Reconfigurable Wireless Interface for Networking Sensors (ReWINS)

Harish Ramamurthy, B.S. Prabhu, and Rajit Gadh

Wireless Internet for the Mobile Enterprise Consortium (WINMEC)
The Henry Samueli School of Engineering and Applied Science, UCLA
38-138/A, 420 Westwood Plaza,
Los Angeles, California 90095
{harish, bsp, gadh}@wireless.ucla.edu

Abstract. Remote Monitor & Control systems are increasingly being used in security, transportation, manufacturing, supply chain, healthcare, biomedical, chemical engineering, etc. In this research, attempt is to develop a solution of wireless monitoring and control for such industrial scenarios. The solution is built on two components – a generic wireless interface for remote data collection/actuation units and control architecture at the central control unit (CCU) for smart data processing. The data collection/actuation unit (sensors/actuators) is intelligent by virtue of smart-reconfigurable-microcontroller based wireless interface, which is reconfigurable using Over-the-Air (OTA) paradigm. The RF link is also reconfigurable to accommodate a variety of RF modules (Bluetooth, 802.11 or RFID) providing plug-n-play capability. These capabilities make the interface flexible and generic. The control architecture supports services such as naming, localization etc., and is based on JavaBeans, which allows a component level description of the system to be maintained, providing flexibility for implementing complex systems.

1 Introduction

The current generation automation systems, control & monitoring systems, security systems, etc., all have the capability to share information over the network and are being increasingly employed to aid real-time decision support. The inter-device communication in such systems can be leveraged to maximize the efficiency and convenience in a variety of situations. Intelligent wireless sensors based controls have gained significant attention due to their flexibility, compactness and ease of use in remote unattended locations and conditions. These wireless sensor modules can be designed to combine sensing, provide in-situ computation, and contact-less communication into a single, compact device, providing ease in deployment, operation and maintenance. Already large-scale wireless sensor networks having different capabilities are being used to monitor real-time application needs.

Different types of sensors (thermal, photo, magneto, pressure, accelerometers, gyros, etc.) having different capabilities, interfaces and supporting different protocols are used for different applications. For remote data collection, RF communication links with different characteristics, such as frequencies, data carrying capacity, bandwidth, susceptibility to interference, power needs, etc., have been employed to transmit the data. The appropriate design of the wireless sensing device depends on the end application needs such as data bandwidth, range, interference, etc.

In the proposed work the attempt is to provide a single comprehensive architecture to support the diverse control automation needs of a variety of industrial applications. The data collection/actuation units will be made intelligent by equipping them with smart microcontroller based wireless interface. Utilizing the plug-n-play modularity of the system architecture, appropriate sensor interfaces and RF communication interfaces can be chosen according to the application requirements. Further, once the interfaces are chosen, the (developed) application interface could be used to implement the specifics like description, placement, interaction of sensors and actuators, etc. For example; the interaction could be a "closed loop" control of a motor where the sensor is an encoder and actuator a motor.

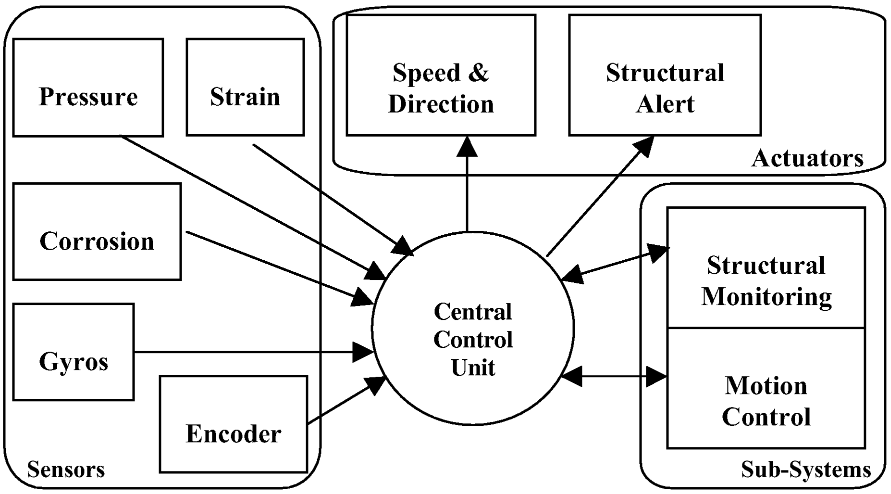

Fig. 1. Application Scenario – Aircraft health monitoring; the arrow direction denotes the direction of signal flow

1.1 Typical Application Scenario

A typical application scenario for our system can be aircraft sub-systems' monitoring.

As depicted in Fig. 1 monitoring the operational status and condition of a number of aircraft sub-systems such as structural monitoring, fuel level, motion control, etc., involves data retrieval and processing from a high-density of sensors, and provide real-time decision support. Each such subsystem may have multiple sensors installed, which continually update the status of the aircraft. The CCU is responsible for continuously aggregating real-time information from sensors and disseminates it to each of the respective subsystem for appropriate response (to actuators). Benefits of wireless communication such as significant space reduction, lower maintenance costs (as armored cables are done away with) and flexibility of deployment can be suitably leveraged. Further, utilizing the proposed reconfigurable and plug-n-play functionality in this research, the sensing system can be upgraded or updated with lesser effort.

1.2 Organization of the Paper

The various sections of the paper are as under. Review of the current published work done in this research field constitutes section 2. A brief overview of the system in explained in Section 3. Section 4 and 5 describe the intelligent wireless sensor architecture and the structure of a micro-controller that provides the intelligent interface between the sensor and RF module. In Section 6, the developed control architecture or application interface is explained. Implementation details, snapshots and some simulation results of the current implementation will be presented in Sections 7 and 8. Finally Section 9 reports the conclusions on the work done so far.

2 Related Work

Early work in the field of wireless sensor networks finds its roots in DARPA's military surveillance and distributed sensor network project, low-power wireless integrated micro-sensor (LWIM) and the SenseIT project. Through these initiatives DARPA has been quite instrumental in pushing the field from concept to a deliverable implementation form. Some of the relevant related research is described here in brief.

Wireless Integrated Network Sensors or the WINS project and NIMS project [1, 2] at University of California, Los Angeles is about ad-hoc wireless sensor network research – dealing mainly with building micro-electronic mechanical sensors (MEMS), efficient circuit design, and design of self-organizing wireless network architecture. Though these projects have been successful in demonstrating a network of self-organized sensor wireless nodes, they seem to have a bias towards environmental and military applications. Also they use proprietary RF communication technology and hence the solutions are restrictive for wide scale deployments in industries.

Motes and Smart Dust project [3] – at University of California, Berkeley involved creating extremely low-cost micro-sensors, which can be suspended in air, buoyed by currents. Crossbow Inc. has commercialized the outcome of this project. Here again the solution is restrictive, as proprietary communication technologies have been used to achieve inter-device communication. Further, the focus has been on development of sensors and their interaction rather than how the sensors will be integrated to form systems (simple or complex). This is generally termed as the "bottom-up" [4] approach, which may not be suitable for building complex systems.

Pico-Radio [4] – A group headed by Jan Rabaey at University of California, Berkeley is trying to build a unified wireless application interface called Sensor Network

Service Platform. An attempt is to develop an interface that will abstract the sensor network and make it transparent to the application layer. A preliminary draft describing the application interface has been recently released [4]. They believe in a "top-down approach" (from control to sensor nodes) for building sensor networks, which is probably more suitable for building complex systems.

Recently, there have been several initiatives like TinyDB [5], Cornell's Cougar [6] etc. to develop a declarative SQL-like language to query sensors and define certain standard query services. Here the implementation is sensor-interface specific and not a generic or abstracted sensor-networking platform. These query services can be implemented with ease on top of our (developed) wireless interface and sensor-networking platform and can be made generic by extending them for other sensors.

Other research initiatives in this field include MIT's μAMPS [7], Columbia University's INSIGNIA [8], Rice University's Monarch [9]. For a more detailed literature survey readers are encouraged to refer [10].

Though there have been a lot of research efforts in developing ad-hoc wireless networks, the focus has been on developing smart wireless sensor interfaces and not much attention has been paid to the actual application integration. Typical approach has been to develop powerful smart wireless interfaces, which supports the important features/requirements for a particular class of applications (like military, environment sensing or more focused applications like fuel-level control in automobiles). The result is a plethora of wireless interfaces appropriate for a certain class of application; but almost no interoperability between them. We believe that the deployment of wireless infrastructure in industries will occur in incremental stages and thus interoperability (between different sensor-networks) and extendibility (according to application needs) will form the basic requirements of any prospective solution. A prospective good solution would be an end-to-end solution, which is modular and extendable and hence capable of addressing the needs of majority of industrial applications, if not all. The ReWINS research initiative is an attempt to develop such an end-to-end solution with support for incremental deployment through a transparent lower layer implementation and control architecture and a user-friendly application interface.

3 System Overview

In this section, we present a brief overview of the system architecture. The system consists of a network of sensors, actuators and aggregators communicating with the central control unit using standard RF-links. The basic scenario is shown in Fig. 2. Here D represents the devices, which can be sensors or actuators and A is the aggregator. The scenario we consider is an infrastructure based deployment. In a typical industrial scenario like aircraft systems monitoring or automotive control, the sensors are typically stationary and have fixed wireless communication links, unlike mobile ad-hoc scenario. Fixed wireless communication links are more reliable than mobile links, which have inherent problems such as loss of connection, varying data rates, etc. Fixed links help in providing certain performance guarantees in terms of data-rate, mean delay, etc., which are essential for deployment in industrial scenarios. We thus believe an infrastructure-based deployment (i.e. fixed wireless deployment) is more suitable in such scenarios.

We term sensor as any kind of transducer which is capable of exchanging information in the form of electrical signals and similarly actuator is any kind of device which will accept data in the form of electrical signals and perform a measured action. Sensors and actuators will be referred generically as devices henceforth. The main function of the aggregator is to collect the data and signals from/to the devices and using a backhaul link (Wi-Fi) to transmit it back to the CCU.

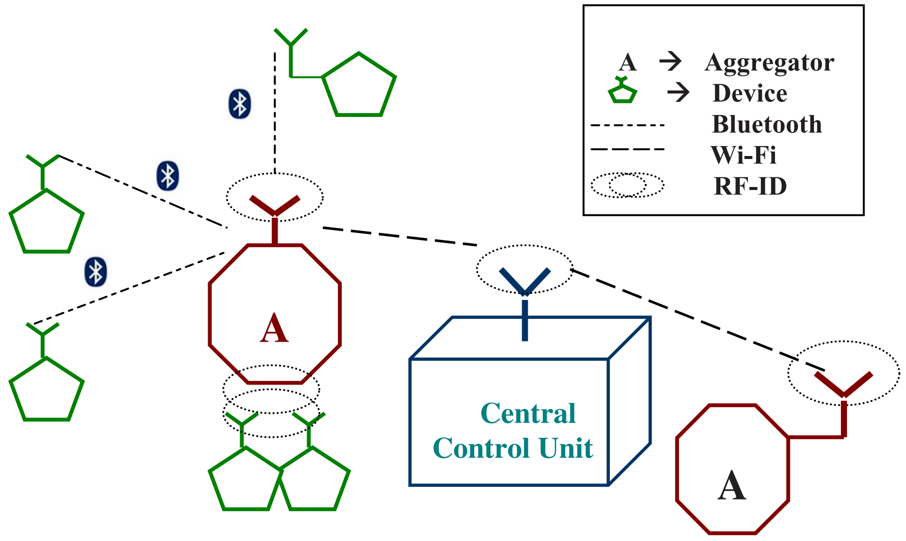

Fig. 2. System Overview

The features of the proposed system include:

1. Reconfigurability: Central Control Unit can set the run-time parameters of the device and/or update/upgrade the firmware of the system over the air (OTA).
2. Plug-n-play: Depending on the application needs/requirements, different infrastructure and their configurations can be deployed quickly.
3. Self-calibration: the intelligent adaptive sensors can accurately measure data and self-calibrate without significant user intervention.
4. Wireless connectivity: provide bi-directional communication over a wireless connection.
5. Lower installation and maintenance cost as no wiring/cables, etc., is required and therefore enabling greater acceptance and speedy deployment.

4 Aggregator – Functions and Description

The Aggregator has been introduced in the architecture to deal with the case of a highly dense network of distributed sensors and where it may not be possible for all sensors to directly communicate with the central control unit. The problems encountered in a direct sensor-control unit network can be summarized as:

1. Paucity of communication channels – The number of RF communication channels available in a given space are limited; thus limiting the number of sensors that can be present in the system. The aggregator with spatial channel re-use functionality will help improve the number of sensors that can be present in the system. Further, this will also bring down the power requirements (smaller transmission range) at each device and thereby increasing the battery life of the device.
2. Diverse sensors and their requirements – Different sensors have diverse capabilities and requirements. For example, a corrosion sensor may not need a stringent monitor/control as a motor rotary system. Also different sensors and different application needs warrant wireless communications with a variety of bandwidth and range requirements. The aggregator model efficiently addresses these issues by supporting multiple wireless interfaces and aggregating the payload.

An aggregator collects data and enables signal flow from the devices and sends it to the CCU and forwards the data/signals from the CCU to the respective recipient devices. The important functions of the aggregator will now be described.

4.1 Connection Establishment and Maintenance

The aggregator maintains connections with the devices within its purview and central control unit. Both connections are of master-slave type. In the device-aggregator connection, an aggregator is the master while in the aggregator-CCU connection the CCU is the master. In this type of connections, the master initiates connections and is responsible for maintaining them. For each type of device, the aggregator maintains the list of devices (i.e. device IDs – Section 5) connected to it. For the backhaul connection i.e. aggregator-CCU, Wi-Fi is used; the devices can use different RF technologies to communicate with aggregator, which supports multiple wireless interfaces.

4.2 Data Forwarding

The aggregator receives data from both the devices & CCU and forwards it to its intended recipients. Data from devices are marked with their specific device IDs at the aggregator and sent to CCU. Data from CCU is marked with device IDs and the aggregator extracts this information and forwards the data to the respective-device.

A typical payload is shown in Fig. 3. The data payload, i.e. the particular order in which bits will be transmitted, is device-specific and can be tailored to application needs. For example; the data payload could be a trigger or acknowledgement as speci-

Aggregator – Central Control Unit Communication

Start	Device ID	Data Payload	Stop

Fig. 3. Payload over the wireless link

fied in IEEE 1451.2 standard or it could be a simple data measurement byte. The format of data payload is fixed during initialization of the device (Section 6.1) and can be modified while the device is in Command mode.

4.3 Exception Handling

In the event of wireless connection drop between the aggregator and any of the devices, the aggregator reports to the CCU about the loss of connection. This helps differentiate between device inactivity and connection loss. The CCU then takes appropriate steps to compensate for the loss of device. This for example includes: entering a "safe-state", instructing other aggregators to start looking for the device, raising an exception, etc. In the event of connection severance with CCU, the aggregator instructs all the devices to enter the "safe-state" and wait for the CCU to respond.

5 Device Architecture

Architecture of the wireless intelligent data collection device is shown in Fig. 4. The device is composed of three components: a Sensor/Actuator, Microcontroller interface and RF interface. Microcontroller acts like an intelligent interface between the sensor or the data collection unit and the RF module. It handles tasks related to data collection, data processing and wireless transmissions. The RF trans-receiver communicates with the aggregator or CCU over the RF-link.

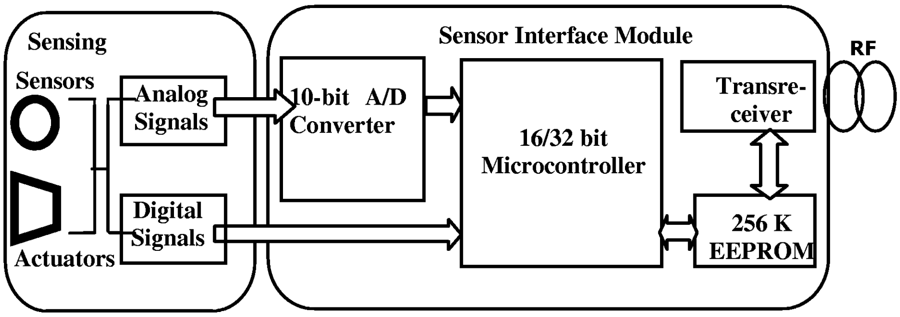

Fig. 4. Hardware design of Intelligent Sensor

5.1 Software Architecture

The device is implemented as an asynchronous finite state machine as shown in the Fig. 5. The device while in operation will be in any one of the two modes: Command Mode and Data Mode. This design facilitates flexible configuration i.e. forming networks, setting up device specific parameters, etc., while the device is in command mode. Further, once the device is configured, the device data (i.e. sensed data or actuator instructions) can be communicated without overhead in the data mode utilizing the wireless link efficiently. We now describe each of the modes in detail.

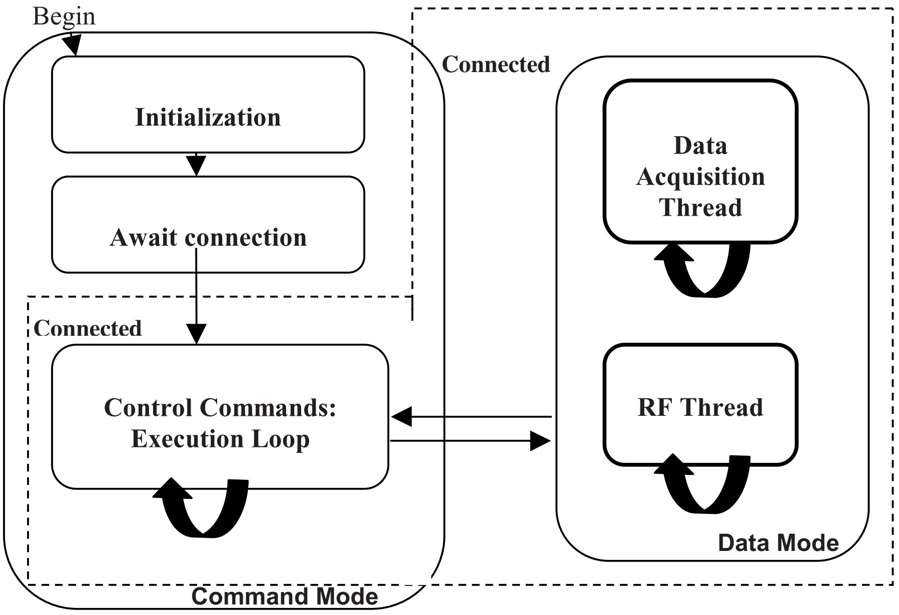

Fig. 5. Software Architecture of the Device (State Machine)

5.2 Command Mode

The device enters the command mode by default and the first task is the configuration of the wireless interface, after which the device awaits for an authenticated connection. Thus two states are identified in this mode – the “connected” state and “unconnected” state. Upon connection the device still remains in command mode and configures itself as a slave and waits for instructions from master. In case the connection is dropped anytime the device resets itself and waits for a fresh connection.

The control unit typically queries the device type information and sets the format of payload (Sect. 4.2 & 6.1) to be used in command & data mode. It then instructs to start the device specific configuration procedure. Typically in case of an analog device, the A/D or D/A converters are initialized. Further, the control unit may instruct the device to switch to data mode. The device can be reconfigured in this mode.

5.3 Data Mode

Once in Data mode the device will transmit/receive the device data using the RF-link. The device can be switched back to command mode using an escape sequence through the wireless interface. Otherwise the device will remain in data mode and transmit/receive data in the prescribed format. In the data mode, the device has to perform the following two tasks – 1) data collection in case of sensor or issuing instructions in case of actuator, 2) Interfacing and communicating with RF-link. As real-time processing of these tasks is required, separate threads are run for each task on the device.

5.4 Connection Drop and Exception Handling

The device periodically checks the status of the wireless link. In case the connection is dropped and cannot be reinstated; the device executes an "emergency routine" where the device is set to a "safe-state", which is particularly necessary in case of actuators. For example, a safe-state for a motor would be a complete stop. After executing the emergency routine the device just waits for the connection from the aggregator/CCU. The device thus can be only in "connected" state in this mode.

5.5 Memory Organizations and Over-the-Air Reconfiguration

The data stored in the microcontroller/EEPROM is divided into five types and each stored in different portion of memory. The memory allocation is shown in Fig. 6. This kind of memory organization helps in supporting reconfigurability, upgrade & update of firmware over-the-air (OTA) and fast real-time data processing in the device.

Self-identification information contains the unique device ID and type information.

The main program is the basic "monitor" program which initializes the system i.e. device hardware (sensor and wireless) interfaces. The main program cannot be altered and is permanent. Services like naming, service discovery, functionality of reconfigurability and other such core services are supported in the main program.

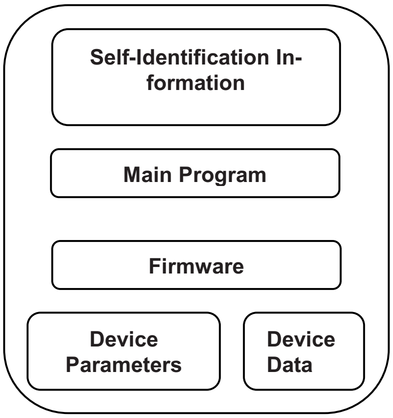

Fig. 6. Memory Organization

The firmware program takes care of data handling, parsing, taking actions and setting run-time parameters. The main program can modify the firmware.

Device Parameters contain the device-specific information like bit accuracy, sampling rate, aggregator identification etc and functions (names) supported by the device like sampling rate modification, toggle switch support etc. The attribute-value pair characterizes these parameters. For function names, the value corresponds to the function version. The device parameter characterization is essential for supporting querying services over the device. Reconfiguration Over-The-Air (OTA) is initiated by the CCU and is carried out through a set of messages exchange between the CCU and device as shown in Fig. 7.

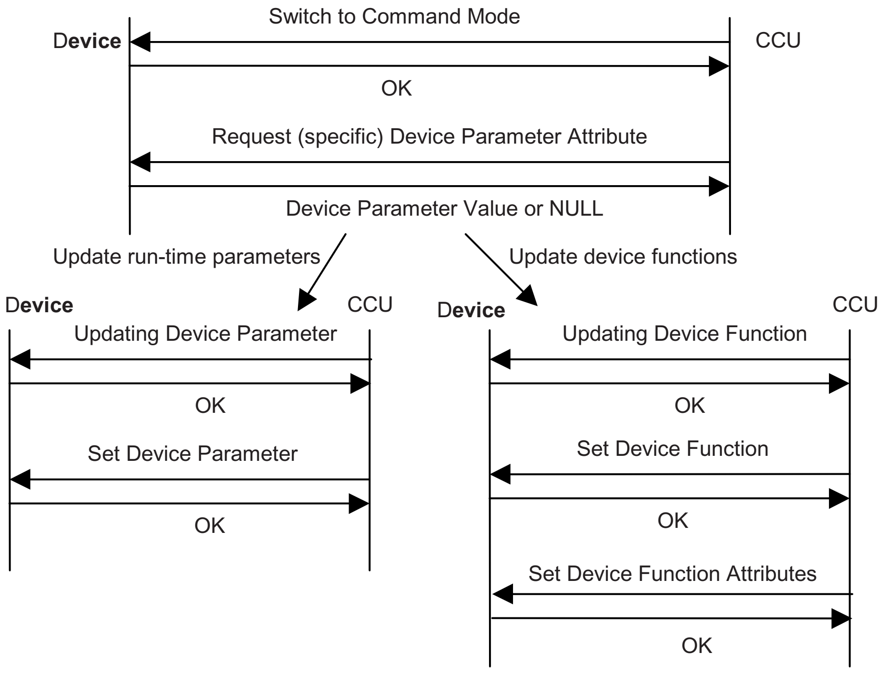
Switch to Command Mode
Device
CCU
OK
Request (specific) Device Parameter Attribute
Device Parameter Value or NULL
Update run-time parameters
Update device functions
Device
CCU
Updating Device Parameter
OK
Set Device Parameter
OK
Device
CCU
Updating Device Function
OK
Set Device Function
OK
Set Device Function Attributes
OK

Fig. 7. Message Exchange Flowchart for Device Reconfiguration (OTA)

5.6 Conformation to IEEE 1451 Standard

The IEEE 1451 [11] group is attempting to standardize the sensor (or actuator) interfaces, i.e. the way they measure and report (or get instructions) parameters and values. More specifically, the standard defines the physical interface – Smart Transducer Interface Module (STIM), the Transducer Electronic Data Sheet (TEDS) and Network Capable Application Processors (NCAPS) – which derives information and controls the transducers. Utilizing the modularity and reconfigurability functionality of our device interface, adhering to IEEE 1451 standard is envisioned in future. The Control unit coupled with aggregator will be the network capable application process (NCAPs) as specified in the preliminary draft of IEEE 1451. The developed sensor interface can be configured to adhere to the STIM specifications and TEDS will be available from each device through a query service (defined by the control architecture).

6 Central Control Unit (CCU) – Software Architecture

This section provides a brief overview of the software architecture of the CCU. The developed system is targeted to be used with complex systems such as aircraft subsystem monitoring where a number of devices have to be identified, probed and instructed after performing some complicated logical computation such as closed loop control, wing balancing, etc. In order to implement complex systems, we had identified that the following features need to be supported by the CCU:

1. Flexibility of implementation – No restrictions on how device should be placed (geographically and logically) and connected to the system
2. Flexibility of control – Allow inclusion of complex control algorithms over devices & the sub-system formed by these devices
3. Friendly user and control interface – A framework by which the system can be described with ease; i.e. using a declarative language like XML.

The software architecture is what drives the central control unit. Thus in order for the system to support the aforementioned features, the software architecture of the control unit has to address the following issues:

1. Device detection, representation and characterization in the system
2. Inter-device interaction & information sharing; formation of sub-systems by these devices and aggregation of subsystems to form the complete system

6.1 Device Detection and Representation

The devices configure themselves as slaves and wait for connection from the central control unit or the aggregator. After initialization the aggregator will start probing for other devices and the control unit. Since the aim is to maximize device detection, coverage and communication, if possible, all device connections will be routed through an aggregator. Devices are represented as entities with data interfaces in the control system software architecture. The data interface provisions the exchange of information among devices.

6.2 Device Query Services

We are currently developing a standard query service model which will support IEEE 1451 and provide a query platform to implement query services like Cornell's Cougar or Berkeley's PicoRadio and Tiny DB, etc. Standard query services like service discovery, identification, device parameters modification etc. are currently supported.

6.3 Formation of Subsystems

A subsystem can be defined as a collection of logically connected devices; for example a simple motor control subsystem will have a motor (actuator) and an encoder (sensor). The logical connection specifies how data present in the device can be exchanged as information, for e.g., the control logic like closed-loop control of a motor control system. Further only those devices, which have some information to share, can be connected. Here we will like to differentiate between "data" and "information". Information is what devices can extract from data to self-adjust and modify their behavior. For example for a motor, rotary position from an encoder is "information" but linear position from a sensor may be just "data". The complete system has a hierarchical architecture and hence facilitates a very user-friendly control interface. The control architecture has been implemented using the Java Beans API [12], where each entity of the system is represented as a component and a reusable software unit. Further using the application builder tools of Java Beans these software units can be visually composed into composite systems. This makes the user interface extremely friendly but still gives the power of building complex systems.

7 Experimental Setup and the Current Status of the Project

Currently the proof-of-concept implementation of the networking platform has been demonstrated. In the current version of the system different type of sensors, viz., rotary and linear have been interfaced. Class -1 Bluetooth has been used for the RF communication link. Work on supporting other RF technologies like UWB and RFID is currently being done. In the current implementation, as the aggregator is just a logical component, the aggregator resides on the CCU itself. Here the aim was to support both "open-loop" and "closed-loop" control on a motor as actuator. An encoder was used to sense the position of motor. Fig. 8 shows the block-diagram and snapshots of the current implementation.

8 Results

Preliminary tests of the system were carried out in certain scenarios and the results are presented below:

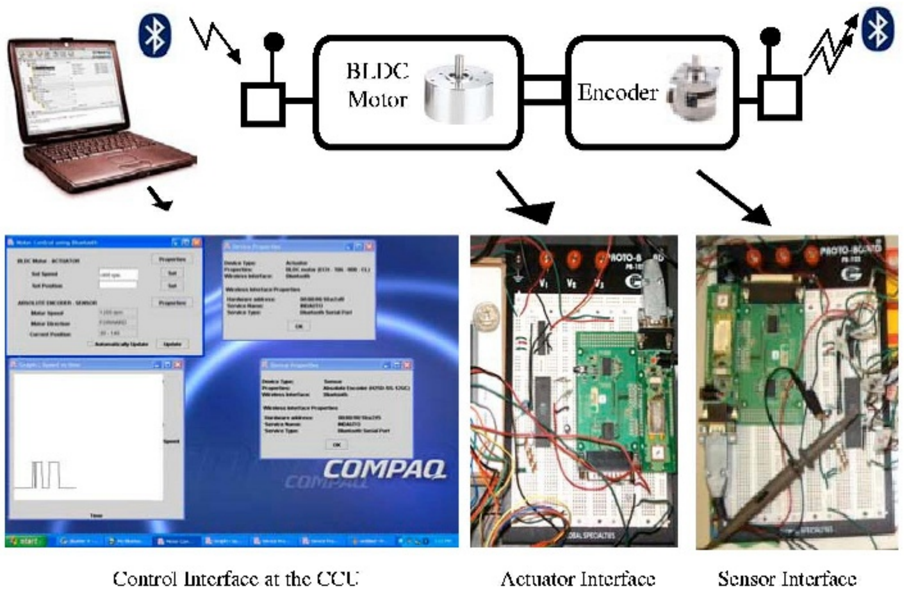

Fig. 8. Snapshots of the Current Implementation

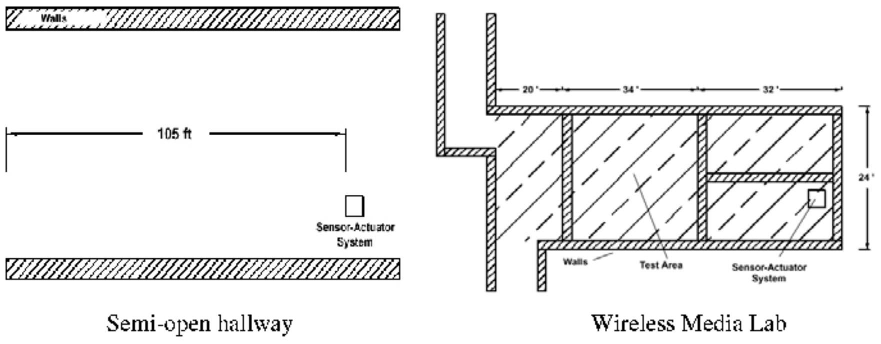

Fig. 9. System Test Scenarios

8.1 Geographical Range Tests

The system was tested in two scenarios as shown in Fig. 9. Here the walls are shown as hatched rectangles and the rectangular box is the sensor-actuator system. The laptop (control-unit) was able to control the sensor-actuator system from 105ft without degradation in the performance. In the second scenario, the hatched area shows where system performance didn't degrade. The system performed well even in presence of obstacles like walls (2).

8.2 Varied Sensors and Actuators

The system in its present implementation can support the following devices - Absolute Encoder, BLDC motor, Gyro Sensor, Incremental Encoder and Linear Position Sensor.

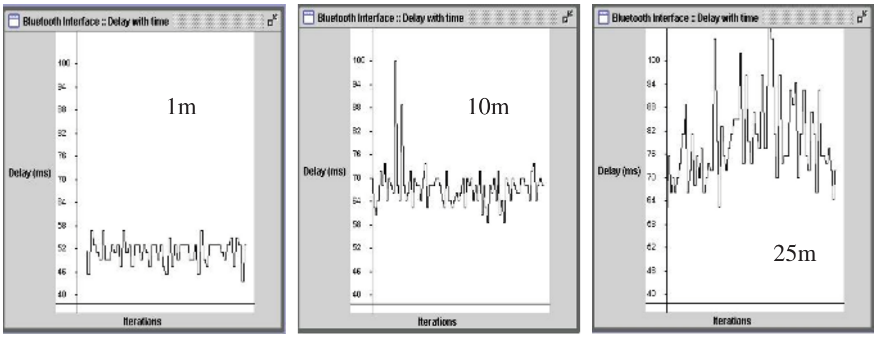

Fig. 10. Delay Performance with varying testing distance

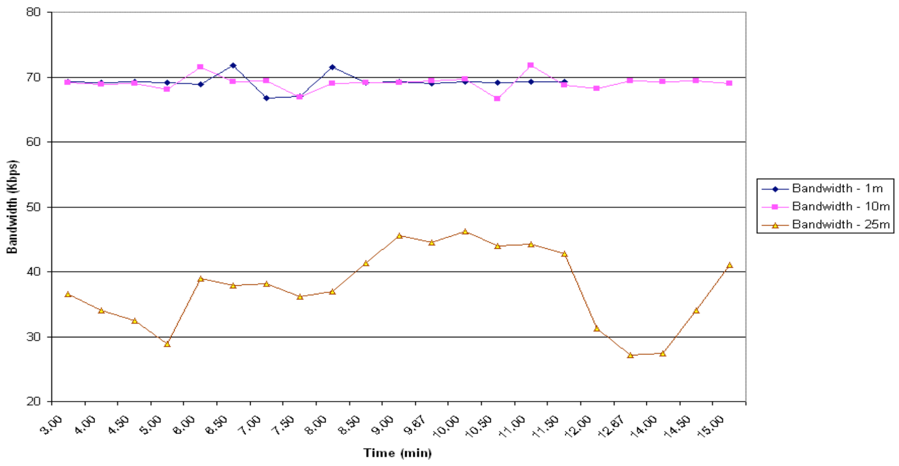

Fig. 11. Bandwidth Performance with distance

8.3 System Performance Tests

Since these systems will be used for real-time monitoring and control, performance studies were carried out. Delay and bandwidth have a significant effect on the fidelity and responsiveness of the system. To characterize the parameters, simulations were performed in an "echo-scenario", i.e. whenever the CCU sends a packet to the device; the device simply echoes back the packet. Measuring the delay from start of transmission from CCU to end of reception at CCU gives the round-trip-delay of the link

Bandwidth is measured by the data rate. Simulations were carried out for different inter-device distances and the results are shown in Fig. 10 and 11.

As can be seen from simulation results the round-trip-delay increases with distance. Further, the delay becomes jittery as the distance is increased. This is attributed to the increased interference and fading at longer distances. Similarly bandwidth decreases with distance and becomes jittery due to the same reason

9 Conclusion

In this research, an end-to-end solution for wireless monitoring & control in industrial scenarios has been proposed. It was shown how traditional sensors could be transformed into intelligent wireless sensors, capable of making real-time decisions using the developed generic wireless interface. A proof-of-concept working model of the solution is also demonstrated with successful integration of a variety of sen sors/actuators by using the developed application interface. To illustrate, a suitable application scenario was also discussed for such a remote data collection system.

Acknowledgements. We would like to thank Xiaoyong Su and other members of Wireless Media Lab and WINMEC [http://winmec.ucla.edu] for their valuable comments and suggestions.

References

1. Wireless Integrated Network Sensors project at University of California at Los Angeles, URL: http://www.janet.ucla.edu/WINS/
2. NIMS – Networked Info-Mechanical Systems project at University of California at Los Angeles, URL: http://www.cens.ucla.edu
3. Smart Dust and motes project at University of California, Berkeley, URL: http://robotics.ee-cs.berkeley.edu/~pister/SmartDust/
4. Pico-Radio project at University of California, Berkeley, URL: http://bwrc.eecs.berkeley.ed-u/Research/Pico_Radio/
5. TinyDB project at University of California, Berkeley, URL: http://telegraph.cs.berkeley.edu /tinydb/
6. Cougar project at Cornell University, URL: http://www.cs.cornell.edu/database/couga-r/index.htm
7. Micro-Adaptive Multi-domain Power-aware Sensors (µAMPS) project at University of California, Berkeley, URL: http://www-mtl.mit.edu/research/icsystems/uamps/
8. Insignia project at Columbia University, URL: http://comet.ctr.columbia.edu/insignia/
9. Monarch project at Rice University, URL: http://www.monarch.cs.rice.edu/
10. Wireless Ad-hoc Network Links at NIST webpage, URL: http://www.antd.nist.gov-/wctg/manet/adhoclinks.html
11. IEEE 1451.2 Standard, URL: http://grouper.ieee.org/groups/1451/2/
12. Java Beans specification, URL: http://java.sun.com/products/javabeans/