Data-logging and supervisory control in wireless sensor networks

Article *in* International Journal of Sensor Networks · January 2009

Impact Factor: 0.92 · DOI: 10.1504/IJSNET.2009.028022 · Source: DBLP

4 authors, including:

Dan Popa
University of Texas at Arlington
153 PUBLICATIONS **1,199** CITATIONS
SEE PROFILE

Prasanna Ballal
Caterpillar Inc.
18 PUBLICATIONS **203** CITATIONS
SEE PROFILE

Data-Logging and Supervisory Control in Wireless Sensor Networks

Aditya N. Das
Automation & Robotics Research Institute, University of Texas at Arlington,
7300 Jack Newell Blvd. S., Fort Worth, TX 76118, USA
E-Mail: aditya@arri.uta.edu

Dan O. Popa
Automation & Robotics Research Institute, University of Texas at Arlington,
Department of Electrical Engineering
7300 Jack Newell Blvd. S., Fort Worth, TX 76118, USA
E-Mail: popa@uta.edu

Prasanna Ballal
Automation & Robotics Research Institute, University of Texas at Arlington,
7300 Jack Newell Blvd. S., Fort Worth, TX 76118, USA
E-Mail: pballal@arri.uta.edu

Frank L. Lewis
Automation & Robotics Research Institute, University of Texas at Arlington,
7300 Jack Newell Blvd. S., Fort Worth, TX 76118, USA
E-Mail: lewis@uta.edu

Abstract: Wireless Sensor Networks (WSN) are being increasingly used in a multitude of applications such as environmental and structural health monitoring, and condition-based maintenance. Even though the sensors collect a vast amount of data, only a tiny fraction of this data may be useful. This paper proposes a data-logging & supervisory control architecture to manage the information gathered by the WSN and make decisions based on this information. We present an application module which would be able to effectively manipulate the sensor data and to support a Discrete Event Controller (DEC). The DEC is responsible for generating rule-based tasks in order to address information-centric issues such as data-logging, alarm & event reporting and security. A combined data-logging and supervisory control framework (DSC) is proposed to address data pre-processing challenges such as acquiring and recording signals, online analysis, offline analysis, report generation, and data sharing.

Keywords: wireless sensor networks, data-logging, supervisory control, discrete event controller, cross bow sensors and mobile robots

Reference to this paper should be made as follows: Das Aditya N., Lewis Frank L. and Popa Dan O. (2004) 'Data-logging and Supervisory Control in Wireless Sensor Networks', *Int. J. of Wireless and Mobile Computing*, Vol. 00, Nos. 0/0/0, pp.000–000.

Biographical notes: Aditya N. Das received his M.S.E.E. from University of Texas at Arlington in 2005. He is currently pursuing his Ph.D. at the Automation& Robotics Research Institute (ARRI) at UTA. His current research interests include wireless sensor networks, robotics and control systems, microrobotics and MEMS.

1. INTRODUCTION

With the advancement of technology through time the physical parameters involved in modern days sensors have grown widely diverse, most of which are beyond human perception. Concurrently, the volume of information, and computational requirements have vastly increased. In order to monitor and manage this constantly expanding dimension of physical parameters, it has become imperative that an

efficient supervisory control system be available to manage sensory data. The supervisor should not only observe and collect data, but also correlate and preserve it efficiently, derive appropriate conclusions, and prepare interpretations with minimum error. Using heterogeneous arrays of tiny sensors and high-tech robots deployed through vast areas of interest and coalesce into smart networked environments are becoming a breathtaking reality. Astounding progress in areas such as computers, sensors, microcontrollers and communication algorithms within the past decade has made it possible to see sensors deployed almost everywhere, whether it is inside a common kitchen microwave oven, or inside a one-of-a-kind Mars Rover.

Whether it is one sensor or many sensors working in unison in a network, whether it is wired or wireless, whether it is for simple measurements or complex applications; the concept behind the scene follows the same simple rule. The deployment and operation of sensor networks encompasses three major aspects: sensing, communication and computing, with their associated hardware, software and algorithms [4]. Significant past research in Wireless Sensor Networks (WSN) has been dedicated to developing bandwidth, energy, or latency efficient communication protocols [16]. As demands increased and popularity grown, WSN applications require increasingly larger and robust networks, additional aspects such as fault tolerance, effective deployment, mobility, and data management [15, 16]. One of the aspects addressed in this paper is efficient data acquisition and processing that plays a vital role in automating a sensor network. In this paper we present data logging and supervisory control environment for WSN that builds upon previous research with tethered sensors and processors.

2. RELATED WORK

Observing and trying to understand the surroundings is the primitive form of sensing and information gathering and perhaps as old as human race. However serious thoughts on habitat monitoring and coordinated data recording came into picture much later, with the industrial revolution in the eighteenth and nineteenth centuries during which period many inventions such as steam engine, cotton mills, automobiles etc greatly moved the globe. Later during the world wars in the early half of the twentieth century and then during the space race period communication techniques developed immensely which laid the foundation for sensor networks. Controller Area Networks (CAN) [1], Wireless Sensor Networks (WSN) [7], concepts of Condition Based Maintenance (CBM) [14], supervisory control concepts are built upon these previous platforms.

2.1 CONTROLLER AREA NETWORK (CAN)

Controller Area Networks (CAN) [1, 2] were originally developed in the 80's for the interconnection of control components in automotive vehicles. CAN enable a huge reduction in wiring complexity and additionally made it possible to interconnect several devices using a single pair of wires allowing data exchange between them at the same time. The ability to send data on an event basis means that bus load utilization can be kept to a minimal amount. The basic features of CAN are high-speed serial interface, low-cost physical medium, short data length, fast reaction time, multi-master & peer-to-peer communication, error detection and correction etc. CAN can be viewed as a real-time version of Ethernet with an emphasis on data exchange with time-delay guarantees. Many types of supervisory controllers have been implemented to manage CAN networks. The CAN framework has a number of limitations such as network faults, issues with scalability and limited range due to requirements of real-time control over physical wires.

2.2 WIRELESS SENSOR NETWORKS (WSN)

The needs of Wireless Sensor Networks (WSN) for data management and communications are not unlike CAN. Development of sensor networks was originally motivated by military applications such as large scale acoustic surveillance such as ocean surveillance and ground target detection by unattended ground sensors (UGS) [3]. However, the availability of low-cost sensors has enabled the development of many other potential applications such as infrastructure security, industrial sensing, environment & habitat monitoring, traffic control etc.

In many applications, WSN require "zero-admin deployment", distributed algorithms, efficient debugging and data collection interfaces. TinyDB is a database style interface that is used for automating the WSN. Several other examples of sensor database visualization and querying can be found in literature [16,17].

TinyDB [2] is an overlay application intended to run X-Bow sensor motes in the sensor network and provide an SQL like interface to the network which is presented as a table of streaming data. Through this application the SQL-like query is entered and transmitted to the network via the base station. The extended SQL language allows for the selection of data from any sensor attached to the motes in the network and can provide once off or streaming data. It can also be used to transmit event based operations to the network.

Though TinyDB and other WSN database schemes have advantages of operating a sensor network efficiently and scalably, they are predominantly network-centric, e.g. they have limitations in interpreting the information contained in the sensor data and defining specific WSN actions based on it. TinyDB stores the data in the EPROM which is extremely small in size for extended database applications. Large amount of data such as historical data trends can not be stored, hindering the applications such as data trend preparation, large-scale event handling, data processing and analysis, data security etc.

2.3 CONDITION BASED MAINTAINANCE (CBM)

With most research efforts targeting on applications like habitat monitoring, area monitoring, surveillance etc, environmental sensing and processing remains the principle stimulant in the evolution of sensor networks. Most of the protocol architectures, viz. S-MAC, PAMAS, are thus designed for applications where data is acquired only when an interesting event occurs or when prompted by user. In contrast to this, also there are applications requiring turn-wise, continuous, periodic, and real-time transmission of data from sensors. One such application is condition based

maintenance (CBM) [15] of machinery and equipment for reliability and health maintenance. Real-time monitoring and control increases equipment utilization and lifetime, and positively impacts system yield and throughput. Distributed data acquisition and real-time data interpretation are two primary ingredients of an efficient CBM system. These two are mutually dependent on each other. Data interpretation algorithms are learning systems that mature with time. Distributed data acquisition should thus be adequate for both machine maintenance and learning by the monitoring system. In control theory terms, one needs both a component to control the machinery and a component to probe or identify the system. Wireless networks have a distinctive edge over wire networks in condition based maintenance. In wired systems, the installation of enough sensors is often limited by the cost of wiring. Previously inaccessible locations, rotating machinery, hazardous or restricted areas, and mobile assets can now be reached with wireless sensors. These can be easily moved, should a sensor need to be relocated.

As the number of sensors deployed in the network increases and their type greatly varies, one major issue that arises is how to manage the large amount of data gathered over a period of time and how to control the entire network to achieve the desired condition based maintenance. A data-logging and supervisory control seems to be a plausible solution.

2.4 *DATA-LOGGING & SUPERVISORY CONTROL*

Data logging and recording is a very common measurement application. In its most basic form, data logging is the measurement and recording of physical or electrical parameters over a period of time. The data can be temperature, strain, displacement, flow, pressure, voltage, current, resistance, power, or any of a wide range of other parameters [11]. Real-world data logging applications such as those required in CBM, are typically more involved than just acquiring and recording signals, and require some combination of online analysis, offline analysis, display, report generation, and data sharing. Moreover, many data logging applications are beginning to require the acquisition and storage of different types of data like analog, discrete etc [12].

The idea of integrating logic with continuous dynamics in the control of complex systems leads to the evolution of supervisory control systems. Supervisory control and data acquisition systems (SCADA) are high-level control schemes that have a long history in managing large and/or heterogeneous systems, such as electrical network grids, nuclear power plants, etc. In general, the supervisor implements a decision-making scheme based on the condition of the system [4,5]. However the underlying principles remains more or less the same; i.e. data acquisition, data analysis, data display, feedback control and data storage. Figure 1 describes the basic characteristics of sensor network operation.

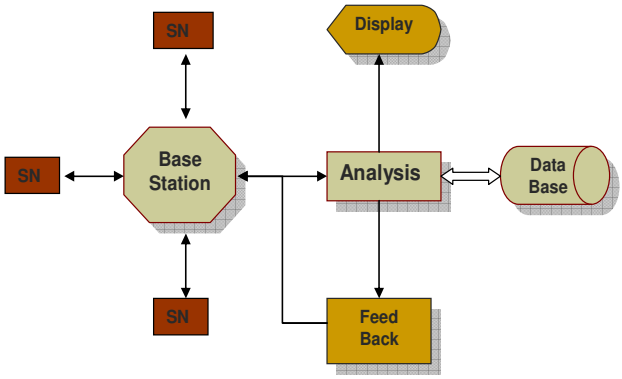

Figure 1: Operating principle of sensor networks

This paper is written with the perspective of traditional Supervisory Control applied to a WSN context. We describe an information-centric framework for the management of data and decisions in wireless sensor nets. The framework is based on a database and supervisory controller implemented using National Instruments' Labview®. The basic layout of this framework implementing a Data Logging and Supervisory Control Layer (DSC) is shown in Figure 2. The paper is organized as follows: in Section 3 we describe the target applications for the database; in Section 4 we describe the discrete event controller (DEC); in Sections 5 and 6 we describe a data-logging and supervisory controller (DSC) for WSN and a number of implementation challenges addressed by our framework; in Section 7 we describe show how the DSC was implemented on the WSN test-bed; finally, Section 8 concludes the paper.

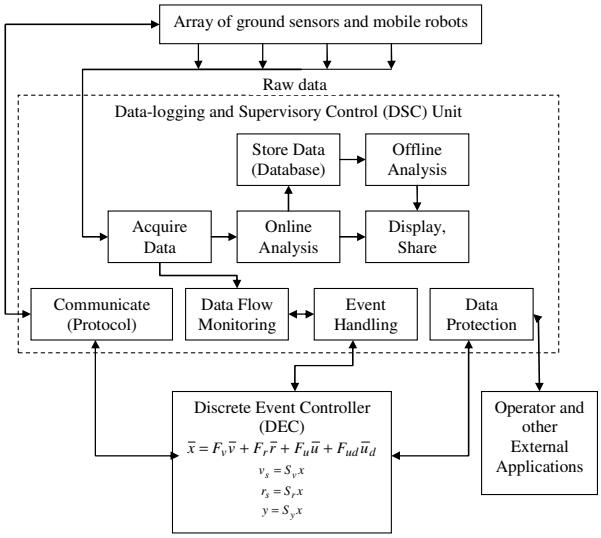

Figure 2: Data-logging and Supervisory Control (DSC) Layer for WSN.

3. TARGET APPLICATION

The proposed DSC Layer is currently being tested at the Distributed Intelligence and Autonomy Lab (DIAL) at UTA's Automation & Robotics Research Institute in two WSN scenarios:

3.1. Strain, vibration, and temperature wireless sensors for Condition Based Maintenance

We implemented a sensor network to facilitate real-time monitoring and extensive data processing for Heating & Air Conditioning Plant monitoring. In our implementation, we used the X-link wireless measurement system from Micro Strain, Inc. There are three types of sensor nodes– G-link (MEMS accelerometer - ±10G full scale range), SG-link (strain gauge), and V-link (supports any sensor generating voltage differences). Each node is complete wireless measurement systems with a Microchip PIC 16f877A microcontroller Nodes contain low power RF Monolithic transceiver using on-off keyed (OOK) modulation of 916MHz carrier frequency and providing transmission rate of 19.2 Kbps up to 30 m of range. Sensor nodes are multi-channel, with maximum of 8 sensors supported by a single wireless node. A single receiver (Base Station) addresses multiple nodes; maximum of 2^{16} nodes can be addressed as each node has 16-bit unique address. All nodes support a 9V external rechargeable battery.

3.2. Vibration, light, magnetism, temperature and color sensors in stationary unattended ground sensors (UGS), and mobile WSN nodes mounted on robots

The environment is being monitored by the UGS which send the surrounding parameter data to a central control unit. A fleet of 20 inexpensive mobile sensor network nodes (ARRI-Bots) is currently being built at ARRI's DIAL lab (Figure 3), as part of a larger testbed composed of CyberGuard SR2/ESP robots, Acroname Garcia robots and wireless Xbow MICA-2 and Cricket Motes. The inexpensive ARRI-Bots are equipped with wheel encoders for localization, and share this data through a Mote. Modulating the transmitter power makes it possible for two ARRI-Bots to go out of range if the distance between them is greater than a few feet, enabling the study of WSN with limited transmission range. The rovers are controlled by a Javelin Stamp CPU and are also equipped with a color sensor that can measure 256 RGB values on the lab floor.

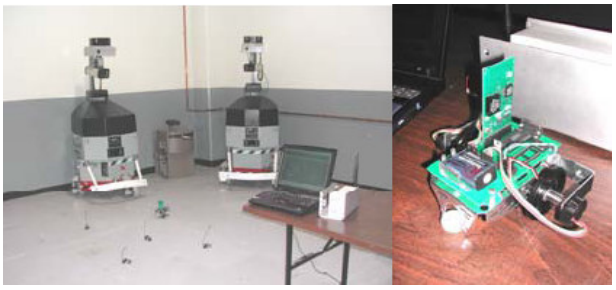

Figure 3: CyberGuard SR2/ESP, inexpensive Rovers, and MICA Xbow motes on the DIAL lab floor

4. DISCRETE EVENT CONTROLLER (DEC)

As discussed above, for condition based maintenance with sensor networks we require a control scheme which would monitor the state of the system and in the event of any unwanted change in state it would manipulate with the available resources to bring the system back to stability. Moreover the controller can also be designed to modify the system in order to behave in a certain way in case of the occurrence of a predefined event. The discrete event control (DEC) [9] discussed in this section is one such controller which is ideal for issue based event handling in wireless sensor networks and hence has been incorporated with this research work.

Mobile WSN comprise of multiple heterogeneous resources capable of performing diverse tasks such as measuring, manipulating, moving, sensing, etc. In mobile sensor networks, a strong one-to-many mapping between a resource and the tasks that the resource can perform occurs. This mapping can be statically assigned resulting in shared resources, or dynamically assigned resulting in both shared and routing resources. Shared resources arise when multiple tasks contend for a single shared resource, while routing resources arise when multiple resources contend to perform a single task. The use of shared or routing resources is a major problem occurring in discrete event systems, including in WSN.

The DEC is a rule based matrix controller [8] that has inner loops for non-shared resources and outer decision making loops for shared resources. Since the controller is based on matrices, it provides efficient mathematical framework for discrete event analysis and design according to linguistic *if-then* rules:

Rule i: If <conditions^i hold > *then* <consequences^i>

For coordination problems of multi-agent systems (e.g. a mobile wireless sensor network), we can write down a set of if-then rules to define the mission planning of the sensor agents, For example:

Rule i: If <sensor1 has completed task1, robot2 is available and a chemical alert is detected > *then* <robot 2 starts task4 and sensor 1 is released>

Let r be the vector of resources used in the system (*e.g. mobile robots and UGSs*), v the vector of tasks that the resources can perform (e.g. *go to a prescribed location*, *take a measurement*, *retrieve and deploy UGS*), u the vector of input events (*occurrence of sensor detection events, node failures, etc.*) and y the vector of completed missions (*outputs*). Finally, let x be the state logical vector of the rules of the DE controller. Then the controller state equation [8, 9] is given as:

Controller state: $\bar{x} = F_v \bar{v} + F_r \bar{r} + F_u \bar{u} + F_{ud} \bar{u}_d$

Operation or job start: $v_s = S_v x$

Resource release: $r_s = S_r x$

Job completion: $y = S_y x$

Where;

x is the task or state logical vector
F_v is the task sequencing matrix
F_r is the resource requirement matrix
F_u is the input matrix
F_{ud} is the conflict resolution matrix
U_d is the conflict resolution vector
S_v is the task start matrix
S_r is the resource release matrix
S_y is the output matrix

The task sequencing matrices (F_v and S_v) are written down from the required operational task sequencing, e.g. as generated by the Mission Commander or a computer

software planner. On the other hand, the resource requirements matrices (F_r, S_r) are written down based on the resources needed to perform the tasks and are assigned independently of the task sequencing matrices.

S_v is the task start matrix and has element *(i,j)* set to '1' if logic state x_j determines the activation of task *i*. S_r is the resource release matrix and has element *(i,j)* set to '1' if the activation of logic state x_j determines the release of resource *i*. S_y is the output matrix and has element *(i,j)* set to '1' if the activation of logic state x_j determines the completion of mission *i*. The task start equation computes which tasks are activated and may be started, the resource release equation computes which resources should be released (due to completed tasks) and the mission completion equation computes which missions have been successfully completed.

Vector v_s, whose `1' entries denote which tasks are to be started, and vector $r_s$, whose `1' entries denote which resources are to be released, represent the commands sent to the DE system by the controller. '1' entries in vector *y* denote which missions have been successfully completed.

Given the presence of shared resources, simultaneous activation of conflicting rules may arise. Matrix F_{ud} is used to resolve conflicts of shared resources, i.e. conflicts deriving by the simultaneous activation of rules which start different tasks requiring the same resource. Matrix F_{ud} has as many columns as the number of tasks performed by shared resources. Element *(i,j)* is set to '1' if completion of shared task *j* is an immediate prerequisite for the activation of logic state x_i. Then an entry of '1' in position *j* in the conflict resolution vector u_d, determines the inhibition of logic state x_i (rule *i* cannot be fired). It results that, depending on the way one selects the conflict-resolution strategy to generate vector u_d, different dispatching strategies can be selected to avoid resource conflicts or deadlocks.

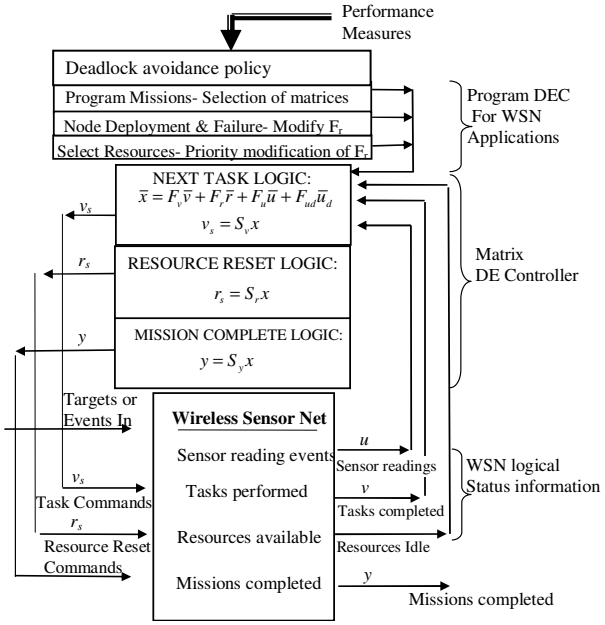

Figure 4: Discrete Event Controller Architecture [8, 9]

A *mission* is defined to be a prescribed sequence of tasks to be performed by the WSN based on observed sensor readings. Many missions may be programmed into a WSN, and several missions may execute simultaneously depending on sensor readings, node failures, or operator queries. Adaptability is the ability of an agent team to change its behavior according to the dynamical evolution of the environment. In our framework, adaptability occurs on two levels. On one level, the DEC allows for dynamic on-line assignment of tasks and resources based on sensor events, given fixed DEC matrices *Fv, Fr, Sv, Sr*. These matrices contain the task sequencing and resource assignment information for fixed prescribed missions. Therefore, on a second higher level, programming missions into the WSN amounts to appropriately defining the DEC matrices. On this higher level, adaptability requires modification of the DEC matrices as mission change, nodes fail, or additional nodes are deployed.

For multiple missions selection of the task sequencing matrices *Fv, Sv* and the resource assignment matrices *Fr, Sr* are done using computer science high-level planners such as HTN planners. For WSN, extensions are needed to this procedure since missions may change in real time based on sensor events, node failures, or human user desires. The overall complex environment monitoring operation is decomposed into multiple missions. Each mission has defined tasks and requires rules for task sequencing, resource dispatching and conflict resolution.

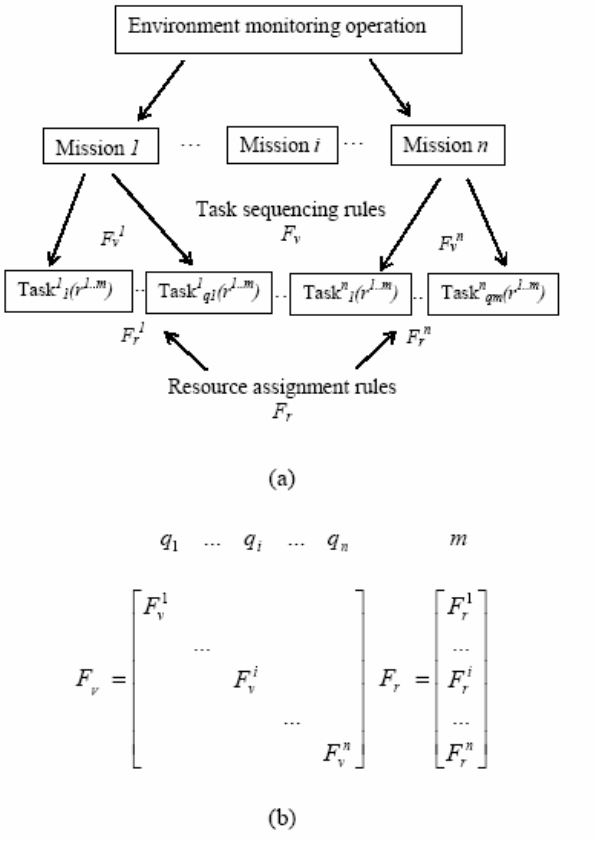

Figure 5: Multiple mission representation with DEC

Our DEC contains an intuitive matrix representation of this architecture and allows missions to be designed and

combined in a very direct manner using *block matrix* techniques. This amounts to *programming the WSN* to accomplish the desired missions. The following figure shows one example of how multiple missions are incorporated with DEC [18].

The discrete event controller is a mixture of centralized and decentralized control in a single WSN. In fact, decentralized algorithms appear in the DEC as compatibly partitioned block diagonal matrices *Fv, Fr, Sv, Sr*, for compatibility of the block matrix sizes allows one to decompose the DEC into several decentralized controllers that can be locally implemented. Coordination between decentralized controllers is revealed as entries outside the blocks that effectively couple the decentralized blocks together. This can be interpreted as a hierarchal control structure where a high-level supervisor coordinates the activities of the decentralized controllers.

5. WSN DATA LOGGING & SUPERVISORY CONTROL

There are many practical challenges [3] encountered in operating a large network of un-tethered sensors such as:

- *Challenges in Data Acquisition*

UGSs send data continuously to the central control unit. This data need to be processed, demodulated, identified and sorted-out before extracting any kind of substantial information from it.

- *Challenges in Data Storage & Retrieval*

Once the data has been processed, it must be stored. As storage space is limited, the data should be stored efficiently. Only the essential and significant part of the acquired data should be stored. Also the stored data should be readily available for decision making.

- *Challenges in Event Handling*

If the acquired data shows a critical event in the sensed environment, alarms with appropriate priority levels need to be generated. The alarms would warn the DEC in the central control unit about the emergency, which in turn can initiate a sequence of tasks to resolve the situation and bring the system back to normalcy.

- *Challenges in Network Control and Routing*

The network must allocate resources such as energy, bandwidth and node processing power that are dynamically changing & system should change configuration as required.

- *Challenges in Information Processing*

The nodes in an ad hoc sensor network collaborate to collect and process data to generate useful information [4]. Processing data from more sensors generally results in better performance but also requires more communication resources and thus energy. Similarly, less information is lost when communicating information at a lower level but require more bandwidth. Tradeoffs between performance & resource utilization are issues of consideration.

- *Challenges in Tasking Querying*

It is important that users have a simple interface to interactively task and query the sensor network. The users should be able to command access to information, e.g., operational priority and type of target, while hiding details about individual sensors. One challenge is to develop a language for querying and tasking, as well as a database that can be readily queried. Other challenges include finding efficient distributed mechanisms for query and task compilation and placement, data organization, and caching.

- *Challenges in Ad Hoc Network Discovery*

In a network each node needs to know the identity and location of its neighbors to support processing and collaboration. In the case of a mobile network, since the topology is always evolving, mechanisms should be provided for the different fixed and mobile sensors to discover each other. When self-location by GPS is not feasible or too expensive, other means of self-location, such as relative positioning algorithms, have to be provided.

- *Challenges in Security*

As the wireless sensor networks handle large amount of raw data containing secure information, this data must be protected from unauthorized access. A proper user authorization protocol must grant access privileges to handle the data, database or the discrete event controller itself.

To deal with the challenges mentioned above, the need for a data-centric supervisory control module is proposed in this paper. The basic architecture [10] of this supervisory control system is shown in Figure 6.

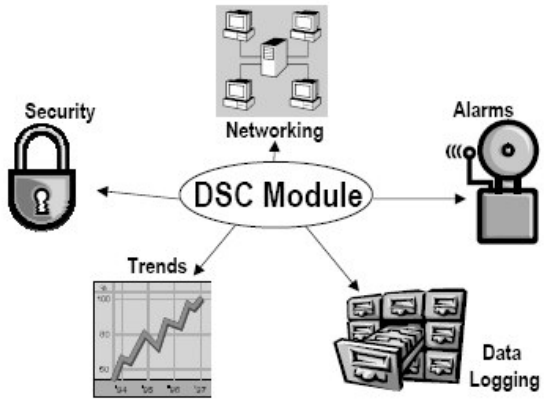

Figure 6: Application Structure of DSC

We have proposed a human machine interface and supervisory control and data acquisition (HMI/SCADA) framework which we refer to as the Data-logging and Supervisory Control (DSC) unit. The features of this DSC unit are listed as follows:

- Data-Logging
  - Data Acquisition
    - Sensor Deployment, A/D Conversion
    - Signal Connectivity
    - Signal Conditioning
      - Amplification, Attenuation
      - Sampling, Multiplexing
      - Filtering, Linearization
  - Online Analysis
    - Channel Scaling, Feedback Control
    - Alarming & Event Handling
  - Logging & Storage
    - Database Management
    - Dead-band Filtration
  - Offline Analysis
    - Statistics Computation

- Frequency Analysis
- Display, Sharing & Reporting
  - Historical Data Trending
  - Current Data Trending
- Supervisory controller
  - Connection Establishment, Dataflow rate monitoring, Criticality Identification
  - Data Protection, User Management, Data Export

To implement the above mentioned DSC with wireless sensor networks we have taken use of National Instrument's LabVIEW® as primary development tool as it is one of the highly acclaimed and widely used measurement and control application software, throughout the globe. Along with it we use the Citadel® database as the data processing system in our application. The advantage of using these software packages is that they are quick and flexible to design, they allow interfacing with all hardware the same way, simplifying development and maintenance of the system. Furthermore, as the hardware is kept abstract in the DSC module, we can add and remove features without affecting the hardware, and change the hardware implementation without affecting how we manage the data. Logging data to a database, preventing unauthorized users from accessing data, and sharing data across a network are made easy with these applications [10].

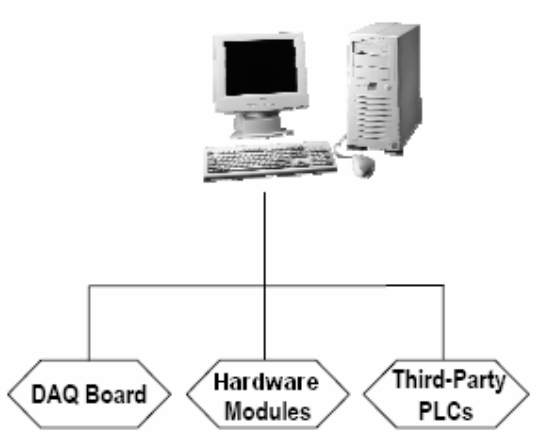

Figure 7: Hardware Portability of DSC

The block diagram in Figure 8 shows how the DSC module fits into the interfacing application between a universal software and different hardware profiles. The DSC Module can be seen as an OPC client. All hardware that communicates with the DSC Module must have an OPC server. OPC stands for OLE for Process Control, OLE stands for Object Linking and Embedding, and is a set of driver specifications that is actively developed and maintained by a network of individuals and corporations. The technology allows any piece of hardware to communicate with the computer in the same fashion, regardless of how it actually behaves. This flexibility allows a developer to be totally unaware of the intended hardware of a system, simply using OPC to communicate where needed. DSC takes this functionality and builds upon it to allow even greater flexibility through the use of tags. As an end-user, OPC means that we have interoperability with different hardware vendors. All of the hardware that supports OPC can be used from the DSC module OPC is set up as client-server architecture. The server's job is to talk to the hardware and provide data items to the clients. The server is a complete piece of software that is generally provided by the hardware manufacturer.

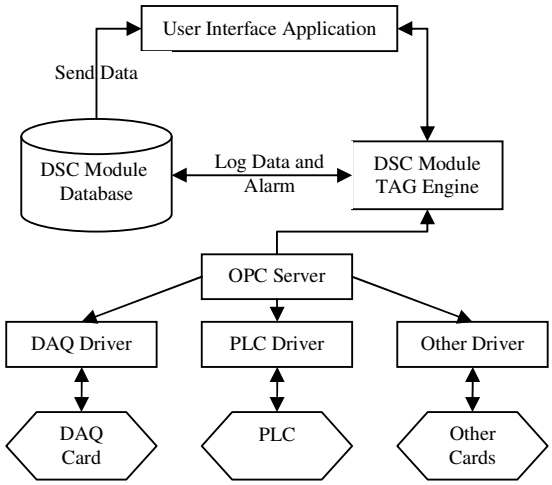

Figure 8: DSC functionality block diagram in application

6. DETAILED FUNCTIONALITY OF DSC

There are three main components of the data-logging and supervisory control unit [10]:

1. Server
2. Tag
3. Database

As discussed in the previous section, the server interfaces the hardware to the computer and maintains connection with it. Transfer of data and instruction is done between the computer and hardware through I/O channels. These I/O channels are configured and monitored by tags which are run by a tag engine that runs in the background. The tags that are configured in the tag engine get updated and read by the OPC Servers. The application then can read and write directly from the tag engine. The tag engine provides direct access to hardware values of multiple OPC Servers all in one place. There are four different ways to access a tag, namely:

- Input - only read values from the tag
- Output - only write values to the tag
- Input/Output - both read and write values to the tag
- Memory - An Input/Output tag that is not connected to any hardware device

Depending upon the functionality there can be four types of tags:

- Analog - Represent continuous values within a range
- Discrete - Represent two state values, that is, true/false or on/off or 0/1
- Bit Array - Multi-bit values up to 32 discrete values, represented by an array of Booleans

- String - ASCII or binary character representations of values. A tag not fitting the previous tag types could be processed as a string tag, for example, bar codes

Finally, the third major component of DSC is the database. First, the Tag Engine acquires the value of a tag and updates it. This value appears in the software interface and is also sent to the Citadel database. Citadel then determines the appropriate location in the database to store the data. Citadel can accumulate and store this data. These stored data points, often referred to as Historical Data, are available for use by other programs at any time. Citadel stores a variety of information in its database. Of course, for tags that are configured to log their values, Citadel will keep those in its database. Events are logged in the same manner. Events logged include any alarm state changes and any tag values changes made by the operator. Citadel does not store data at selected intervals; rather, Citadel stores data only when it changes. This allows Citadel to conserve disk space, as storing the value of every point of data that changes only once a day would be rather inefficient. Data is logged with a timestamp of when it changes. Logging data in this manner makes it easy to determine the value at any point in time. It is important to notice that data logging using is more event driven than time driven. This is evident in the manner in which data is logged based on significant change.

6.1 EFFICIENCY IN DATA MANAGEMENT

When logging data continuously for a long period of time to the Citadel database, it becomes necessary to organize the recorded data. If data is recorded to one database, it may get large in size, making it harder to manage and back up. Storing in a different database periodically or using the archiving tools allows backing up the databases more easily. Also, using one database to store data for a smaller amount of time allows locating data faster, instead of having to parse through one big database that contains all the data. Some methods of managing databases that have been incorporated with our application include the following: using data sets, managing database size, backing up or archiving databases, merging databases, dead-band, and so on.

6.2 DEAD-BAND

The Citadel database logs data only when there is significant change in value and thus optimizes the database storage space. This change in value can be defined by setting up dead-bands in the tag engine. A dead-band is a region where values can change, but it does not update the output value. There are four places in the DSC Module where Dead-banding is used:

- I/O Group Dead-band (% of range)
- Update Dead-band (% of range)
- Alarm Dead-band (% of range)
- Log Dead-band (% of range)

Each is important in eliminating unnecessary data processing and data logging.

6.2.1 I/O Group Dead-band

This dead-band is applied to the OPC Server itself. I/O group dead-band is a percentage of the total range defined by the OPC Server. When set, the OPC Server does not send an interrupt to the Tag Engine to update a value unless the value change is greater than the dead-band. The I/O Group Dead-band is a part of the OPC Server subscription that is defined in the I/O Group. Like other settings in the I/O Group, the I/O group dead-band is a request that is sent to the OPC Server. The OPC Server can ignore the I/O group dead-band.

6.2.2 Update Dead-band

This dead-band is similar to the I/O Group Dead-band, but it is applied to the tag engine. The update dead-band is a percentage of the engineering scale. When this is set, the tag engine ignores updates from the OPC Server if the value change is not greater than the dead-band.

6.2.3 Alarm Dead-band

This dead-band is also a percentage of the engineering scale. The alarm dead-band acts like a hysteresis in that the tag does not return to a normal state until it has left the alarm condition by at least Alarm Dead-band. For example, if the range is 0-100 and the Alarm Dead-band is 1% and you have a HI Alarm set at 75, if the tag value goes above 75, the tag is in alarm. After the tag is in alarm, the value must drop below 74 to go out of alarm. This eliminates the problem of causing multiple alarms, while a noisy signal hovers about the alarm value.

6.2.4 Log Dead-band

This dead-band allows users to limit the data that is being logged to the database. To get the most efficient use of hard drive space, the Citadel database only logs data on change. If a log dead-band is set, then Citadel only logs data if it changes by more than the dead-band.

6.3 SCALING

The scaling converts the raw data read from the server into engineering units suitable for operators. Engineering units are standard measurement units such as ^{O}C, ^{O}F, Meter, Volt, Ampere, Joule etc. The following lists types of analog scaling:

- Linear: eng unit = m(raw measurement) + b, where b = eng min, m = (eng max-eng min)/(raw max-raw min)
- Square root: eng unit = b + m * sqrt(raw measurement), where b = eng min, m = (eng max-eng min)/sqrt(raw max-raw min)

The raw scale is the range of values read from the server. If the server already performs scaling, then additional scaling might not be necessary. The engineering scale determines the range of values used by the Tag Engine and user application.

6.4 SECURITY

There are two areas of security in the DSC Module. The first area is the software environment itself. Here access can be restricted the user interface to certain users or groups of users. The second area is in the tag configuration editor. Security can be implemented so that certain users cannot change the configuration file by adding/removing tags or

changing any of the other settings in the configuration editor. Network security can also be implemented to restrict tag access to specified computers or IP addresses.

The diagram in Figure 9 shows the overall functional block diagram of the data-logging and supervisory control (DSC) system for wireless sensor networks.

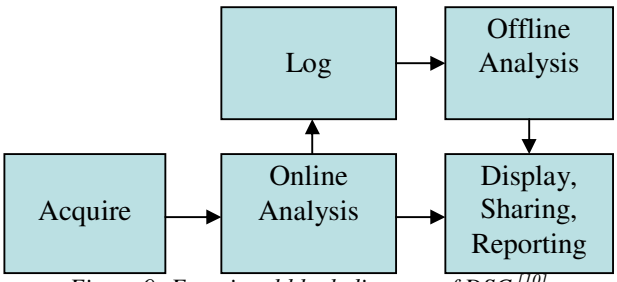

Figure 9: Functional block diagram of DSC [10]

7. PROGRAMMING OF THE DSC AND DEC

Sensor data management involves almost all the features of supervisory control system, such as connecting to various data sources, monitoring and processing the raw data, storing the extracted data, alarm & event management, trending and report generation, data protection, data exportation etc. The wireless sensor network implements the discrete event controller (DEC) to generate a rule-based task system to automate a "smart" environment. The DSC scans the raw data and prepares a useful set of data and stores it in a database. This data then serves the purpose of informing the environment information to the DEC from which the DEC determines the state and resource requirement. Whenever a critical situation arises, the supervisory control system (DSC) raises an alarm indicating the DEC that an event needs attention. In turn, the DEC prepares a series of tasks that need to be carried out in order to deal with the situation. The DEC mission could be, for instance, that a mobile robot is sent to the alarm UGS to take a backup sensor measurement. Once the mission has been accomplished successfully the DSC disables the alarm, and the event is logged for future reference. The DSC adds networking features to the DEC enhancing its capability to run across a network. Hence the DEC can be run remotely from a secure location or one with advanced facilities. For instance, if the wireless sensor network is deployed in the field, DSC can allow the DEC to run from a remote saving appreciable amount of operating costs and reducing the risk factors. Finally, data security requires restricting control of the DEC from unauthorized users. The DSC sets up security levels of security such as administrator, operator, guest etc. The DSC also categorizes the read/write/modify privileges for data.

As mentioned earlier, the implementation of DSC software for WSN has been built and tested at ARRI's DIAL Lab using National Instrument's LabVIEW®. The DSC Module provides solutions for supervisory control of a wide variety of distributed systems and data-logging schemes using the flexibility of graphical LabVIEW® programming. The LabVIEW® Data-logging and Supervisory Control (DSC) Module uses the National Instruments Citadel® historical database. The DSC Module also includes the Citadel® ODBC driver that has commands to perform data transforms.

The data-logging and supervisory control unit of the wireless sensor network uses a VI (virtual interface) based server as communication channel. The server supplies data points from several input items to the tag engine as these points are read. The server automatically timestamps values as they are acquired from items. After the creation and registration of the VI-based server, a tag is created to maintain connection with the I/O points. In this configuration process attributes for the tag were set. These mainly include scaling, connections, alarms and dead-bands. A dead-band filter has been used to extract useful data. Any changes in value from a data point are compared to the previous value. Only if the difference between the new value and the previous value exceeds the dead-band does the new value replace the old. The DSC ignores an operation if the change in data is not considered significant. By increasing the dead-band size, the strain on the DSC can be reduced, though this might compromise data resolution.

Security in the application has been implemented by setting up of user and group accounts. The DSC has a User Account Manager which creates and edits the properties of groups, user accounts, assigns users to groups and manages security accounts for applications.

Another important application of WSN DSC module is the publishing of the control console. The DSC console can be accessed and controlled from the web. This provides web access to sensory information from all WSN nodes. The data from the historical database can be exported to html file. This enhances the performance of the DEC by broadening the spectrum of operation and adding more versatility to the control system. The interactive user interface designed for monitoring and controlling the wireless sensor network is shown in Figure 10.

Figure 10: DSC User Interface for WSN

In a monitoring application, this interface was used to monitor the WSN comprising of several UGS and several Cyber-Motions mobile sentry robots, as shown in Figure 11. Sensor readings are collected and scaled to proper engineering values. The X-bow motes have onboard sensors for measuring light, temperature, sound & vibration and magnetism. The values of these parameters represent the network status. Along with these sensors the mobile robots have their own onboard sensor package which include:

ultrasonic intrusion detector, voice channel, optimal flame detector, auxiliary gas sensor, dual passive infrared oxygen sensor, microwave intrusion radar, time lapse VTR, smoke sensor, inventory tag reader, gas sensor infrared illuminator, temperature sensor, optical pyrometer, humidity sensor, zoom lens, ambient light sensor, video level, measurement system, and video transmitter.

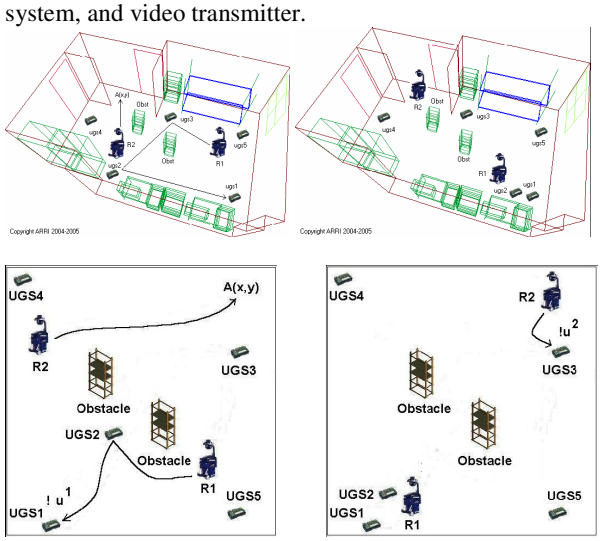

Figure 11: Diagram of a Wireless Sensor Network (WSN) for warehouse monitoring

The data-logging module constantly monitors these widely varied data and depending upon the dead-band set by the user, logs the data into the Citadel database. The DEC is fed the task based rule for any particular mission, such as sample mission sequences and the issue based rule table [18] shown in Tables 1-2.

The DEC works in tandem with the DSC. Upon occurrence of an event the DSC inform the DEC about it. Then the DEC according to the rule-base predefined for the particular sequence carries out the tasks. A snapshot of the software application which control the Discrete Event Controller (DEC) [18] is shown in Figures 12 and 13.

Table 1: Sample Mission 1 Task Sequence

mission1	notation	description
Input 1	u^1	*UGS1 launches chemical alert*
Task 1	$S4m^1$	*UGS4 takes measurement*
Task 2	$S5m^1$	*UGS5 takes measurement*
Task 3	$R1gS2^1$	*R1 goes to UGS2*
Task 4	$R2gA^1$	*R2 goes to location A*
Task 5	$R1rS2^1$	*R1 retrieves UGS2*
Task 6	$R1lis^1$	*R1 listens for interrupts*
Task 7	$R1gS1^1$	*R1 gores to UGS1*
Task 8	$R2m^1$	*R2 takes measurement*
Task 9	$R1dS2^1$	*R1 deploys UGS2*
Task 10	$R1m^1$	*R1 takes measurement*
Task 11	$S2m^1$	*S2 takes measurement*
output	y^1	*Mission 1 completed*

Table 2: DEC Rule-Base for Mission 1

Mission1-operation sequence	
Rule1 x_1^1	If u^1 *occurs and S4 available and S5available then start* $S4m^1$ *and start* $S5m^1$
Rule2 x_2^1	If $S4m^1$ *and* $S5m^1$ *completed* and *R1 and R2 available* then *start* $R1gS2^1$ *and* $R2gA^1$ and *release S4 and S5*
Rule3 x_3^1	If $R1gs2^1$ *and* $R2gA^1$ *competed* then *start* $R1rS2^1$ and *release R2*
Rule4 x_4^1	If $R1rS2^1$ *completed* then *start* $R1lis^1$ and *release R1*
Rule5 x_5^1	If $R1lis^1$ *completed* and *R1 and R2 available* then *start* $R2m^1$ *and* $R1gS1^1$
Rule6 x_6^1	If $R1gS1^1$ and $R2m^1$ *completed* then *start* $R1dS2^1$ and *release R2*
Rule7 x_7^1	If $R1dS2^1$ *completed* then *start* $R1m^1$
Rule8 x_8^1	If $R1m^1$ *completed* and *S2 available* then *start* $S2m^1$ and *release R1*
Rule9 x_9^1	If $S2m^1$ *completed* then *release S2* and *terminate mission1* y^1

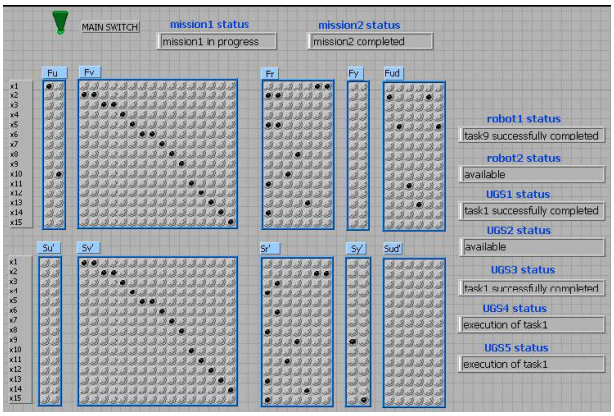

Figure 12: Application for Discrete Event Controller (DEC)

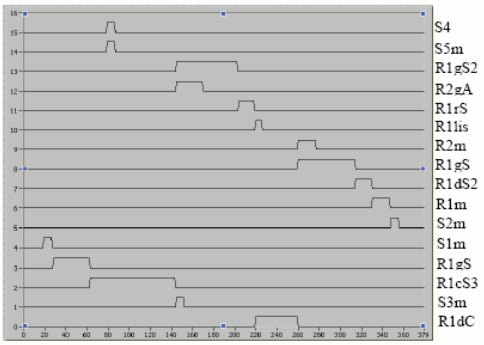

Figure 13: Time trace simulation result of DEC operation

In the process of coordinating the mission, the DSC collects network status data which plays vital role in characterizing the network performance through various online and offline analysis. Both historical and current data trends are important in deriving conclusions about the network. The DSC made this possible by proving the data sets required for preparing these trends and reports. A sample current, historical data trend report and data-set are shown in Figures 14 and 15.

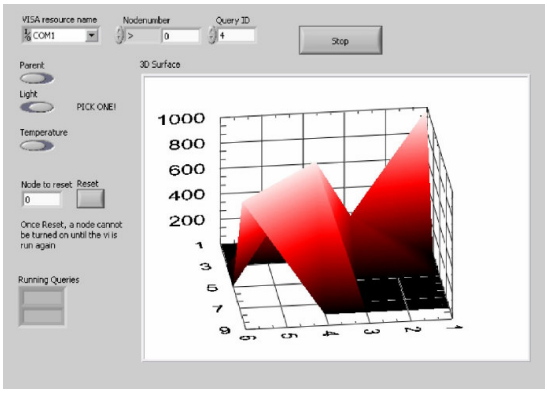

Figure 14: Current Data Trend on Temperature Profile

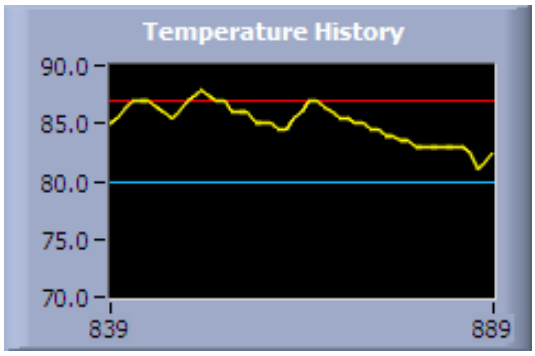

Figure 15: Historical Data Trend on Temperature Profile

Apart from designing applications for preparing data trends, modules have also been designed for efficient data acquisition and data processing. For example, the DSC acquired the data from multiple X-bow motes and used running-average technique to ensure accuracy in readings. Offline analysis is performing mathematical functions on data after it has been acquired in order to extract important information. Types of offline analysis include computing basic statistics of measured parameters, as well as more advanced functions. Advanced analysis applications such as Kurtosis, Bollinger band analysis, Kalman Filter, frequency and transient analysis, waterfall display etc are also developed and augmented to the main DSC module to strengthen the capability and scope of data interpretation as depicted in Figures 16-19.

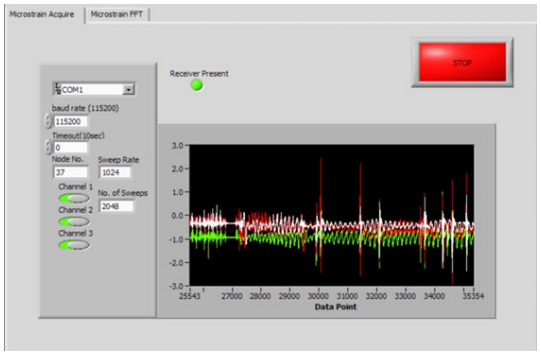

Figure 16: Vibration data from Micro-strain sensors in x, y & z (amplitude vs. time). The spikes indicate tapping

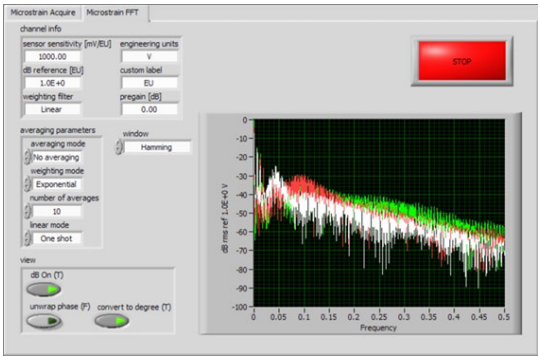

Figure 17: FFT of vibration data

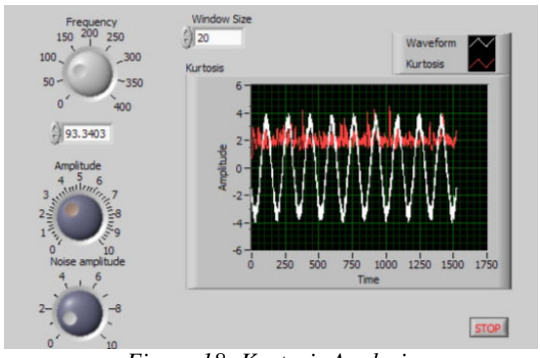

Figure 18: Kurtosis Analysis

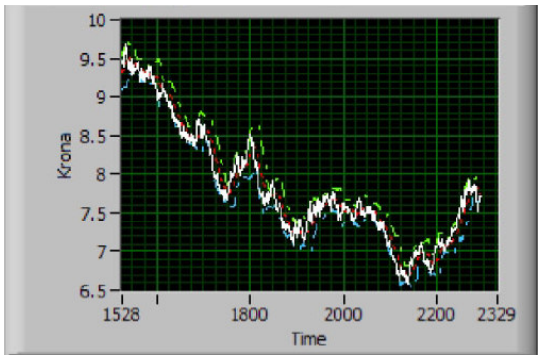

Figure 19: Bollinger Band Analysis

Finally, NI-LabVIEW® offers CAN interfaces on a variety of platforms, including PCI, PXI, PCMCIA, and National Instruments CompactRIO. NI interfaces are available with high-speed, low-speed/fault-tolerant, single-wire, and software-selectable physical layers. This feature is very useful if the WSN application is co-located with traditional wired networks for industrial monitoring.

8. CONCLUSION

This paper includes a theoretical framework and practical implementation of various data acquisition and data-logging techniques, historical and current data trending, issue-based event handling and security for wireless sensor networks. By using the LabVIEW® programming interface we enable robust and cross platform monitoring, programming, and web-based maintenance of the WSN. Many advanced analysis modules working on the spot and in tandem with the main application reduce programming complexity and enhances system

performance. DSC adds more strength and flexibility to WSN by allowing the network to interface to a mission-aware Discrete Event Controller (DEC) application. Future work includes studying the scalability, robustness and reconfiguration ability of the DSC/DEC framework to address different WSN scenarios, and distribution of more computation to the sensor nodes.

ACKNOWLEDGEMENT

This work has been supported by NSF GRANTS IIS-0326505 and CNS-0421282, ARO grant M-47928-CI- RIP-05075-1, and by National Instruments Lead User Grant. The authors would also like to thank V. Giordano and S. Gorthi for their contributions to the implementation and testing of the DSC.

REFERENCES

[1] Barbosa M., Farsi M., Ratcliff K., "An overview of controller area network", Computing & Control Engineering Journal, Volume: 10, Issue: 3, Page(s): 113-120, Aug 1999

[2] Mayer K., Taylor K., "TinyDB by remote", World Conf. On Integrated Design and Process Tech., Austin, Texas, 3-6 Dec 2003

[3] Chong C., Kumar S.P., "Senor Networks: Evolution, Opportunities and Challenges", Proceedings of the IEEE, VOL. 91, No. 8 Aug 2003

[4] Cook D., Harris B., Lewis F., "Machine planning for manufacturing: dynamic resource allocation and on-line supervisory control", Journal of Intelligent Manufacturing, p. 413-430, vol. 9, 1998

[5] Giordano V., Lewis F., Turchiano B., Ballal P., Zhang J. B., "Supervisory control of mobile sensor networks: discussion and implementation", IEEE Trans. Systems, Man, Cybernetics, 2006.

[6] Krumm J., Shafer S., Wilson A., "Ubiquitous Computing Group Microsoft Research Microsoft Corporation: How a Smart Environment Can Use Perception", UBICOMP 2001 Workshop on Perception for Ubiquitous Computing, Oct 2001

[7] Lewis F., "Wireless sensor networks: Smart environments-technologies, protocols, and applications", ed. D. J. Cook and S. K. Das, John Wiley, New York, 2004.

[8] Lewis F., Mireles J., "Intelligent material handling: development and implementation of a matrix-based discrete event controller" IEEE Transactions on Industrial Electronics, vol. 48, Issue: 6, Dec. 2001

[9] Lewis F., Tacconi D., "A new matrix model for discrete event systems: application to simulation", IEEE Transactions on Industrial Informatics, Volume: 1, Issue: 1, Page(s) 39-46, Feb 2005

[10] National Instruments, LabVIEW® Data-logging and Supervisory Control module developer's manual, run-time manual and VI-based Server Development toolkit manual

[11] Rahimi M., Sibley G., Sukhatme G., "Robomote: a tiny mobile platform for large scale ad-hoc sensor networks", Proceedings of International Conference on Robotics and Automation (ICRA '02), vol. 2, Page(s) 1143-1148, May 2002

[12] Shin K.G., Zuberi M., "Design and implementation of efficient message scheduling for controller area network", IEEE transactions on computers, vol 49, No. 2, February 2000

[13] Tilak S., Abu-Ghazaleh N., Heinzelman W., "A taxonomy of wireless micro-sensor network models," ACM Mobile Computing and Communications Review, Vol. 6, No. 2, 2002

[14] Tiwari A., Lewis F., "Wireless Sensor Networks for Machine Condition Based Maintenance", Proceedings of International Conference on Control, Automation, Robotics, and Vision, Page(s) 461-467, Kunming, China, Dec 2004

[15] Tiwari A., Ballal P., Lewis F., "Energy-Efficient Wireless Sensor Network Design & Implementation for Condition Based Maintenance", Submitted to ACM Trans. on Sensor Networks, Aug 2005

[16] Goldin D., "Faster In-Network Evaluation of Spatial Aggregation in Sensor Networks", International Conference on Data Engineering (ICDE 2006), Atlanta, GA, April 2006

[17] Xia P., Chrysanthis P.K., Labrinidis A., "Similarity-Aware Query Processing in Sensor Networks" 14th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS'06), Island of Rhodes, Greece, on April 25-26, 2006

[18] V. Giordano, P. Ballal, F.L. Lewis, B. Turchiano, J.B. Zhang, "Supervisory control of mobile sensor networks: math formulation, simulation, implementation," IEEE Trans. Systems, Man, Cybernetics Part B, 2006