Building Blocks for Adaptive Modular Sensing Systems

Anita Jain and Michael D. Naish
Sensing and Mechatronic Systems Laboratory
Dept. of Mechanical and Materials Engineering
The University of Western Ontario
London, Ontario, Canada
ajain27@uwo.ca, naish@eng.uwo.ca

***Abstract*—This paper describes the design and implementation of a set of building blocks for the construction of adaptive modular sensing systems. A hardware architecture is proposed which allows the development of a general set of modular components with embedded knowledge about their capabilities. These modular components are referred to as Transducer Interface Modules (TIMs). A knowledge representation scheme enables individual TIMs to exchange information about their capabilities and form a collective identity. The overall system functionality is determined by the manner in which TIMs are connected. This functionality is supported by a distributed software architecture that allows configuration-specific algorithms to be automatically loaded into each module. These building blocks can be rapidly reconfigured to form modular systems that are expected to prove useful in many applications, including industrial control, inspection systems, mobile robotics, monitoring, and data acquisition.**

I. Introduction

In recent years, sensor systems have become extremely popular for an array of applications. Several embedded solutions have enabled the implementation of a broad range of modular reconfigurable sensor systems. One of the best known systems is an extremely lightweight data gathering architecture based on Crossbow's MICA sensor nodes, known as Motes [1], which utilizes the Berkeley TinyOS operating system [2]. Motes largely place emphasis on networking, communication, and data propagation. Smart-its [3], sharing several features with Motes, are flexible self-contained units providing support for rapid assembly. eBlocks [4, 5] are best suited for developing basic sensor systems involving only logical transformations and/or state maintenance of sensor information. More complex sensing systems such as the Modular Architecture for Sensor Systems (MASS) [6], I-BLOCKS [7] aim to provide application flexibility due to their modular hardware and software design approaches.

Most of the systems discussed above share many features including sensing units, a core processing unit such as a microcontroller, a wireless transceiver unit, and a software architecture to manage the overall system. However, these systems also differ from each other due to the applications that they were designed for. Motes and Smart-its have limited processing capability, making it extremely difficult to perform complicated in-network fusion or analysis. The processing capabilities of eBlocks are limited to performing basic logic transformations (e.g., AND, OR, NOT) or basic state functions (e.g., prolong, toggle, trip). MASS and I-BLOCKS are more capable as they implement a distributed software control architecture. While most of the systems utilize a wireless approach to accomplish communication, eBlocks rely largely on wired communication.

In general, these systems do not support rapid reconfiguration, and hence, are not very efficient solutions for constantly changing application requirements. The objective of this research is to create a set of intelligent modular sensing and actuating components that can be rapidly reconfigured for various applications. These modular components are referred to as Transducer Interface Modules (TIMs). The TIMs each provide a single sensing or actuating function and have embedded knowledge about their processing and communication capabilities. The user has the freedom to connect these blocks in various configurations to create custom solutions. A distributed software architecture [8] executed on the blocks enables automatic acquisition of configuration-specific algorithms, enabling the modules to be rapidly reconfigured for various applications. Potential applications of composite sensing systems include industrial control, inspection systems, mobile robotics, monitoring, and data acquisition.

The remainder of this paper is organized as follows: In Section II, the major system requirements based on the research objectives are discussed. In Section III, a detailed description of the internal hardware architecture and geometry of Transducer Interface Modules — the core of the adaptive modular sensing systems — is presented. A knowledge representation scheme that enables the modules within a composite group to communicate their capabilities to one another is presented in Section IV. In Section V, a description of additional modules types (interconnect modules, power modules, and administration modules) is provided. A brief overview of the software architecture utilized on the modules is provided in Section VI, followed by conclusions and future work in Section VII.

II. System Requirements

In this section, a set of system requirements is established based on the research objectives discussed in Section I. One of the major system requirements is the design and development of a modular hardware platform which is independent of the software architecture controlling the system. The hardware platform is encapsulated in a module that can be interfaced to a multitude of sensors and actuators to form a variety of sensor- and actuator-based building blocks. The hardware platform should be capable of executing complex system algorithms. There is also a need for a knowledge representation scheme that allows modules to communicate their capabilities and form a collective identity. Another important requirement is to provide the user with the flexibility to reconfigure the modules in a number of different assemblies for various applications. The overall functionality of a group of modules is determined by the manner in which modules are connected. To support more complex applications, each module should be able to determine its position and orientation relative to other modules.

Additional requirements for the modules include: compactness, low weight, low power consumption, and low cost. Low power consumption is an important factor since modules can be battery powered. The power consumption of the system is heavily dependent on the internal hardware architecture and the drivers used to control it. By selecting hardware components having extremely low power requirements and designing an efficient set of drivers, power consumption can be reduced significantly. Ideally, a module should be able to be powered by a standard battery for at least 8 hours, independent of the power requirements of the device attached to the module. The overall size and weight of the module is a trade off between the ease with which users can handle the modules and the stability of the module itself. Each module must be able to accommodate the internal hardware as well as support external sensors and actuators.

III. Transducer Interface Modules

In this section, a detailed discussion of the Transducer Interface Modules (TIMs) is presented. The TIMs serve as the base modules in the formation of adaptive modular sensing systems. Each module is capable of performing a single sensing or actuation function. Fig. 1 shows the basic block diagram of a TIM.

A. Internal Hardware Architecture

The internal hardware architecture is an interface between the transducers and the communication medium. The architecture is designed to provide an interface to a multitude of sensors and actuators, execute application specific system algorithms, and manage communication with other modules in the system. The architecture is a stackable system consisting of various layers which include a controller/communication layer, a transmission gate layer, an interface layer, and a power layer. This design facilitates the modification or extension of the hardware on any layer, simplifies the addition of new layers, and results in compact modules.

The core of this stackable architecture is the controller/communication layer. The architecture utilizes a single microcontroller, referred to as the module controller, to accomplish data acquisition, overall module control, application

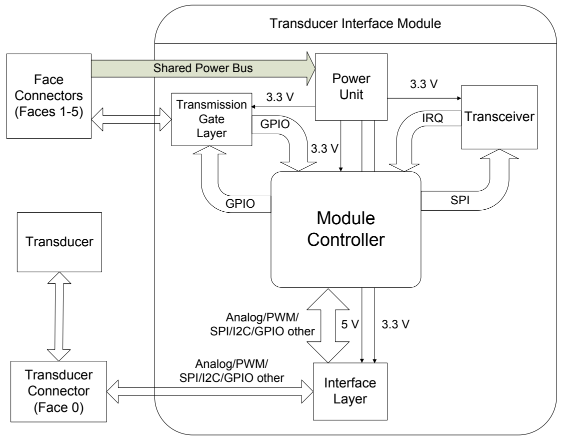
Transducer Interface Module
Face Connectors (Faces 1-5)
Shared Power Bus
Power Unit
3.3 V
3.3 V
Transmission Gate Layer
GPIO
3.3 V
IRQ
Transceiver
GPIO
SPI
Module Controller
Transducer
Analog/PWM/ SPI/I2C/GPIO other
5 V
3.3 V
Analog/PWM/ SPI/I2C/GPIO other
Transducer Connector (Face 0)
Interface Layer

Fig. 1. Block diagram of the Transducer Interface Module

specific processing, and communication with other modules. The NXP LPC2148 microcontroller [9], based on the 16-bit/32-bit ARM7TDMI-S central processing unit, is selected for the prototype. The ARM controllers are widely used in the design of embedded applications due to their small size, low power consumption, and high performance. The module controller enables drivers that provide an abstraction layer between the software and hardware architectures, and also drives a multi-channel wireless transceiver unit used for inter-module communications. The transceiver selected for the prototype is the nRF24L01 from Nordic VLSI [10], operating in the 2.4 GHz ISM band. It is a single-chip radio transceiver consisting of a fully integrated frequency synthesizer, a power amplifier, a crystal oscillator, a modulator, and built-in power-down capability. Output power and frequency channels are easily programmable.

The next layer is the transmission gate layer, providing an interface between the hardware architecture and the face connectors which are discussed in detail in Section III-C. The layer consists of five transmission gates, each dedicated to one face of the module. The gates are connected to the general purpose input/output (GPIO) bus of the architecture. The gates send signals from the controller/communication layer and receive control signals from the respective face connectors. These signals are then processed by the controller/communication layer to determine the pose information for the module.

The power layer forms the next layer of the stackable architecture. It supplies voltage and current to the various layers in the hardware architecture. The layer performs voltage regulation to convert the input voltage from a battery or external power modules to a voltage range of 3.3 V to 9 V. A supervisory circuit with battery back-up may be employed in the power layer to permit a smooth transition between the connection of shared power modules and batteries.

The final layer of the hardware platform is the interface layer that supports a variety of sensors and actuators. The layer requires all analog signals to be represented as a voltage. Hence, signal conversions such as current to voltage or frequency to voltage are performed externally. The interface layer performs basic analog signal conditioning such as amplification. The amplified signals are digitized by a 10-bit analog-to-digital converter on the module controller. The layer also provides an interface for the most popular digital sensors such as 1-wire, I^{2}C 2-wire [11], Maxim 3-wire, SPI 4-wire [12], and MICROWIRE 4-wire interface standards. RS-232 and general purpose input/output (GPIO) interfaces are also included. Depending on the power requirements, the interface layer may send the control signals to actuators directly or direct them to appropriate external hardware drivers.

B. Module Geometry

The hardware stack is encapsulated in a case which serves as the platform for interfacing sensors and actuators. The geometry of the case plays an important role in determining the flexibility of connecting modules in different configurations. Several factors were taken into account while selecting the case geometry. These include the number of assemblies supported by a particular geometry type, the stability of the module, and the ease with which hardware can be accommodated inside the module. Regular polyhedrons emerged as the strongest candidates due to the fact that they can support a number of different orientations. Also, regular polyhedrons are easier to work with since all of the faces are identical.

The case geometry of the TIMs is a cube, as shown in Fig. 2. A cube supports different assemblies of modules and can easily accommodate the stackable hardware architecture discussed in Section III-A. The cube itself is modular in nature and is assembled by connecting six faces. One of the faces is recessed (see Fig. 2) and is reserved for connection to a sensor or actuator, while the remaining five faces have a mechanism to enable the physical connection of modules. This design allows sensors to be securely housed inside the module. Other sensors that need to be exposed to their environment, such as cameras and temperature sensors, can be placed on the top of the face.

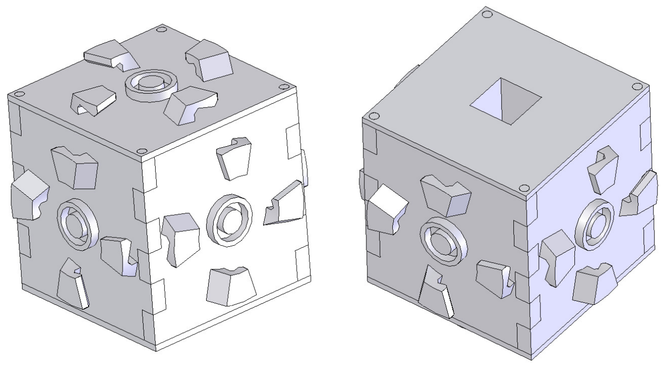
Fig. 2. Geometry of the module

C. Face Connectors

Each face connector consists of four mechanical clips and a central power contact, Fig. 3. The clips enable the physical connection of the modules to one another in four different orientations (90 degrees apart) and the power contact provides a power bus that can be shared between the modules. The clips are asymmetrical, arranged radially around the power contact.

In order to connect two modules, one module is held at an angle with respect to the other, and rotated clockwise to bring the clips into contact and lock the two modules. The locking mechanism fully constrains the movement of the connected modules in every direction except for counter-clockwise axial rotation. In this mode, movement is constrained by the angled clips, which present spring and frictional resistance to applied counter-clockwise torque. Should the clips prove insufficient for a particular application, rods may be inserted into holes drilled through each face. While the design supports a number of different assemblies, it is not suitable for building a dense assembly of modules due to the fact that the modules have to be held at an angle with respect to each other to facilitate locking. One possible way of overcoming this problem is to mount the clips on a rotating disc. In this case the disc, and not the module, needs to be rotated for a connection. This design would allow the modules to be tightly packed.

The functionality of a composite system is determined by the pose of the connected modules. The overall pose of the system can be established by determining the relative orientations and face connections (faces are in contact when two modules are physically connected) between all of the modules present in a composite system.

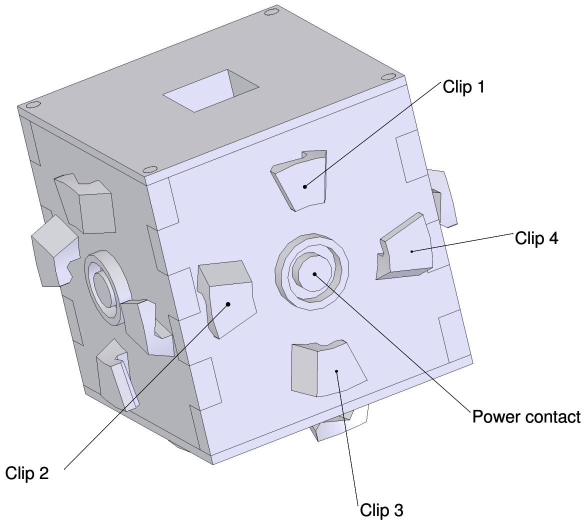

Fig. 3. Face connector

The module utilizes the clips on each face to determine the face connectivity and relative orientation information. The transmission gate layer, as discussed in Section III-A, provides an interface between the clips and the hardware architecture. The controller/communication layer sends and receives signals from the clips via the transmission gate layer. The controller

TABLE I
POLLING SEQUENCES FOR DETECTING FACE CONNECTION, RELATIVE ORIENTATION, AND TRANSMIT-RECEIVE PAIR FOR A FACE

Relative Orientation	Master Clip $\xrightarrow{\text{Request}}$ Slave Clip	Master Clip $\xleftarrow{\text{Acknowledgement}}$ Slave Clip	Master Clip $\xrightarrow{\text{Tx}}$ Slave Clip Master Clip $\xleftarrow{\text{Rx}}$ Slave Clip
1	$1 \longrightarrow 1$	$2 \longleftarrow 4$ $3 \longleftarrow 3$ $4 \longleftarrow 2$	$1 \longrightarrow 1$ $4 \longleftarrow 2$
2	$1 \longrightarrow 2$	$2 \longleftarrow 1$	$1 \longrightarrow 2$ $2 \longleftarrow 1$
3	$1 \longrightarrow 3$	$3 \longleftarrow 1$	$1 \longrightarrow 3$ $3 \longleftarrow 1$
4	$1 \longrightarrow 4$	$4 \longleftarrow 1$	$1 \longrightarrow 4$ $4 \longleftarrow 1$

processes these signals to determine the face connectivity and relative orientation information.

The controller utilizes a send request-receive acknowledgement scheme to determine the face connection between modules. For each face, the controller executes a polling sequence twice. Based on the polling sequence, the controller assigns different functions to the clips. In the first sequence, the controller tries to establish the identity of the face as master. The controller sets Clip 1 as output while Clips 2, 3, 4 serve as inputs. It then sends a request signal on Clip 1 and monitors Clips 2, 3, 4 for a random period of time for acknowledgements. If the controller detects acknowledgement on one or more clips, it assigns the face as master. Based on the clips receiving the acknowledgement, the module controller determines the orientation of the module. In the relative orientation '1', Clips 2, 3, 4 receive acknowledgements. The remaining three orientations are determined based on the individual clips receiving the acknowledgement.

The module controller executes a second polling sequence to establish the identity of the face as slave in the event that the first sequence fails. In the slave mode, the controller sets all the clips as inputs and monitors the clips for a random period of time for request. If the controller detects a signal on any of the clips, it assigns the face as slave. The controller then sets one or more clips as output and sends acknowledgement on the output clip(s). In the slave mode, the relative orientation is determined based on the clip receiving the request signal. On detecting an input on Clip 1, the slave establishes the relative orientation as '1' and sends acknowledgement on the Clips 2, 3, 4. In the remaining three orientations, only Clip 1 is set as the output.

Once a face connection has been established, the controller determines the transmit-receive pair for the face in order to exchange information with the connected module. The transmit-receive pair is formed based on the relative orientation of the connected modules. In relative orientation 1, the master transmits information on Clip 1 and receives information on Clip 4, while the slave transmits information on Clip 2 and receives information on Clip 1. In the remaining three orientations, the master and the slave always transmit information on their respective Clip 1s. The receive contact for the slave is determined by monitoring the clip receiving the request from the master. Similarly, the receive contact for the master is determined by monitoring the clip receiving acknowledgement from Clip 1 of the slave. Table I summarizes the master and slave modes for different face connections.

The information exchanged between the modules include the module address, face number, relative orientation, and the initial position and orientation of transducer with respect to module (obtained from the TEDS fields, discussed in Section IV). The module controller executes the polling sequences periodically to check for both disconnections and new connections on each face of the module.

IV. Knowledge Representation Scheme

In order to make a composite system functional, a logical algorithm [8] is loaded into each module in the system. The algorithm imparts a collective identity to the composite group, and processes data based on the capabilities of the transducers. To facilitate this process, each module must be able to store and communicate its capabilities. A Transducer Electronic Datasheet (TEDS), similar to the definition of TEDS presented in the IEEE 1451 standards [13], is designed to enable the various modules to communicate standard information required by a logical algorithm to assign a collective identity to the composite system.

A. Basic TEDS

Each module contains a Basic TEDS, as shown in Table II, that uniquely identifies the associated transducer. The basic TEDS is made up of 64 bits, and is a minimum requirement for each module.

TABLE II
Basic TEDS content

Manufacturer ID	Model number	Version letter	Version number	Serial number
14 bits	15 bits	5 bits	6 bits	24 bits

TABLE III
EXAMPLES OF STANDARD TEMPLATE FIELDS

Sensitivity and Mapping Properties	
Field name	Description
%Sens	Sensitivity of transducer
%MinElecVal	Minimum value of electrical signal range
%MaxElecVal	Maximum value of electrical signal range
⋮	⋮
Electrical Signal Properties	
%ElecSigType	Type of electrical signal (enumerated)
%RespTime	Response time
%DiscSigType	Discrete signal type
%Gain	Gain of preamplifier
⋮	⋮
Calibration Properties	
%CalDate	The date of the last calibration
%CalInitials	Calibration initials
⋮	⋮

B. Templates

The remaining data contained in the TEDS may be defined by a standard IEEE template, a manufacturer published template, or a user defined template. Templates are used to read or write TEDS data, and describe the memory structure which defines the transducer identity, and provides transducer specific data. The IEEE 1451.4 defines standard templates, containing a standard set of properties such as sensitivity and mapping properties, electrical signal properties, and calibration properties, for common classes of transducers. Each property has several fields that provide a complete description of the transducer characteristics. A standard set of fields remain fixed for each template. Table III lists a sample of the standard fields which are common to all transducers.

In addition to the fields found in a standard TEDS, an additional field is appended to the standard TEDS in order to define a position and orientation of a transducer with respect to its associated TIM, shown in Table IV. This field is referred to as the Pose field. As discussed in Section III-B, the geometry of the TIMs is a cube. By assigning a coordinate frame to each face and performing homogeneous transformations, the position and orientation of the transducer with respect to each face of the module is determined and stored in the Pose field. This information is then utilized by the composite system to establish the overall pose of the system.

V. ADDITIONAL MODULES

In order to make a fully functional composite sensing system, a few additional module types may be required. These additional modules include interconnect modules, power modules, and administration modules. The modules do not perform sensing and actuation tasks and, depending on the application, may be absent altogether.

TABLE IV
USER DEFINED TEMPLATE FIELDS

Pose Properties	
%Position	x_i, y_i, z_i, coordinates of transducer with respect to each face (i = 1, 2, …, 6)
%Orientation	Orientation (θ_i, γ_i, α_i) of transducer with respect to each face (i = 1, 2, …, 6)

A. Interconnect Modules

To support more complex configurations, an assembly of connected modules may require angular or translational offsets. An efficient way to support complex assemblies of modules is through the use of additional modules called interconnect modules. Interconnect modules are available in various shapes and provide a variety of angular and translational offsets (e.g., 45°, 30°, and 5 cm).

Interconnects are passive modules containing an embedded microcontroller that stores a datasheet describing the attributes of the interconnect. The modules are powered by the TIMs to which they are connected. All communication on interconnects occurs through a wired interface, eliminating the need for wireless units, reducing overall power consumption and system cost. Interconnects use the same connection mechanism as the module face connectors, discussed in Section III-C.

Once interconnects are connected to modules, they must inform the modules about their basic properties. In addition, the interconnects need to identify if they are a part of an assembly of interconnects or simply connected directly between two modules. An assembly of interconnects assumes a single identity representing the overall offset provided.

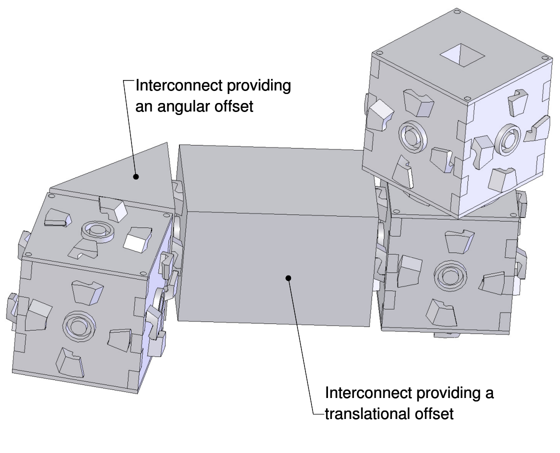

Fig. 4. An assembly of modules using interconnects

B. Power Module

This module supplies power to the various modules in a sensing system, and should maintain a regulated output voltage under variations in supply voltage or load. The design of a power module is not fixed. Power modules may be based on rechargeable batteries or derive power from a wall outlet. They may also include additional hardware to perform signal conditioning such as rectification and filtering to provide a constant DC output. Depending on the application, a composite system may require only one power module capable of supporting all modules, or require power modules to be connected in parallel in order to increase the available current. TIMs can utilize and share these power sources using the common power bus on their face connectors.

C. Administration Module

This module can be viewed as a system manager used to allow a user to interact with and control TIMs in its vicinity through a graphical user interface. It may be a self-contained unit consisting of a power supply, a module controller, and a transceiver, or be implemented as part of a computer system. Its administrative functionality is provided by the software architecture running on it. Using the software architecture described in [8], an administration module detects the presence of TIMs in its vicinity by listening for packet transmissions on a special control channel, and also serves as a local repository for processing algorithms used to implement the cooperative behavior of TIMs.

VI. Software Architecture Overview

This section discusses briefly the features of the software architecture utilized to support the cooperative operation of modules [8]. The key features of the software architecture include support for the decentralized operation of the various modules in a sensing system and enabling suitable hardware-independent processing algorithms to be automatically loaded onto nearby connected TIMs, depending on their positions and orientations, from a remote repository stored on an administration module. These processing algorithms enable TIMs to cooperate with each other. The software architecture also provides the system with the ability to adapt to the addition, removal, and failure of modules.

The software architecture is built upon a pre-emptive real-time operating system (RTOS), which enables a TIM to operate as a part of multiple composite systems simultaneously. A virtual machine running on the top of the RTOS enables processing algorithms to be written using abstract instructions, enabling the algorithms to be written once and be independent of the TIM hardware architecture thereafter. This allows the algorithms to be utilized on any hardware architecture that supports the virtual machine, even if these hardware architectures vary internally or are modified.

VII. Conclusions and Future Work

In this paper, the basic building blocks of adaptive modular sensing systems, known as Transducer Interface Modules (TIMs), have been discussed. These modules have embedded processing and communication capabilities, and may be interfaced to a broad range of sensors and actuators. The user has the freedom to connect these sensing and actuating units in various configurations to create custom solutions. A knowledge representation scheme which allows the modules to communicate their capabilities and form a collective identity is discussed. The design of additional module types (interconnects, power modules, and administration modules) is also described.

Future work involves building hardware prototypes and testing the hardware architecture to determine if it satisfies the requirements outlined in Section II. The design of the face connectors will also be implemented and evaluated.

Acknowledgments

The financial support of the Natural Sciences and Engineering Research Council of Canada (NSERC) is gratefully acknowledged.

References

[1] Crossbow Technology Inc., “Motes, smart dust sensors, wireless sensor networks.” [Online]. Available: http://www.xbow.com/Products/Wireless_Sensor_Networks.htm, 2004.

[2] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, “System architecture directions for networked sensors,” in *Architectural Support for Programming Languages and Operating Systems*, (Cambridge, MA), pp. 93–104, November 2000.

[3] L. E. Holmquist, H.-W. Gellersen, G. Kortuem, S. Antifakos, F. Michahelles, B. Schiele, M. Beigl, and R. Maze, “Building intelligent enviornments with Smart-Its,” *IEEE Computer Graphics and Applications*, vol. 24, pp. 56–64, Jan–Feb 2004.

[4] S. Cotterell, K. Downey, and F. Vahid, “Applications and experiments with eBlocks — electronic blocks for basic sensor-based systems,” *IEEE Sensor and Ad Hoc Communications and Networks (SECON)*, pp. 7–15, October 4–7, 2004.

[5] S. Cotterell, R. Mannion, F. Vahid, and H. Hsieh, “eBlocks — an enabling technology for basic sensor based systems,” in *IPSN Track on Sensor Platform, Tools, and Design Methods for Networked Embedded Systems (SPOTS)*, (Los Angeles, California), pp. 422–427, April 2005.

[6] N. Edmonds, D. Stark, and J. Davis, “MASS: Modular architecture for sensor systems,” in *Proceedings of the Fourth International Symposium on Information Processing in Sensor Networks*, (Los Angeles, CA), pp. 393–397, April 25–27, 2005.

[7] T. D. Ngo and H. H. Lund, “Modular artefacts,” *Position paper for ECOOP 2004 workshop: Components-oriented Approaches to Context-aware Computing*, June 14, 2004.

[8] A. C. Lyle and M. D. Naish, “A software architecture for adaptive modular sensing systems,” in *Proceedings of the 2007 International Conference on Systems, Man, and Cybernetics*, (Montréal, Canada), October 7–10, 2007.

[9] Philips Semiconductors, “LPC2141/42/44/46/48 Single-chip 16-bit/32-bit microcontrollers; up to 512 kB flash with ISP/IAP, USB 2.0 full-speed device, 10-bit ADC and DAC,” August 2006.

[10] Nordic Semiconductor, “Single chip 2.4 GHz transceiver nRF24L01,” March 2006.

[11] Philips Semiconductor, “The I^2C-bus specification,” January 2000.

[12] Motorola, Inc., “SPI block guide V03.06,” Feburary 2003.

[13] National Instruments, “An overview of IEEE 1451.4 Transducer Electronic Datasheets (TEDS).” [Online]. Available: http://standards.ieee.org/regauth/1451/IEEE_1451d4_Templates_Tutorial_061804.pdf.