al*. (2006) investigated several commercially available MEMS accelerometers in the context of structural monitoring applications and found that the ST Microelectronics accelerometer had better performance than other comparably priced sensors. Though MEMS accelerometers are available with lower noise levels, the ST Micro accelerometer offers an excellent price/performance ratio. In addition, it provides three axes of acceleration on a single chip. The specifications for the accelerometer are given in Table 1. If lower noise characteristics are required for a specific application, a higher-cost accelerometer, such as the SD1221 (Silicon Designs 2007) or the Si-Flex SF1500S (Colibrys 2007), could be incorporated into the board design with appropriate measures to accommodate higher power requirements.

According to the datasheet noise density values for the accelerometer, the expected corresponding *RMS* noise over a 20-Hz bandwidth is 0.10 - 0.13 mg for the x and y axes and 0.13 - 0.26 mg for the z-axis. One hundred SHM-A boards were tested to determine their noise performance and their calibration constants (scale and offset). The sensors were placed flat on a desk and data was collected at 50 Hz (with a cutoff frequency of 20 Hz) for 1.5 minutes with no external excitation. Additional tests were conducted with 8 sensor boards at a 1000-Hz sample rate with a cutoff frequency of 250 Hz to assess the higher frequency performance. To ensure that the desk was not vibrating, simultaneous measurements were taken with a low-noise seismic accelerometer (PCB 393C).

The average measured *RMS* vibration level was 0.29 mg for the x- and y-axes and 0.67 mg for the z-axis. The higher noise levels in the z-axis appear to be intrinsic to the most recent ST Micro accelerometer revision (LIS344ALH) which exhibits higher $1/f$ noise characteristics (higher noise at

Table 2 QF4A512 Specifications (Quickfilter Technologies 2007)

Parameter	Condition	Value
Channels	Single or double ended	4
Signal-to-Noise Ratio (SNR)	f_s = 2 kHz	82 dB
	f_s = 2 MHz	69 dB
Throughput (w/40 MHz SPI bus rate)	1 channel active	1.47 Msps
	4 channels active	390 ksps
Nominal Resolution		16 bits
Effective Number of Bits (ENOB)	f_s = 2 kHz	13.2 bits
	f_s = 2 MHz	11.2 bits
Power Consumption	Quiescent	2.0 mW
	Power down	0.96 mW
	f_s = 1 kHz, 1 channel	84.8 mW
	f_s = 1 kHz, 4 channels	230.4 mW

lower frequencies) along that axis; this noise was not observed in the previous versions of the accelerometer. The *x*- and *y*-axes should be used as the primary measurement axes when possible.

The Quickfilter QF4A512 programmable signal conditioner (Quickfilter Technologies 2007) employs a versatile four-channel, 16-bit resolution ADC. Each channel has a selectable gain (up to 8x), an analog anti-aliasing filter, individually selectable sampling frequencies and individually programmable digital FIR filters (up to 512 filter coefficients). The primary reasons for the selection of the QF4A512 are its ability to achieve un-aliased signals regardless of the measurement bandwidth (outlined in detail in the following paragraph) and the option to implement high-coefficient FIR filters on each sensor channel. Although the power consumption of the QF4A512 is higher than other products, this drawback offset by the added features and power management approaches discussed later in this paper. The key features of the QF4A512 are summarized in Table 2.

The QF4A512 performs oversampling, filtering and decimation to achieve two purposes in the digitization of the measured signal. The first purpose of oversampling is to improve the resolution of the output by decreasing the noise from quantization error. The resolution of the ADC dictates the smallest measurable increment that can be resolved. Quantization introduces a constant level of noise energy which is uniformly distributed over the measured bandwidth. The higher the sampling frequency, the wider the frequency range over which the noise energy is distributed. Because the energy of the noise is constant, increasing the Nyquist frequency lowers the amplitude of the noise. When a digital decimation filter is applied to the oversampled signal, the noise energy above the new Nyquist frequency is eliminated, thereby improving the resolution of the signal. A 4-times oversampling rate lowers the quantization noise floor by 6 dB or the equivalent of achieving one additional bit in resolution.

The QF4A512 provides variable anti-aliasing filters by following the unaliased, oversampled signal with digital filtering and decimation. The analog anti-aliasing filters are 3rd order Bessel filters with a cutoff frequency of 500 kHz. While this cutoff frequency may seem high for structural monitoring applications, with bandwidths of interest typically below 20 Hz, it is set to ensure that aliasing is avoided when the signal is digitized at the much higher oversampling rates that are used (e.g., 12.5 MHz) without limiting the final measurement bandwidth. The digital decimation filters are Cascaded-Integrator-Comb (CIC) filters, working in combination with the Cascaded-Integrator Halfband (CIH) filters to ensure that the integrity of the signal is maintained upon decimation to the

final user-specified sampling frequency. This combination of filters provides excellent amplitude response while preserving a linear phase response (Hogenauer 1981). This method of oversampling, filtering and decimation to remove aliasing is common for PC-based analyzer systems. After the signal is downsampled to the effective sampling rate, the user-defined FIR filters (low-pass, high-pass or band-pass) are applied to further reduce noise and preserve the bandwidth of interest for the particular application.

A software driver for the SHM-A board was developed in TinyOS. The purpose of the driver is to control the functions of the QF4A512, such as loading the filter coefficients, allocating memory, time-stamping, writing data, etc. At the beginning of sensing, the driver first initializes the ADC and then triggers the sampling to start. During sampling, the samples are released from the QF4A512 and written to the Imote2 buffers as two-byte signed integers (16-bit).

Tests were conducted to calibrate each channel of the accelerometer. The SHM-A board mounted on an Imote2 was placed on an accelerometer calibration frame which ensured a level measurement surface. Measurements were taken with the board oriented so that signals corresponding to -1 g, 0 g and +1 g were measured for each of the measurement axes. The results provided the necessary calibration constants (DC offset and scale) which can be directly implemented in the sensing application.

In addition to the static calibration, dynamic calibration testing was conducted on a bench-scale shake table alongside a wired reference sensor to evaluate the sensor boards at frequencies above the DC (static) response. Focus was placed on the lower frequency range of 0 to 20 Hz range as well as higher frequencies up to 250 Hz. The results of the dynamic tests were in agreement with the static calibration tests. There is some challenge in verifying the performance of the SHM-A sensor board below 1 Hz due to limitation in the motion of the shake table and the presence of typical $1/f$ noise in the accelerometer output. The result is some discrepancy between the reference sensor and the SHM-A board. More detail on the calibration results may be found in Rice and Spencer (2009) and the datasheet for the SHM-A sensor board (http://shm.cs.uiuc.edu/files/SHM-A_Datasheet.pdf).

The environmental sensors on the SHM-A board include a digital light sensor and a digital combination temperature/humidity sensor. Initial sensor board test results exhibited large drifts in the mean value of the acceleration data over time, even when the sensor was stationary. Subsequent investigation revealed that the drift was due to the temperature sensitivity of the mean value of the accelerometer in response to self-heating of the sensor board caused by the operation of the Imote2 processor and the QF4A512 ADC. To address the mean value drift of the accelerometer output resulting from temperature changes, onboard temperature compensation has been implemented in software. By simultaneously measuring the temperature and the acceleration, the direct relationship between the self heating of the board and the accelerometer output can be determined. This correction factor is built into the driver and automatically accounted for during sensing. The temperature sensors were not individually calibrated; performing calibration of the temperature sensors may yield more accurate temperature correction results.

2.3 Unified sensing

In addition to the previously described hardware components, specific software components have been implemented to ensure accurate data sampling while maintaining memory efficient operation. The *Unified Sensing* service provides a convenient, general-purpose application programming interface,

replacing the standard TinyOS sensing interface for the Imote2 and extending its functionality to include precise time-stamping of the data and providing transparent support for a variety of sensor boards, including the SHM-A board. The complete and self-contained data representation employed in *Unified Sensing* makes it easy to pass around and modify the data without hard-coding connections between components that use only parts of this data. This approach facilitates data being passed directly to the application services described in the following section. More detail on the *Unified Sensing* service can be found in Rice *et al.* (2010).

3. WSSN software development framework

SHM applications implemented on WSSNs require complex programming, ranging from network functionality to algorithm implementation. Software development is made even more difficult by the fact that many smart sensor platforms employ special-purpose operating systems, such as TinyOS, without support for common programming environments. Although TinyOS is widely used for WSSN applications, it is a very challenging environment for non-programmers to develop network control and application software. The embedded system expertise required to develop SHM applications has limited the use of WSSN technology for monitoring of civil infrastructure.

3.1 Service-oriented architecture

A common approach to the issue of software complexity is to divide the software system into smaller, more manageable components. Service-oriented architecture (SOA) has been proposed as a way to use this design philosophy in building dynamic, heterogeneous distributed applications (Singh and Huhns 2005, Tsai 2005). Services, in SOA terminology, are self-describing software components in an open or modifiable distributed system. The description of a service, called a contract, lists its inputs and outputs, explains the provided functionality, and describes non-functional aspects of execution (e.g., resource consumption). An application built using SOA consists of a number of linked services within a middleware runtime system that provides communication and coordination between them. Data is passed among the services in a common format and the services do not require knowledge of its origin. Different applications can be built from the same set of services, depending on how they are linked and on the execution context (Gu *et al*. 2005). This approach provides support for dynamic, highly adaptive applications without the need to revisit and adapt the implementation of each service for each new application. SOA has the potential to address the challenges inherent to designing complex and dynamic WSSN applications (Liu and Zhao 2005, Mechitov *et al*. 2007).

An attractive aspect of SOA is that it separates the tasks undertaken in WSSN application development. In sensor networks used for structural monitoring, the application designer is likely to also be the application user, having expertise in SHM applications and the desired output of the network, but limited knowledge on network programming and the hardware-software interface. As such, it is important for the less complex, high-level design of the application and the domain-specific algorithms used by the services to be separated from the often more complex, low-level infrastructure necessary to make the WSSN work. SOA in WSSNs makes it possible to compose and deploy complex applications through a user interface suitable for non-programmers, potentially accelerating the use of WSSNs of SHM applications.

The Illinois Structural Health Monitoring Project (ISHMP), a collaborative effort between researchers in civil engineering and computer science at the University of Illinois at Urbana-Champaign, has sought to tackle the complexity associated with creating WSSN applications by developing a software framework based on the design principles of SOA. This framework provides a suite of services implementing key middleware infrastructure necessary to provide high-quality sensor data and to reliably communicate it within the sensor network, as well as number of commonly used numerical algorithms. This framework is intended to allow researchers and engineers to focus their attention on the advancement of SHM approaches without having to concern themselves with low-level networking, communication and numerical sub-routines. This software is open-source and available for public use at http://shm.cs.uiuc.edu/software.html.

The service-based software framework provides an open-source software library of customizable services for, and examples of, SHM applications utilizing WSSNs. SHM middleware services and distributed damage detection algorithms reported in Nagayama and Spencer (2007), along with a collection of tools, utilities and algorithms, have been implemented to enable efficient development of flexible and robust SHM applications on WSSNs. Additional services that enable autonomous network operation have also been included in the software toolsuite, as described in Section 4.

3.2 ISHMP toolsuite

The components of the service-based framework provided by the ISHMP Toolsuite can be divided into three primary categories: (1) foundation services, (2) application services and (3) tools and utilities. In addition, a library of supporting numerical functions that are common to many SHM algorithms is provided including fast Fourier transform (FFT), singular value decomposition (SVD), eigenvalue analysis, etc. In the description of the services that follows, *leaf nodes* are defined as the nodes that comprise the sensor network while the *gateway node* is the Imote2 that is connected to the base station PC that operates the network. The network topology that is utilized varies from application to application. More information on these applications and the topologies they employ can be found in Sim and Spencer (2009) and Rice *et al.* (2010).

The foundation services implement the functionality required to support the application and other services. When used together, one of the primary purposes of the foundation services is to enable applications acquire synchronized data from a network of sensors. The foundation services include network time synchronization (*TimeSync*), the previously described *Unified Sensing*, reliable communication of both short messages and long data records (*ReliableComm*), and a service that supports the reliable dissemination of network commands (*RemoteCommand*).

The application services provide the numerical algorithms necessary to implement SHM applications on the Imote2s and may also be used independently. For each application service, an application module to test the algorithm on both the PC and the Imote2 has been developed. The applications services include synchronized sensing (*SyncSensing*), correlation function estimation (*CFE*), the Eigensystem Realization Algorithm (*ERA*), Stochastic Subspace Identification (*SSI*), Frequency Domain Decomposition (*FDD*), and the Stochastic Damage Locating Vector method (*SDLV*).

Documentation is provided for each service and test application within their respective directories in the ISHMP software package. This documentation gives details on requirements and formats of the inputs and outputs for each service.

The tools and utilities are used for network testing and debugging and are necessary in any large-scale or long-term WSSN deployment to evaluate the network conditions at the structure, determine

appropriate values of adjustable system parameters, and assess power consumption and longevity issues. Included are utilities for resetting nodes remotely, measuring battery voltage, and changing the radio channel and power for gateway and leaf nodes.

The application tools can be categorized as either those operating on a single node (typically the gateway node) or those that operate on multiple, distributed leaf nodes. The single-node application tools include a gateway node sensing tool (*LocalSensing*), a terminal program for interfacing the base station PC with the gateway Imote2 (*imote2comm*), and a numerical service that simulates the identification of potential structural damage locations from injected acceleration data (*TestServices*).

The distributed nature of the multi-node application tools requires careful scheduling and coordination of network tasks, making them more susceptible to failure if any of the nodes in the network malfunctions. Efforts have been made to ensure that the applications continue to operate even when one or more of the nodes in the network exhibit unexpected behavior. The distributed applications tools include a tool to test the raw bi-directional communication between a sender and a group of receiving nodes (*TestRadio*), a flexible network-wide synchronized sensing application (*RemoteSensing*), and a sample application that demonstrates a decentralized approach to SHM data collection and aggregation (*DecentralizedDataAggregation*, Sim and Spencer 2009).

All of the services introduced here are described in more detail in Rice *et al.* (2010). Sim and Spencer (2009) provide instructions and examples for creating applications using the ISHMP Toolsuite.

4. Autonomous monitoring

Three critical deployment issues drive the WSSN software that is presented in this section: 1) continuous and autonomous monitoring, 2) efficient power management and 3) data inundation mitigation. While these may appear to be conflicting goals, careful application design can meet the requirements for all three. The solution is to implement a network that is only minimally active during non-critical structural response, but becomes fully active to measure higher response levels. The software presented in the previous section lays the groundwork for full-scale, autonomous monitoring of civil infrastructure however, it does not address these concerns that arise when moving from a laboratory setting to a full-scale deployment.

Ideally, full-scale WSSN deployments should require minimal external interaction after some initialization involving the establishment of network operation parameters, unless instructed otherwise by the network administrator. Special care must be taken in the design of the application software to ensure a continuous and autonomous operating scenario is achieved while maintaining power efficiency. These measures can be divided into three categories: schedule-based operations, trigger-based operations and safeguard features. This section presents software developments in each of these categories that, when integrated, enable full-scale, autonomous network operation.

4.1 Sleep cycling

In a traditional wired sensor implementation, power management is of little concern. The sensors can remain active at all times and thus have the ability to be interrogated at any time to acquire data. Unlike wired systems, one of the most critical features of a successful WSSN deployment is the implementation of careful power management strategies. A common approach to achieving significant energy savings in sensor network applications is to allow the sensor nodes to sleep

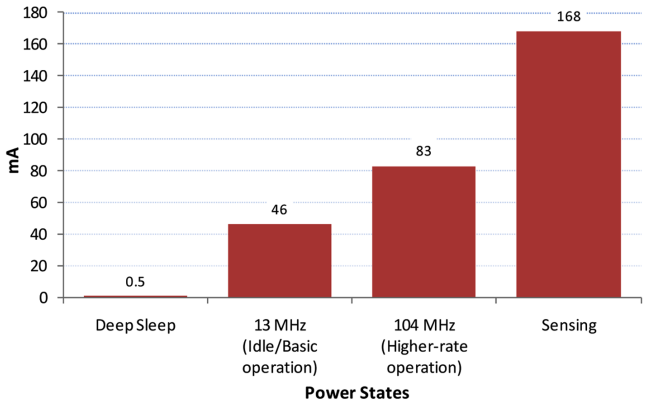

Fig. 4 Current draw in various operational modes of the Imote2 with the SHM-A sensor board

during periods of inactivity while waking periodically to listen for instructions (Ye *et al*. 2002, Wang and Xiao 2006). The Imote2 allows the processor to be put into a deep sleep mode, whereby only the clock component of the processor is supplied power; all other components are powered down. Fig.4 illustrates the power savings associated with the deep sleep mode relative to other operations of the sensor node. When the node is in the deep sleep state, it cannot send data or receive commands via the radio or the serial ports, and the LEDs do not function. Effectively, the node has no power until the sleep time expires.

While it may seem advantageous to keep the leaf nodes in the deep sleep mode for extended periods of time to save power, this approach limits the ability of the gateway node to access the network at random to send inquiries or initiate network operations. To take advantage of the power savings of the deep sleep mode, while still allowing the gateway node access to the leaf nodes, a sleep/wake cycle service called *SnoozeAlarm* has been implemented. When *SnoozeAlarm* operates on the leaf nodes (i.e., they are in the *SnoozeAlarm* mode), they sleep for a set period of time and then wake up for a relatively short period of time, during which they can listen and receive message. The ratio between the time spent awake and the sum of sleep time and the awake time is the *SnoozeAlarm* duty cycle. For optimal power saving, the duty cycle should be minimized while still allowing a long enough listen time to receive and process commands (>500 ms). The actual overall power savings associated with the use of *SnoozeAlar*m is dependent on the application in which it is implemented.

The *SnoozeAlarm* wakeup command provides an efficient method for waking a network of leaf nodes in *SnoozeAlarm* mode. The *ReliableComm* protocol for broadcasting messages to a group of leaf nodes is only successful if all of the destination nodes respond with an acknowledgement in a set period of time, thus limiting its use for waking the network. Instead, the network is woken in a serial manner using successive unicast commands from the gateway node to individual leaf nodes in the network. The gateway node cycles through the list of sleeping leaf nodes, sending a wakeup command to one node in the list each time a preset time-out timer fires. When a leaf node sends back an acknowledgement, thus indicating it received the message and is remaining awake, it is removed from the list of nodes to wake up and added to the list of nodes that have been successfully woken. This process continues until all nodes are awake or until a time-out timer expires.

4.2 Threshold triggering

One approach to ensure that important occurrences are captured by the sensor network is to designate a subset of the network to sense data on a more frequent basis than the rest of the network and provide alerts (Hui *et al*. 2003, Wang and Xiao 2006). In this research, the *Threshold Sentry* tool allows a subset of the leaf nodes to act as *sentry nodes*, in addition to their duties as leaf nodes. *ThresholdSentry* is setup on the gateway node by specifying the IDs of the leaf nodes that comprise the sentry network, the interval at which they will be asked to sense data, the duration of the data check on each sentry node, the sampling parameters for the data check, and the threshold value used for comparison in the data check. If a sentry node determines that the threshold value has been exceeded during its observation time, it sends an alert message to the gateway node. Upon receipt of the flag, the gateway node makes the decision on the next actions to implement in the network, such as waking the remaining nodes and initiating *RemoteSensing*. The current implementation of *ThresholdSentry* utilizes acceleration measurements; however other triggers, such as strain levels or wind speed, may be incorporated into the application.

The selection of sentry nodes and their threshold values should be made such that the threshold is exceeded often enough for adequate structural monitoring, but not an excessive number of times at the risk of data inundation and higher power consumption levels. Because a single threshold value is used for all sentry nodes, the nodes selected as sentry nodes should measure similar levels of vibration to ensure consistency in the events that trigger the network. For example, on a long-span bridge, the nodes located near the support piers are expected to experience much lower vibration levels than those near the mid-span of the bridge. Sentry nodes both of these locations and would exceed the threshold under very different loading circumstances.

The temporal resolution of the sentry node wakeup events is a user-defined parameter that is based on the structure being monitored, the amount of data that is required from the network and power constraints of the sentry nodes. The sentry nodes are identical to the leaf nodes in hardware and functionality. Since they wake up more often than leaf nodes to carry out sentry sensing, they will use more power. The more sentry nodes that are utilized and the less often they wake up, the more the burden of sentry sensing can be shared, thus reducing the levels of additional power usage.

The current implementation of *ThresholdSentry* used in conjunction with *RemoteSensing* allows the network to capture the occurrence of longer-duration, lower-frequency events such as high wind; however, it does not support capturing short-term, transient events such as an earthquake. The time required to wake the network and perform time synchronization prior to the collection of data would cause such events to be missed. In future network development, the network wakeup time could be reduced using a propagating wakeup message with optimized communication parameters and the order of data collection and synchronization could be switched to facilitate faster initiation of sensing.

4.3 Autonomous network operation

Achieving an autonomous SHM implementation on a network of smart sensors requires a high-level application to coordinate each of its components in response to various events. *AutoMonitor* has been developed to provide this functionality. *AutoMonitor* is present on the gateway node and serves the purpose of maintaining the *RemoteSensing* and *ThresholdSentry* parameters and coordinating the associated network tasks.

AutoMonitor is initiated via an input file that sets the parameters for each of the tasks it coordinates. Once started, it requires no additional input from the user. The selection of most of these parameters is highly application-dependent and will take a period of adjustment and refinement to optimize for each case. Many of these parameters have power consumption implications; their effect on power management must be carefully considered. Jang *et al.* (2010) describes the selection of these parameters for the long span bridge deployment described in the following section.

After the input file containing all of the necessary parameters is read, *AutoMonitor* initiates *ThresholdSentry*. *AutoMonitor* employs a timer and a counter to limit the number of *RemoteSensing* events that occur in a particular time period. When the gateway node receives the alert message that the threshold has been exceeded on a sentry node, it first checks whether the maximum number of *RemoteSensing* events in the set time period has been reached. If maximum has been reached, *ThresholdSentry* is resumed. If not, *AutoMonitor* sends a command to wake the network and initiate *RemoteSensing.* After *RemoteSensing* completes and all of the data is transferred *ThresholdSentry* is reinitiated.

The software components presented in this section enable continuous operation of WSSNs for SHM applications outside of the laboratory setting. This software allows critical structural response to be captured while maintaining low-power network operation the majority of the time. The *AutoMonitor* network management application coordinates the operation of *ThresholdSentry*, *SnoozeAlarm* and *RemoteSensing* to ensure autonomous and continuous functionality of the network. More details on the software implementation may be found in Rice and Spencer (2009).

5. Framework integration

The software and hardware developed in this research were validated on a cable-stayed bridge (the 2nd Jindo Bridge) in South Korea. This deployment is part of a trilateral collaboration between South Korea (KAIST), Japan (University of Tokyo) and the USA (University of Illinois at Urbana-Champaign). The purpose of the deployment is to demonstrate the suitability of the Imote2 smart sensor platform, the SHM-A sensor board, and the ISHMP software for the full-scale SHM of a bridge. The Jindo Bridges (Fig. 5) are twin cable-stayed bridges that connect Jindo Island to the far southwestern tip of the Korean Peninsula near the town of Haenam (Fig. 6). The older span finished

Fig. 5 Twin Jindo Bridges connecting Jindo Island with the Korean Peninsula with the 2nd Jindo Bridge on the left

Fig. 6 Location of the Jindo Bridges on the Korean Peninsula (left), between Jindo and Haenam (right)

construction in 1984 and the newer span (the 2nd Jindo Bridge) was completed in 2005. The 2nd Jindo Bridge, the subject of this study, is on the left in Fig. 5.

The primary goal of the Jindo Bridge deployment is to realize the first large-scale, autonomous network of smart sensors utilized for SHM. This deployment is expected to highlight the challenges and opportunities associated with such a large scale test-bed and thus provide rich information for researchers and engineers interested in achieving a similar SHM system. For the research presented in this paper, the primary goals of the deployment are to validate the performance of the SHM-A sensor board for full-scale testing and validate the autonomous network operation software.

In total, 70 Imote2 leaf nodes with SHM-A sensor boards have been installed on the Jindo Bridge. Currently, the network has been in continuous operation for over four months with the primary deployment challenges associated with software parameter optimization. The SHM-A sensor board has successfully captured ambient traffic loading with peak acceleration ranging from less than 5 mg to over 30 mg. Further analysis of the data resulted in the successful identification of the first twelve modes of vibration on the deck, as well as tension forces of 10 cables with large tensile stresses. More detail on the Jindo Bridge deployment, including the implementation of renewable power sources, the selection and implication of various network parameters, and the analysis result of measured data can be found in Jang *et al.* (2010) and Cho *et al.* (2010).

6. Conclusions

The research presented in this paper has laid the foundation for autonomous implementations of WSSNs for full-scale SHM of large structures. The flexible framework encompasses the necessary hardware, software and implementation considerations to enable distributed SHM strategies in a wide range of applications. The result of this research is a road map for achieving autonomous, full-scale WSSN applications to improve infrastructure monitoring practices.

The autonomous network management software implemented on a network of Imote2 smart sensors employing the SHM-A sensor board has enabled the ongoing large-scale deployment on the 2nd Jindo Bridge in South Korea. The *AutoMonitor* application has successfully managed the network

by running *ThresholdSentry* to trigger network-wide synchronized sensing when it is necessary while limiting unnecessary or undesired data acquisition events. Thus far, the 2nd Jindo Bridge deployment has tested the limits of single-hop WSSN implementations with reasonable success and has already provided a wealth of information and insight into the critical issues that are still being addressed as part of the ongoing study. The results from this bridge represent the first autonomous, large-scale deployment of a WSSN for structural monitoring.

Acknowledgments

Financial support for this research was provided in part by the National Science Foundation under NSF grants CMS 03-01140 and CMS 06-00433 (Dr. S.C. Liu, Program Manager) and Intel Corporation Research. The first and second authors were supported by Vodafone-U.S. Foundation Graduate Fellowships. The authors gratefully acknowledge the valuable technical assistance of Dr. Mariella DeFino from Politecnico di Bari, Lama Nachman and Robbie Adler from Intel, Dr. Ralph Kling from Crossbow and Aaron Headley from Quickfilter.

References

Arms, S.W., Galbreath, J.H., Newhard, A.T. and Townsend, C.P. (2004), "Remotely reprogrammable sensors for structural health monitoring", *Proceedings of the Structural Materials Technology (SMT): NDE/NDT for Highways and Bridges*, Buffalo, NY, September.

Cho, S., Jo, H., Jang, S., Park, J., Jung, H.J., Yun, C.B., Spencer, Jr., B.F. and Seo, J. (2010), "Structural health monitoring of a cable-stayed bridge using smart sensor technology: data analysis", *Smart Struct. Syst.*, **6**(5-6), 461-480.

Colibrys Inc. (2007), *Si-Flex SF1500S Accelerometer*, Neuchatel, Switzerland.

Crossbow Technology, Inc. (2007a), *Imote2 Hardware Reference Manual*, Available at http://www.xbow.com/Support/Support_pdf_files/Imote2_Hardware_Reference_Manual.pdf.

Crossbow Technology, Inc. (2007b), *ITS400 - Imote2 Basic Sensor Board*, Available at http://www.xbow.com/Products/productdetails.aspx?sid=261.

Gu, T., Pung, H.K. and Zhang, D.Q. (2005), "A service-oriented middleware for building context-aware services", *J. Netw. Comput. Appl.*, **28**(1), 1-18.

Hogenauer, E.B. (1981), "An economical class of digital filters for decimation and interpolation", *IEEE Trans. Acoust., Speech, Signal Processing,* **29**(2), 155-162.

Hui, J., Ren, Z. and Krogh, B.H. (2003), "Sentry-based power management in wireless sensor networks", *Proceedings of the '03 Information Processing in Sensor Networks, Second International Workshop*, Palo Alto, CA, USA, April.

Jang, S., Jo, H., Cho, S., Mechitov, K., Rice, J.A., Sim, S.H., Jung, H.J., Yun, C.B., Spencer, Jr., B.F. and Agha, G. (2010), "Structural health monitoring of a cable-stayed bridge using smart sensor technology: deployment and evaluation", *Smart Struct. Syst.*, **6**(5-6), 439-459.

Kurata, N., Saruwatari, S. and Morikawa, H. (2006), "Ubiquitous Structural Monitoring using Wireless Sensor Networks", *Proceedings of the '06 International Symposium on Intelligent Signal Processing and Communication Systems*, Tokyo, December.

Levis, P., Madden, S., Polastre, J., Szewczyk, R., Whitehouse, K., Woo, A., Gay, D., Hill, J., Welsh, M., Brewer, E. and Culler, D. (2005), *TinyOS: An Operating System for Sensor Networks*, Ambient Intelligence (Ed. Weber, W., Rabaey, J.M., Aarts, E.), Springer Berlin Heidelberg.

Liu, J. and Zhao, F. (2005), "Towards semantic services for sensor-rich information systems", *Proceedings of the International Workshop on Broadband Advanced Sensor Networks*, Boston, MA, October.

Mechitov, K., Razavi, R. and Agha, G. (2007), "Architecture Design Principles to Support Adaptive Service Orchestration in WSN Applications", *ACM SIGBED Review*, **4**(3), 37-42.

Nagayama, T. and Spencer, Jr., B.F. (2007), *Structural health monitoring using smart sensors*, NSEL Report Series 001, University of Illinois at Urbana-Champaign, Available at https://www.ideals.illinois.edu/handle/2142/3521.

Nagayama, T., Rice, J.A. and Spencer, Jr., B.F. (2006), "Efficacy of Intel's Imote2 wireless sensor platform for structural health monitoring applications", *Proceedings of the Asia-Pacific Workshop on Structural Health Monitoring*, Yokohama, Japan.

Nagayama, T., Sim, S.H., Miyamori, Y. and Spencer, Jr., B.F. (2007), "Issues in structural health monitoring employing smart sensors", *Smart Struct. Syst*., **3**(3), 299-320.

Pakzad, S.N., Fenves, G.L., Kim, S. and Culler, D.E. (2008), "Design and Implementation of Scalable Wireless Sensor Network for Structural Monitoring", *J. Infrastruct. Syst*., **14**(1), 89-101.

Quickfilter Technologies, Inc. (2007), *QF4A512 4-Channel Programmable Signal Conditioner*, Allen, TX.

Rice, J.A. and Spencer, Jr., B.F. (2009), *Flexible Smart Sensor Framework for Autonomous Full-scale Structural Health Monitoring*, NSEL Report Series 018, University of Illinois at Urbana-Champaign, Available at http://www.ideals.illinois.edu/handle/2142/13635.

Rice, J.A., Mechitov, K.A., Sim, S.H., Spencer, Jr., B.F. and Agha, G. (2010), "Enabling Framework for Structural Health Monitoring Using Smart Sensors", *Struct. Control Health Monit.*, Published Online, Available at http://www3.interscience.wiley.com/journal/123320575/abstract.

Silicon Designs, Inc. (2007), *Model 1221 Low Noise Analog Accelerometer*, Issaquah, WA.

Sim, S.H. and Spencer, Jr., B.F. (2009), *Decentralized Strategies for Monitoring Structures using Wireless Smart Sensor Networks*, NSEL Report Series, 019, University of Illinois at Urbana-Champaign, Available at http://www.ideals.illinois.edu/handle/2142/14280.

Singh, M.P. and Huhns, M.N. (2005), *Service-Oriented Computing: Semantics, Processes, Agents*, John Wiley and Sons, New Jersey.

STMicroelectronics (2008), *LIS344ALH - ultracompact MEMS inertial sensor high performance 3-axis ±2/±6g ultracompact linear accelerometer*, Available at http://www.st.com/stonline/books/pdf/docs/14337.pdf.

Tsai, W.T. (2005), "Service-Oriented System Engineering: A New Paradigm", *Proceedings of the IEEE International Workshop on Service-Oriented System Engineering*, October.

Wang, L. and Xiao, Y. (2006), "A Survey of Energy-Efficient Scheduling Mechanisms in Sensor Networks", *Mobile Netw. Appl*., **11**(5), 723-740.

Whelan, M.J and Janoyan, K.D. (2009), "Design of a Robust, High-rate Wireless Sensor Network for Static and Dynamic Structural Monitoring", *J. Intel. Mat. Syst. Str*., **20**(7), 849-863.

Ye, W., Heidemann, J. and Estrin, D. (2002), "An energy-efficient MAC protocol for wireless sensor networks", *Proceedings of the 21st Conference of the IEEE Computer and Communications Societies (INFOCOM)*, New York, NY, USA.