Real-Time Monitoring of Hurricane Winds using Wireless and Sensor Technology

Carlos E. Otero
Florida Institute of Technology/Electrical and Computer Engineering, Melbourne, FL, USA
Email: cotero@fit.edu

Antonio Velazquez, Ivica Kostanic, Chelakara Subramanian, Jean-Paul Pinelli and Larry Buist
Florida Institute of Technology, Melbourne, FL, USA
Email: {avelazqu, kostanic, subraman, pinelli, lbuist }@fit.edu

Abstract— **This paper presents a distributed software system for a wireless sensor network application that remotely monitors the effects of hurricane winds on man-made structures. The software system is divided into three independent segments that are distributed across the Internet to provide real-time collection and transmission of data between wireless remote sensor units and a centralized server. The software system uses a custom-designed communications architecture that is built upon existing wireless networking standards (IEEE 802.11 and HSPA) and that benefit from capabilities of Microsoft .NET development framework. By segmenting the software and separating application-specific code from the communications architecture, the software can be reused and applied towards a wide variety of wireless sensor networks operating in harsh environments. The system is currently under test and will be deployed for the 2009 hurricane season.**

Index Terms— **Wireless sensor networks, hurricane monitoring, architecture, structure health monitoring**

I. INTRODUCTION

Hurricanes have historically posed serious threats to man made structures. In 1992, hurricane Andrew destroyed approximately 25,524 homes and damaged another 101,241, combining for estimated damages of $25 billion [1]. In the 2005 hurricane season alone, hurricanes Dennis, Katrina, Rita, and Wilma caused extensive roof damages along their path and combined for a whopping 32.83 billion dollars in damages [2] [3] [4]. These large financial figures point to an inadequacy of the existing building codes and accentuate the need for improving these codes to avoid future disasters. As suggested in [5], the implementation of affordable solutions that mitigates potential damage requires an accurate characterization of the wind forces causing the destruction, and development of appropriate theoretical models that relate the storm forces and capacity of man made structures to resist them. Therefore, it is necessary to develop systems capable of measuring real-time wind pressures and wind speeds as they batter the structures under study. Furthermore, these systems must be autonomous and capable of operating remotely while reporting data to a central location, away from the hurricane, where data can be analyzed safely.

This paper describes a distributed software system designed to remotely monitor hurricane winds using wireless sensor technology. Specifically, it serves as blueprint to give direction to researchers on the design and implementation of similar software systems that require remote collection of wireless sensor data in harsh environments. The system is a second generation design based on the work reported in [6] [7] and is made of custom designed wireless sensor nodes mounted on residential houses that continuously record hurricane wind data. The software system coordinates data collection and transmits to a remote server, where reports are generated and made available through a convenient web interface.

The remainder of the paper is organized as follow. Section II provides an overview of the system for hurricane characterization. Section III describes the software requirements. Section IV describes the software and provides design and architecture details employed in both the real-time and application software. Specifically, it provides the design diagrams (e.g., state diagram, sequence diagram, etc.) created to develop the software system. Section V provides performance analysis for the software system. Section VI and VII describe the benefits obtained by the design employed by the software system and identify several avenues for future research to extend the efficiency and usefulness of the software system. Finally, Section VIII summarizes the paper and provides concluding remarks.

II. SYSTEM DESCRIPTION

A diagram of the hurricane characterization system [5] is presented in Figure 1. As seen, the system consists of remote sensor units installed on the structure under study and communicating with an associated base unit. The base unit communicates with the field laptop, which connects individual house installations to a central server.

Manuscript received June 25, 2008; revised November 19, 2008; Accepted January 17, 2008.

doi:10.4304/jcp.4.12.1275-1285

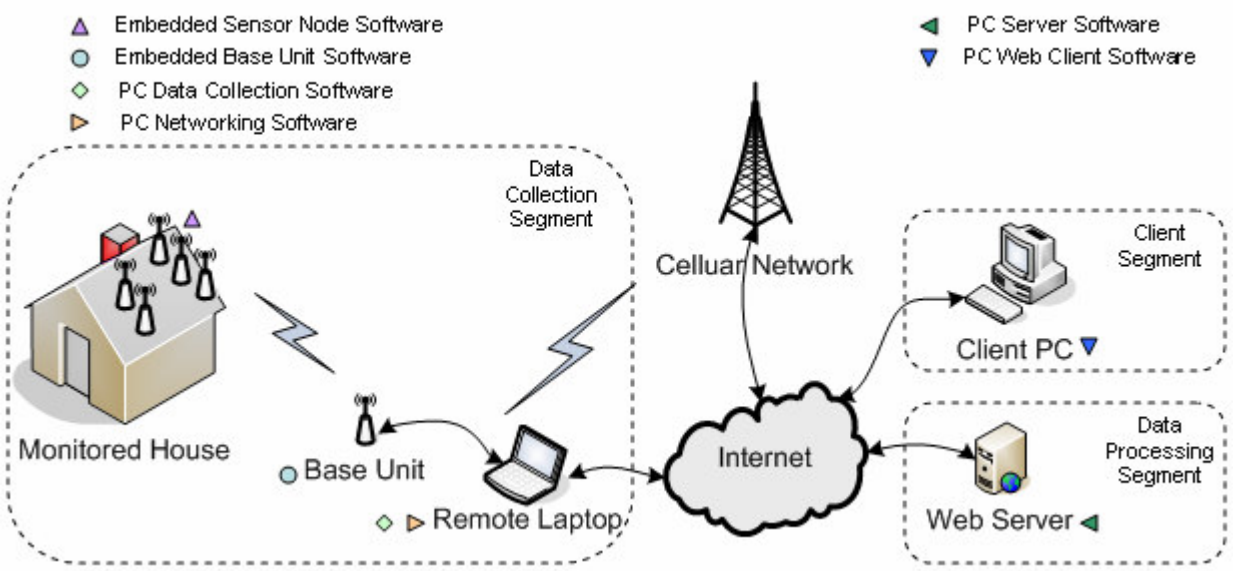

Figure 1 – System Description

Finally, client computers access the central server using the Internet to view the parameters associated with wind characterization.

The house installation may be linked to the server in two ways. The first one is using IEEE 802.11 (WiFi) connection. If the monitored house is equipped with the wireless Internet, the laptop connected to the base station utilizes the wireless Internet connection as a link to the central server. Alternatively, for house installations where there is no availability of 802.11 networks, the connection is provided through a cellular data network using an UMTS/HSPA/HSUPA card [8]. This card provides theoretical uplink (i.e. from base unit towards server) data throughput of 5.76 Mbps. However, in practical scenarios, the achievable throughput is limited by coverage, signal quality and the overall data traffic demand placed on the cellular network in the deployment area. In most cases, a throughput of several hundred Kbps to 1.5 Mbps is easily achieved. Two wireless networks provide redundant connections. By default, the sensor network is connected using the 802.11 interface. However, in the case of power outage on the house or in the case when the 802.11 connection is not available, HSUPA provides a seamless alternative to the 802.11 connection.

III. System Requirements

The system requirements for the hurricane characterization system consists of both functional and performance requirements. Functional requirements dictate the basic capabilities that the system must provide, while performance requirements dictate the efficiency level (e.g., response times, throughput) at which the system must operate to provide the required capabilities.

A. Functional Requirements

Given the application settings, the system needs to satisfy the following functional requirements:

- Support of up to 30 sensor units per house installation
- Support for multiple house installations operating simultaneously
- Capability of extended deployments (more than 96 hours) in a harsh environment of hurricane storms
- Collection of measurements required for characterization of hurricane winds including wind pressure, wind speed, wind direction and temperature
- Report system's health status including remote unit signal reception quality (RSSI) and battery charge
- Reliable local storage of measured data
- Reliable transfer of data to a central server in near real-time
- Near real-time data processing and presentation of measured data
- Near real-time control of deployed sensors

B. Performance Requirements

The main performance requirements imposed on the system ensure acceptable levels of the system's collection sampling rate and transmission efficiency. The system shall provide the following:

- Sampling rate of at least 10 samples/sec for pressure measurements
- Transmissions rate of at least 5MB of collected data to the remote site every 5-minute interval

IV. Software Description

The software system is composed of three segments, namely: *Collection Segment*, *Data Processing Segment*, and *Client Segment* [9]. The *Collection Segment* is composed of the software that operates on the remote

laptop, base unit, and remote sensor units. Upon starting the collection laptop and base unit, the data collection software recognizes all deployed sensors and establishes the internet connection to the central server. As the data is collected, it is compressed and forwarded to the *Data Processing Segment*. The software at the *Data Processing Segment* is designed to decompress, process, and analyze measured data. Upon analysis, it creates HTML files (for presentation purposes) containing the data and provides them to the *Client Segment*. Therefore, when operators are interested in obtaining near real-time status from a house, they simply log on to the remote server from any client computer via a web browser. Once logged on, the remote server on the *Data Processing Segment* will provide its status data every 5 minutes to the *Client Segment*. Alternatively, advanced users can also log into the remote laptop directly to modify the settings associated with the collection process.

A. Collection Segment

The collection segment is subdivided into two computer software configuration items, namely: Remote and Base Unit, and Field Laptop. The remote sensor units are mounted on the structure under study and are tasked with monitoring hurricane winds. The base unit resides near the perimeter of the house in a secured storage box together with the remote laptop. Together, the base unit and remote laptop manage data collection and transmission to the Data Processing Segment.

1) Remote and Base Unit

The remote sensor units contain custom hardware hosted by a protective metal casing, as seen in Figure 2. The protective casing is designed carefully so that the electronics are shielded from the elements [5].

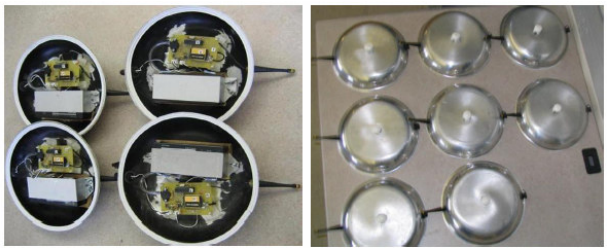

Figure 2 – Remote Sensor Unit

The hardware hosts real-time embedded software designed as a state machine to properly execute built-in tests, data collection, and transmission. When power is applied to the remote sensor units, the embedded software transitions to the Initialization State, where hardware and software tests are performed to ensure proper operation of the remote sensor unit. Once initialization tests are complete, the remote sensor unit software waits for direction from the base unit to report test results. The base unit manages network data transmissions using a scheduling algorithm that assigns internal transmission slots to remote sensor units. The scheduling algorithm uses these time slots to request data from the specific sensor unit associated with the active time slot. When the base unit requests data, the remote sensor unit software transitions to the Reporting State, where it transmits data wirelessly to the base unit using a half-duplex communication link. Upon successful data transmission, the remote sensor unit software transitions to the Operational State, where data collection using the onboard sensors begins. Data collection continues until commanded by the base unit to enter the Reporting State, where another transmission cycle occurs. This process is continuous and implemented in all remote sensor units. Figure 3 shows the state diagram for the remote sensor units.

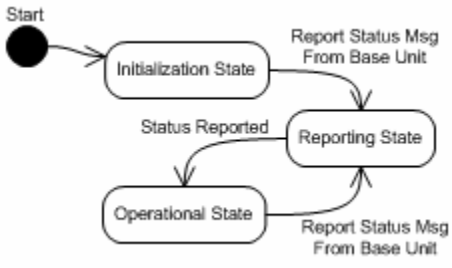

Figure 3 – Remote Sensor Unit State Diagram

2) Field Laptop

The field laptop hosts the hardware and software that manages the wireless sensor network. The minimum hardware requirements for the laptop are: (1) mass data storage, (2) serial (i.e., DB-9) transmission port, (3) PS/2 port, (4) 802.11 wireless card, and (5) HSPA cellular network card. The minimum software requirements for the field laptop are: (1) Windows XP, (2) Microsoft .NET Framework 2.0, (3) Custom Data Collection Software, and (4) Custom Networking Software [7].

a) Data Collection Software

The Data Collection Software orchestrates data collection by interfacing to the base unit using a standard DB-9 serial port and a PS/2 port (for the power supply). Once the sensor network is operational, the Data Collection Software receives wind characterization data from the base unit and deposits it in an outgoing directory monitored by the Custom Networking Software. Typically five minutes of data collection is deposited as a single measured data file. This process is illustrated in Figure 4.

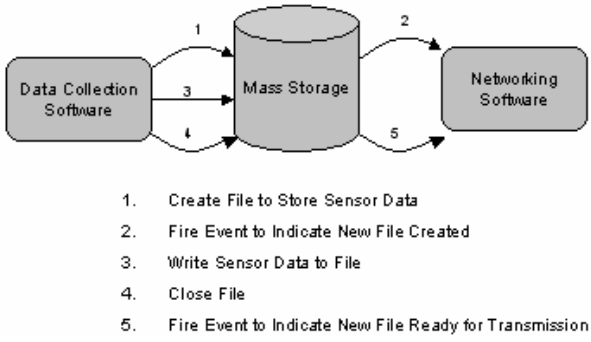

Figure 4 – Data Collection Software & Networking Software Interface

As seen, the files created in the mass storage become a critical section where both the Data Collection Software and Networking Software must share. Therefore, the software is configured to allow only one process to acquire the data file at any given time.

The files created by the Data Collection Software require specific information to allow the Data Processing Segment to identify status and data collected by specific remote sensor units. Therefore, a specific protocol is devised to allow compatibility between the collection segment and the data processing segment. Figure 5 displays the data file format, as specified by the protocol.

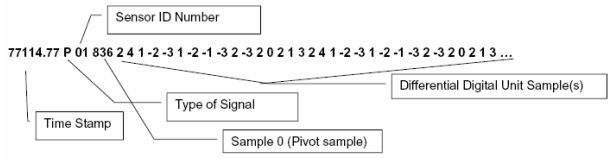

Figure 5 – File Data Format

As seen, the file data contains the following ASCII fields: time stamp, sensor id, type of signal, pivot sample, and differential digital unit samples, all separated by the space character. The Time Stamp field is a coded value that represents the collection time with resolution of a centesimal of a second with respect to midnight (0:00 hrs). For example, a collection time of 21:25:14:46 (i.e., 9:25 p.m. with 14 seconds and 46/100 centesimal of a second) is coded as follow:

$$21(60\,\text{min}\times 60\,\text{sec})+25(60\,\text{sec})+14\,\text{sec}+\frac{46}{100}\text{sec}=77114.46$$

Passing midnight twice causes the time stamp to wrap around and start over from 0:00 hrs, however since the file name contains the creation date, this is not an issue. The Type of Signal field contains information about the specific status reported by the remote sensor units. The following measurements are performed:

- P = Pressure
- S = Wind speed
- T = Temperature
- R = RSSI (Signal quality)
- B = Battery charge
- D = Wind direction

The Sensor ID Number field provides identification for every remote sensor unit. The actual system contains 30 remote sensor units, each assigned an Id from 1 to 30. Finally, the Pivot Sample field and Differential Digital Unit Sample fields contain the actual sampling of the signal (i.e., Pressure, Temperature, RSSI, Wind Direction, Wind Speed, or Battery) in coded format. The Pivot Signal field corresponds to the real value of the signal, which usually results in a relatively large number. The Differential Digital Unit Sample fields contain step-incremental of the signal with respect to the Pivot Signal field value. Once the data is received at the data processing segment, it is decoded and translated into proper units, such as millibars, miles per hour, degrees, etc.

The Data Collection Software is developed using the Visual Basic .NET language, which is natively supported by MS Windows operating system. Figure 6 presents the Graphical User Interface (GUI) of the Data Collection Software.

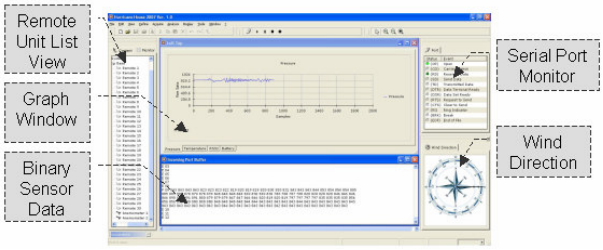

Figure 6 - Data Collection Software GUI

b) *Collection Networking Software*

The Networking Software is a separate software module that provides the necessary communications infrastructure between remote sensor units and the Data Processing Segment. The architecture, which is described extensively in [9], has been extended to include the Global Messaging layer. An overview of the extended networking software's architecture for the hurricane monitoring system is presented in Figure 7.

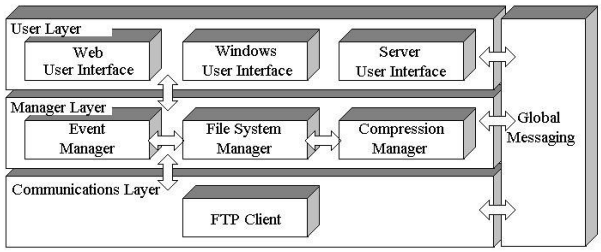

Figure 7 – Networking Software Architecture

The User Layer is responsible for all aspects of the visualization of the data in the system. This layer is necessary to separate the processing and communication of the system from the visuals. The manager layer is where all logic and processing are orchestrated. The main functions managed at this layer are compression, events, and file system common tasks. The manager layer delegates all visualization aspects to the user layer through the observer design pattern [10], and also delegates all details of communication to the communication layer. The communications layer is responsible for handling the details of the FTP communication protocol for successful data transmission. Finally, the Global Layer provides a central repository for functionality common to all layers. Mainly, this layer contains Messaging objects that are passed between layers and serve as the main interface of communication. New messages can be added to provide new tasks that can be passed to any software object residing at any layer of the system.

Upon startup, the networking software reads the windows registry to determine the path where newly collected sensor data is stored. Once the path is identified, the File System Manager uses the Microsoft's .NET FileSystemWatcher object to monitor the path. As wind characterization data from the sensor nodes are collected by the Data Collection Software and placed in the outgoing directory, the File System Manger creates a compressed copy using the Compression Manager. Once compressed, the Networking Software uses the FTP Client, at the communication layer, to transfer the file using FTP. The FTP client component uses the Microsoft's .NET FtpWebRequest object, which resides in the System.Net namespace, to provide capabilities to easily implement reliable FTP clients without the effort required in traditional languages such as C/C++. By leveraging on the .NET framework, implementing features such as file system monitoring, compression, and FTP communication are trivial.

Using the layered design approach [11], new capabilities can be incorporated to the system without affecting any other part of the software, simply by adding new objects at specific layers. Each layer serves as compartment for related software objects, therefore minimizing the dependency between unrelated software objects. Also, by relying on a common messaging interface, software objects whether currently existing or newly added, are always independent from all other objects and from other layers in the software architecture. Figure 9 displays a sequence diagram for the Networking Software operations discussed above.

B. Data Processing Segment

The data processing segment contains one Client Server Communication Interface (CSCI), namely: Remote Server. The Remote Server consists of a Personal Computer (PC) executing customized software. The minimum hardware requirements of the server are: (1) mass data storage capabilities and (2) Ethernet network card. The minimum software requirements are: (1) Windows XP, (2) Microsoft .NET Framework 2.0, (3) Custom Server Software, (4) Microsoft's Internet Information Server (IIS), and (5) FTP Server.

1) Remote Server Configuration

The Remote Server requires custom configuration in order to operate properly. First, the Remote Server requires a public static IP address to allow communication between the collection and client segments. Once communication is established, the FTP server requires multiple different FTP directories where data from the different house locations are stored. FTP clients accounts are created each containing a home directory, username, and password. Each field laptop is configured with specific FTP account, therefore by using unique accounts, the data are sent to the right location (i.e. home directory) in the Remote Server.

2) Remote Server Software

The Custom Server software is a GUI-less windows service that executes in the background from the moment the computer is powered on. Its design reuses the design layers from the Networking Software with small software changes. Specifically, the Custom Server Software design reuses the File System Manager, Compression Manager, Event Manager, and all objects residing in the Global Layer. These objects revert the operations applied to the sensor data at the collection segment before transmission.

The server software is configured in the windows registry to monitor mass storage locations for new files. Upon startup, the server software reads the windows registry to determine the file system paths to monitor. Each file system path stores sensor data from individual houses. Therefore, data collected from one house is not combined with data from other houses. This process is for a case of four houses is illustrated in Figure 8.

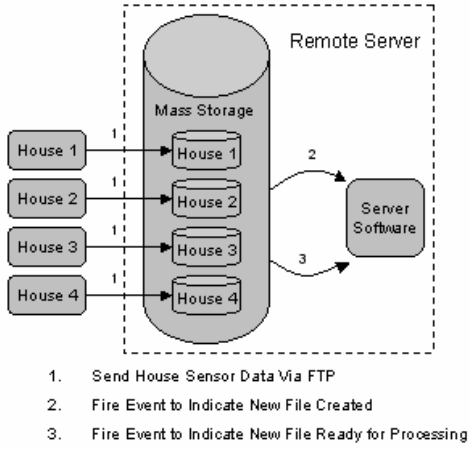

Figure 8 – Server Side Process

Once data are transmitted, the server software decompresses it using the Compression Manager, and creates HTML files using the Web User Interface object that resides in the User Layer. Once in HTML format, client computers can view the data from any location using the Internet Browser. The details of this process are captured in the server software sequence diagram as seen in Figure 9.

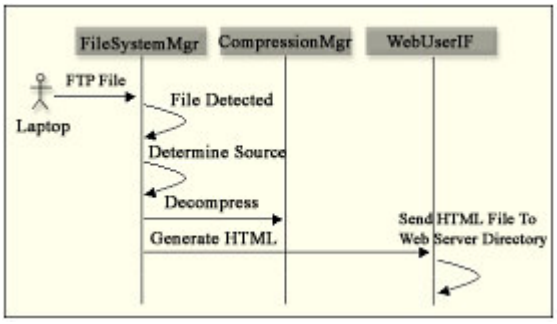

Figure 9 – Server Software Sequence Diagram

C. Client Segment

The client segment is composed of computers connected to the Internet. The main requirements for computers in the client segment are Internet connection and Internet Browser (e.g., Explorer, Firefox, etc.). The client segment software is composed of HTML files that are dynamically created and delivered to the client segment upon requests using the HTTP protocol. The HTML files are configured to initiate an internal timer of 5 minutes. When the timer expires, the browser initiates an HTTP request to the Remote Server (in the Data Collection Segment), which provides an HTTP response containing the latest sensor data collected.

V. PERFORMANCE ANALYSIS

Performance analysis is important to accurately evaluate the efficiency of the system. The main metrics associated with the software system are: sampling rate and network transmission rate. The following sections provide analysis for both metrics.

A) Sampling Rate

Sampling rate is the average number of samples per second collected by each remote sensor unit. Due to limited throughput on the Sensor-Base interface, the sampling rate varies according to the number of remote sensor units installed on the residential structure. A simplified timing diagram outlining the operation of the system is presented in Figure 10. For the sake of simplicity, the diagram shows only two remote units. Extension to the case of more units is straightforward. As seen, there are four variables that describe the data collection process. They are defined as follows:

- T_{CYC} : the time that elapses between two consecutive calls to the same remote unit
- T_{RB} : the time required for data transfer from remote unit to base unit
- T_{BC} : the time required for data transfer from base unit to data collection software
- N_R : the number of active remote sensor units.

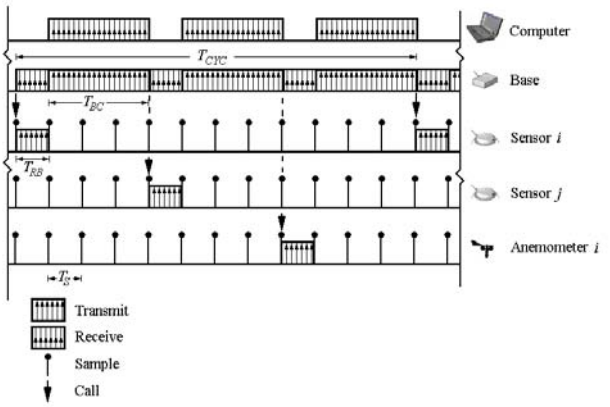

Figure 10 – Simplified Timing Diagram of the System

Initial sampling rate for the software system was computed in [5] using the metrics above, however through the application of optimization techniques in the real time embedded software, the sampling rate calculation formulas are modified as follow:

$$T_{CYC} = (T_{RB} + T_{BC})N_R \quad (1)$$

$$T_{RB} = \frac{8N_S}{R_{air}} + \frac{N_S}{4}T_P + 2T_{RTR} + T_{OHR} \quad (2)$$

$$T_{BC} = \frac{8N_S}{R_{serial}} + \frac{N_S}{4}T_P \quad (3)$$

$$T_S = \left(\frac{32}{R_{air}} + T_P\right)N_R + (2T_{RTR} + T_{OHR}) \quad (4)$$

$$R_S = \frac{1}{T_S} \quad (5)$$

where N_S is the number of samples read on a remote sensor unit; T_P is the time required to ensure data transmission; T_{RTR} is the time required for the transceiver to stabilize after switching from receive and transmit modes; R_{air} is the transmission rate on the air interface (in kbps), and T_{OHR} is the overhead time experienced due to communication delays on remote sensor units caused by bad reception quality (RSSI), low battery charge, or other defects in the remote sensor units. The T_{OHR} variable has been approximated for the current system configuration and categorized in Table I.

TABLE I. VALUES (IN MSEC) AND CATEGORIES FOR T_{OHR}

T_{OHR}	Category
0	Excellent
5	Very Good
10	Good
15	Regular
20	Bad

Given the current system parameters of T_P= 1 msec, T_{RTR} = 3 msec, R_{air} = 45 Kbps, one can easily compute the approximate relationship (Rs) between the number of remote sensor units and achievable sampling rate at each level of T_{OHR} (i.e., Excellent = 0, Very Good = 5, Good = 10, Regular = 15, and Bad = 20 msec.). Results are shown in Figure 11.

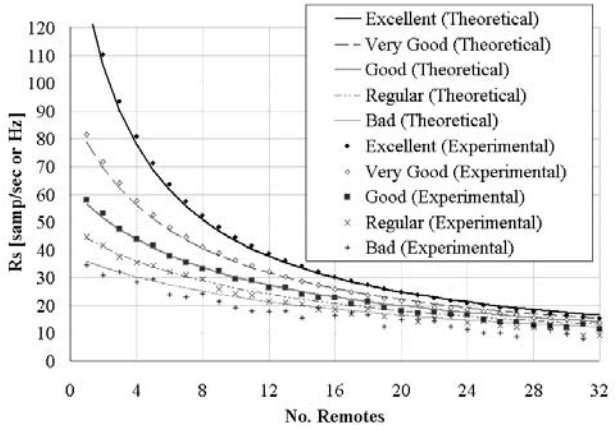

Figure 11 – Sampling Rate versus Number of Remote Sensor Units

Figure 11 shows both the expected theoretical values and the experimental values for *Rs* at different levels of T_{OHR}. It can be seen that the system's measured sampling rate closely matches the expected theoretical sampling rate at all levels of T_{OHR}. Based on Figure 11, Table II provides estimates of achievable sampling rates for system deployment having different number of remote sensor units.

TABLE II. ACHIEVABLE SAMPLING RATE FOR VARIOUS SYSTEM CONFIGURATIONS

N_R	T_{RB} msec	T_{BC} msec	T_{CTC} sec	$R_s actual$ samp/sec
1	7.711	1.842	0.01	134.81
5	14.556	9.211	0.119	71.15
10	23.111	18.421	0.415	44.4
15	31.667	27.632	0.889	31.942
20	40.222	36.842	1.541	24.656
25	48.778	46.053	2.371	19.821
30	57.333	55.263	3.378	16.34
31	59.044	57.105	3.601	15.754
32	60.756	58.947	3.83	15.198

B) Networking Software Efficiency

Most of the communications efficiency achieved by the networking software is attributed to compression provided by the Microsoft's .NET framework. The compression component resides in the System. IO.Compression namespace and allows programmatic compression and decompression using the GZIP format, which uses a broadly accepted algorithm. The built-in .NET compression tools are convenient and easy to use; however, experimental results showed that other formats, such as the .zip format, provide better compression as the size of the data increases [9]. Also, it was noticed that the time for compression is longer for the .NET compression tools. Figure 12 displays the results of compressing several ASCII files using both the .NET compression and the .zip format:

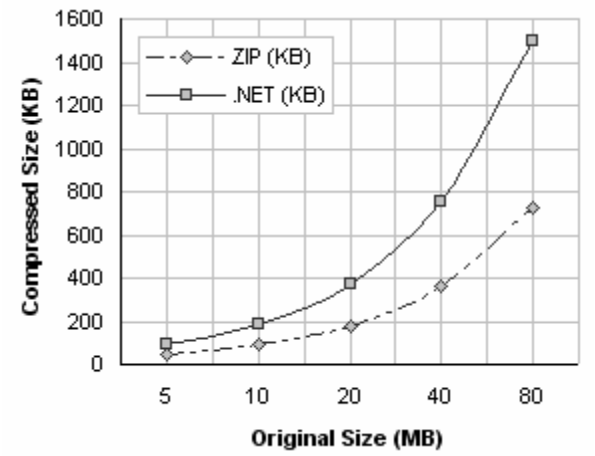

Figure 12 – Comparison between .NET built-in compression and standalone ZIP

It can be seen that the compression algorithms have relatively similar efficiency when the original data are less than 10 MB. After 10 MB the increased compression efficiency of the .zip format is evident. However, since the software system is designed to normally transfer 5 MB of data every 5 minutes, it does not experience much of the .NET compression inefficiency. After compression using .NET, 5 MB of data are compressed to approximately 94 KB, which is easily handled by either one of two wireless networks. Even with the known drawbacks, compression with .NET shows great advantages and significant gains in bandwidth usage, as seen in Figure 13. Figure 13 shows the theoretical characterization of transmission time using the networking software at several supported data size.

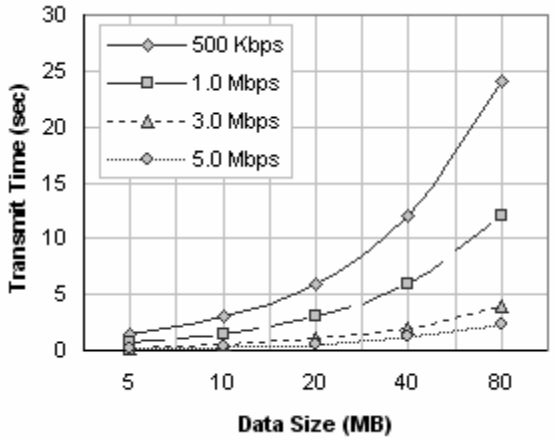

Figure 13 – Transmit Time versus Data Size (in sec)

Both communications links used by the networking software (i.e., WLAN and Wireless Cellular Network) provide theoretical bandwidth above 500 Kbps. However, when using cellular data networks, throughput could be limited by coverage, signal quality, and overall data traffic demand placed on the cellular network in the deployment area. Assuming that degradation lowers the data rate to 500 Kbps, the networking software can still transfer up to 80 MB of data in less than 25 seconds, which is way below the 5 minutes transmission cycle.

Using the theoretical transmission calculation from Figure 13, table III shows the resulting transmission time (in seconds) for various data sizes using the networking software.

TABLE III. THEORETICAL TRANSMISSION TIMES (IN SEC)

Data Size (MB)	500 Kbps	1.0 Mbps	3.0 Mbps	5.0 Mbps
5	1.50	0.75	0.25	0.15
10	3.01	1.50	0.50	0.30
20	6.02	3.01	1.00	0.60
40	12.03	6.02	2.01	1.20
80	24.00	12.00	4.00	2.40

Since the typical operation consists of 5 MB data file transmission, the expected transmission time from the Data Collection Segment to Data Processing Segment is (1.50 + .75 + .25 + .15) /4 = .7 sec. Preliminary tests have resulted in comparable numbers between theoretical and experimental.

VI. BENEFITS OF SOFTWARE SYSTEM

There are many benefits of using the described software system. First, by separating application specific software from the communication framework, the software can easily be ported to a wide variety of sensor network applications with little effort. Second, by leveraging on Microsoft's .NET framework, the software system employs a relative small amount of code to add capabilities that would have made the software more complex and error prone if done with an older language such as C or C++. By embedding well known object oriented design patterns into the architecture, the software has a high level of modularity, which results in reusable and easily maintainable code as proven by the reuse in both collection and data processing segments. Also, addition of new capabilities to the system such as new user interfaces, new processing tasks, or new communication protocols is trivial. Another notable benefit is the use of a convenient web interface for status. This means that information is available anywhere, anytime. Finally, the use of both wireless LAN and cellular networks provide robustness in harsh environments. Power outages will not be an issue as long as the field laptop operates on battery power and the cellular network is available.

VII. CURRENT AND FUTURE WORK

The system was recently tested at the University of Florida using their Hurricane Simulator on a mock up house-roof. A total of 15 pressure sensors and the Young anemometer were deployed. The schematics of the sensor locations and a picture of the deployment are presented in Figure 14. The software system was used for the collection and on-line system performance monitoring. The pressure and wind speed data from selected sensors are shown in Figures 15 and 16. The effective sampling rate of each channel was 30 samples/s. Sensor number 16 was mounted inside an enclosure to measure the reference pressure.

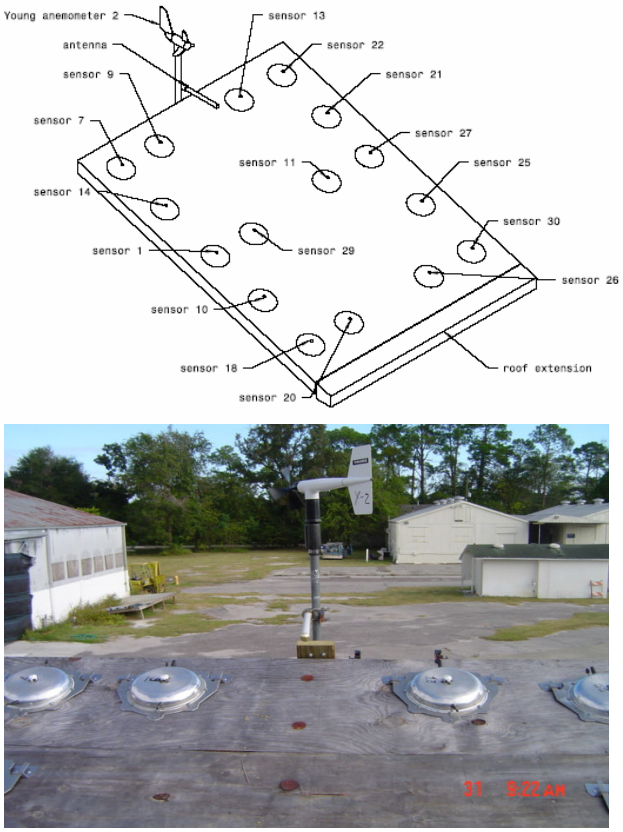

Figure 14 - University of Florida (Gainesville) deployment

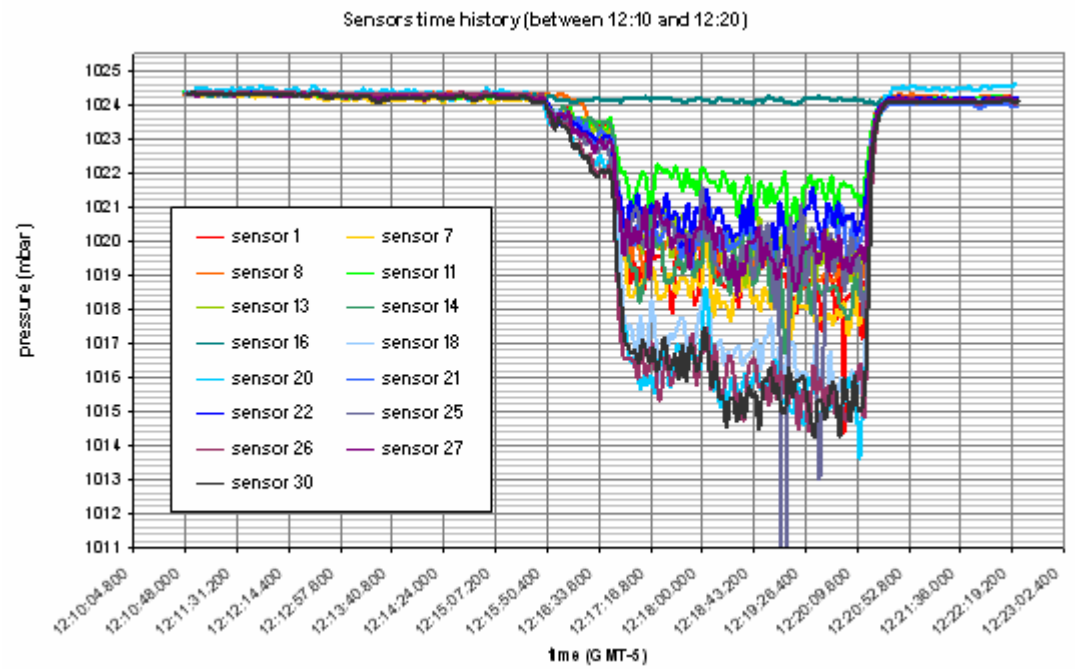

Figure15 - Time trace of pressure variation during wind gust

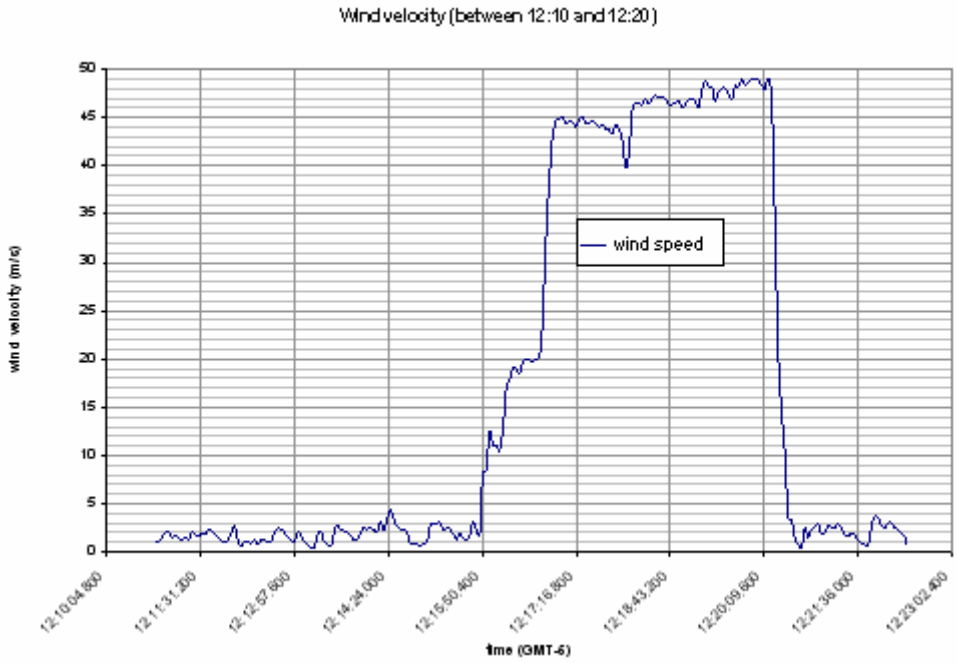

Figure 16 - Time trace of 3-sec average wind gust.

Although the software system described in this paper performs adequately for the task at hand, there are several avenues for its improvement. Some of the possible improvement directions are listed as follows.

A) Encryption

Security issues in network communications can be the most important requirement for some remote wireless sensor network applications. For this reason, the software architectural framework must include cryptographic support to fit the needs of these applications. Cryptography may be one of the most challenging tasks for software developers, since it requires knowledge of advanced encryption algorithms. Native to the .NET framework is the System.Security. Cryptography namespace which includes tools to facilitate the process of providing security to software applications via encryption. Future versions of the architecture presented in this paper could benefit from a Crypto Manager object that resides in the Manager Layer and handles all aspects of encryption and decryption when communicating across remote sties.

B) Connection Management

The current implementation requires configuration of the field laptop to Wireless LAN or cellular network. Switching between communication links requires physical presence at the site. Current efforts are under way to investigate ways in which the communication link can be switched automatically via software in case one of the links fails.

C) Field Laptop Remote Desktop Management

Current efforts are underway to add remote desktop capabilities to the field laptop. Open source software solutions have been tested and initial results allowed complete control of the field laptop from the Remote Server through both wireless LAN and cellular network. Further investigation is required to identify security considerations to safely add this feature to the system.

VIII. CONCLUSION

We have presented a distributed software system used in a real sensor network to collect, transmit, and present hurricane wind data in real time. The main goal is to safely collect wind data as it interacts with residential structures to determine possible effects caused by this interaction. The results of this study may serve as an influential factor in the way engineers design residential structures to minimize loss due to hurricanes. The complete system is currently under test and will be ready for deployment for the 2009 hurricane season.

ACKNOWLEDGMENT

This material is based upon work supported by the National Science Foundation under Grant No. 0625124. This support is gratefully acknowledged. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

The authors are also thankful to the University of Florida's Civil Engineering Department for providing their hurricane simulator test facilities

REFERENCES

[1] Rappaport, Ed., http://www.nhc.noaa.gov/1992andrew.html, Preliminary Report. National Hurricane Center. Retrieved on 2008-04-10.

[2] Beven, Jack. http://www.nhc.noaa.gov/pdf/TCR-AL042005_Dennis.pdf, Retrieved on 2008-04-10.

[3] Knabb, Richard D., Brown, Daniel P., Rhome, Jamie R., http://www.nhc.noaa.gov/pdf/TCR-AL182005_Rita.pdf, NHC Rita Report (English). Retrieved on 6 June, 2008.

[4] Pasch, Richard J., Blake, Eric S., Cobb, Hugh D., Roberts, David P., http://www.nhc.noaa.gov/pdf/TCR-AL252005_Wilma.pdf Retrieved on 6 June, 2008.

[5] Kostanic, I., Subramanian, C.S., Pinelli, J.-P., Buist, L., Velazquez, A., and Wittfeldt, A., "Monitoring of Hurricane Wind Pressures and Wind Speeds on a Residential Home Roof with Wireless Instrumentation," in proceedings of Structural Engineering Congress, Vancouver, Canada, April 24-26, 2008.

[6] Pinelli, J-P., Subramanian, C.S., Lapilli, C., and Buist, L., "Application of a Wireless Pressure Sensing System to Coastal Wind Monitoring," Wind and Structures, An International Journal, Vol. 8, No. 3, 2005, pp 179-196.

[7] Subramanian, C.S., Pinelli, J.-P., C. Lapilli, and Buist, L., "A Wireless Multi-Point Pressure Sensing System: Design and Operation," IEEE Sensors Journal, Vol.5, No.5, October 2005, pp 1068-1074.

[8] Holma, H., Toskala, A., HSDPA/HSUPA for UMTS, John Wiley and Sons, LTD., 2006.

[9] Otero, C. E., Kostanic, I., Subramanian, C., Pinelli, J. P., Velazquez, A., "A Distributed Framework for Remote Monitoring of Hurricane Wind Pressures and Wind Speeds using Wireless Sensor Networks", Proceedings of 2008 International Conference on Wireless Networks, 2008.

[10] Gamma, E., Helm, R., Johnson, R. and Vlissides, J.M., "Design Patterns: Elements of Reusable Object-Oriented Software", Addison-Wesley Professional Computing Series, 1994.

[11] Buschmann, F., Meunier, R., Rohnert, H., and Sommerlad, P., "Pattern-Oriented Software Architecture Volume 1: A System of Patterns", Wiley, 1st edition, 1996.

[12] Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E., "Wireless sensor networks: a survey", Computer Networks 38 (4) (2002) 393–422.

[13] Durst, C.S., "Wind Speeds over short periods of time", Meteor. Mag., 89, 181-187, 1960

[14] Marshall, R.D. and W.R. Krayer, "Gust Factors applied to hurricane winds", Bull. Amer. Meteor. Soc.,73, 613-617, 1992

[15] L. Kristensen and P. Kirkegaard. 'Digitization noise in power spectral analysis'. J. Atmos.Oceanic Technol., 4:228–335, 1987.

[16] J. O. Smith and P. Gosset: "A Flexible Sampling-Rate Conversion Method" ICASSP-84, Volume II, pp. 19.4.1- 19.4.2. New York, IEEE Press M. Young, The Technical Writers Handbook. Mill Valley, CA: University Science, 1989.

[17] T.A Ramstad "Digital Methods for Conversion between Arbitrary Sampling Frequencies". IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-32, No. 3, June 1984.

Dr. Carlos E. Otero was born in 1977 in Bayamon, Puerto Rico. He received a B.S. in computer science, M.S. in software engineering, M.S. in systems engineering, and Ph.D. in computer engineering from Florida Institute of

Technology, in Melbourne, FL. His primary research interests include various topics in simulation, performance analysis, and optimization of wireless ad hoc & sensor networks.

He has over 10 years of industry experience in satellite communications systems, command & control systems, wireless security systems, and unmanned aerial vehicle systems. He currently works at Northrop Grumman Corporation.

Dr. Ivica Kostanic was born in 1968 in Belgrade, Yugoslavia. He obtained BSEE, MSEE and PhD from Belgrade University, Florida Institute of Technology and University of Central Florida respectively. His principal research interests include various topics in radio communication, cellular systems and wireless sensor networks.

Currently, he is faculty member in Electrical and Computer Engineering of Florida Institute of Technology where he teaches courses related wireless communication, personal communication and microwave circuit design. He is the technical director of Wireless Center of Excellence (WiCE) which is a group at Florida Tech dedicated to promoting research in wireless communication and computing technologies.

Dr. Kostanic is a member of IEEE Communication Society.

Dr. Chelakara S. Subramanian was born in Mumbai (formerly Bombay), India on December 24, 1950. His academics qualifications are Ph.D. mechanical engineering,1982, University of Newcastle, Australia**.** M.E. aeronautical engineering, 1975, Indian Institute of Science, Bangalore. B.E. mechanical engineering, 1973, BMS College of Engineering, Bangalore University .

Currently he is a Professor and Program Chair in the Department of Mechanical and Aerospace Engineering at Florida Institute of Technology (FIT), FL, His previous appointments include Sabbatical Visiting Faculty (2002-2003) at Air Force Institute of Technology and Turbine Testing Division, Propulsion Directorate, Wright-Patterson Air Force Base, OH; Program Chair of Aerospace Engineering (1996 - 1999) at Florida Institute of Technology (FIT), FL; ASEE Summer Visiting Faculty Fellow (June - Aug 1996, and 1999) Aerodynamic Measurement Competency and ICASE Summer Visiting Faculty Fellow (June - Aug 1992) at the Flow Physics Branch of NASA Langley Research Center, VA; Adjunct Professor (1988- 1991) in the Department of Mechanical Engineering, Naval Postgraduate School, Monterey, CA; Research Engineer (1986- 88) at British Maritime Technology, Newcastle, UK, and Research Associate (1982- 1986) at Imperial College of Science and Technology, London UK.

Professor Subramanian's current research activities are in advanced fluid dynamics instrumentation, wind engineering, and complex turbulent flows. He is an associate fellow of AIAA, fellow of Society of Engineers, UK, senior member of ASME, licensed Professional Engineer in UK and a member of International Society of Professional Engineers in France. He has authored over 90 technical publications in international journals, books and proceedings and, has a US patent to his credit.

Dr. Jean-Paul Pinelli received a B.S. in civil engineering from the University of Buenos Aires, Argentina, and M.S. and Ph.D. in civil / structural engineering from Georgia Institute of Technology, Atlanta, GA. His main research interests are in wind engineering, risk modeling, and hurricane damage mitigation. He is currently a Professor of civil engineering at Florida Tech, Melbourne, FL.

Dr. Pinelli is a member of the American Association for Wind Engineering, the American Society of Civil Engineers, and the American Society of Engineering Education among others.

Larry Buist was born in 1942 in Grand Rapids, Michigan. He has over 20 years of experience in the electronics industry and holds a patent in the video game industry. He currently works at Florida Institute of Technology in the design and construction of electronic and electro-mechanicals circuits on a variety of research projects.