Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 34 (2014) 103 – 110

Procedia Computer Science

The 9th International Conference on Future Networks and Communications (FNC-2014)

Wireless Sensor Network System Design using Raspberry Pi and Arduino for Environmental Monitoring Applications

Sheikh Ferdoush, Xinrong Li*

Department of Electrical Engineering, University of North Texas, Denton, Texas, 76203, USA

Abstract

With over a decade of intensive research and development, wireless sensor network technology has been emerging as a viable solution to many innovative applications. In this paper, we describe a wireless sensor network system that we have developed using open-source hardware platforms, Arduino and Raspberry Pi. The system is low-cost and highly scalable both in terms of the type of sensors and the number of sensor nodes, which makes it well suited for a wide variety of applications related to environmental monitoring. Overall system architecture and the design of hardware and software components are presented in details in this paper. Some sample deployment and measurement results are also presented to demonstrate the usefulness of the system.

Selection and peer-review under responsibility of Conference Program Chairs

Keywords: Arduino; environmental monitoring; Raspberry Pi; wireless sensor network; ZigBee.

1. Introduction

With over a decade of intensive research and development, wireless sensor network technology has been emerging as a viable solution to many innovative applications. Early works on sensor networks and cyber physical systems have been focused on the development of enabling technologies by addressing a myriad of technical challenges such as multihop routing, communication abstractions, middleware and operating systems (OS), and semantic abstractions and sharing of data. Most of the early testbed systems have been built using early stage sensor network research platforms such as CrossBow (now MEMSIC) motes and TinyOS software framework [1]. The sensor network hardware platforms are basically low-power embedded microcontroller systems with some onboard sensors and analog I/O ports to connect sensors. A suite of software components also need to be developed, including OS, sensor/hardware drivers, networking protocols, and application-specific sensing and processing algorithms.

There are a large number of prior efforts in building wireless sensor network systems in the literature. For example, a large-scale soil moisture monitoring sensor network system is developed using CrossBow motes and it is integrated into a comprehensive environmental infrastructure system, Texas Environmental Observatory (TEO), as described by Yang et al. [1] MoteLab [2] was an experimental wireless sensor network deployed at Harvard University. It provided a

* Corresponding author. Tel.: +1-940-891-6875 ; fax: +1-940-891-6881.
E-mail address: xinrong@unt.edu

Selection and peer-review under responsibility of Conference Program Chairs
doi:10.1016/j.procs.2014.07.059

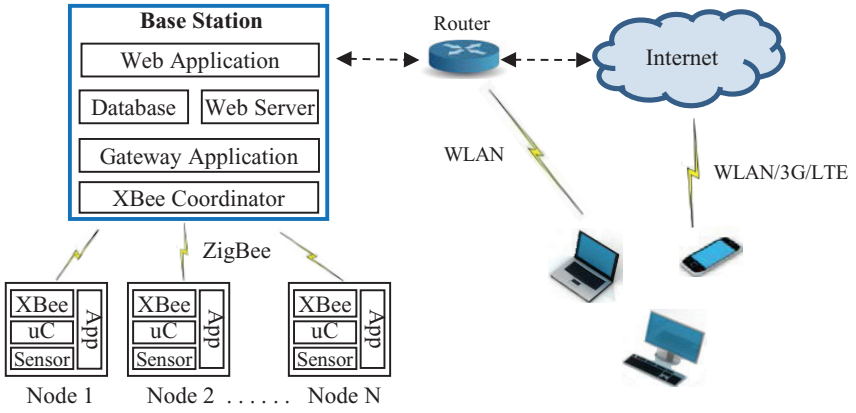


Fig. 1. Overall system architecture.

public testbed for development and testing of sensor network applications via an intuitive web-based interface and it was decommisioned in January 2014. INDRIYA[3] is a three-dimensional wireless sensor network testbed developed at National University of Singapore, following the same design as MoteLab. Some other related works include the web services approach for the design of sensor networks[4], and a number of web-based applications and software architectures for wireless sensor networks[5,6].

More recently, we have witnessed a new wave of developments in open-source hardware/software, standardization, and commercialization of wireless sensor network technologies. The IEEE 802.15.4 standard specifies the physical and medium access control layers for low data-rate wireless personal area networks[9]. ZigBee is a low-cost, low-power, wireless mesh networking standard built upon 802.15.4[10]. The 802.15.4 RF transceivers and ZigBee protocol stacks are now available as commercial-off-the-shelf (COTS) modules for rapid prototyping of wireless sensing and actuation systems. For example, the Digi XBee series OEM modules implement the IEEE 802.15.4 radio and ZigBee networking protocol[11] and have become very popular in application system development.

Sensor network systems, like most embedded systems, needs to be tightly coupled to their applications. However, the aforementioned recent advances have helped to reduce the complexity of implementing wireless sensing and actuation systems and have made it fairly easy to implement a prototype system for proof-of-concept and demonstration purposes. In this paper, we present a wireless sensor network system developed using open-source hardware platforms, Arduino and Raspberry Pi, and the ZigBee module, XBee S2B. Such a design has the advantages of low cost, easy to build, and easy to maintain, as compared to some earlier designs such as the TEO system[1]. The major obstacles for sensor network technology to become a transformational force in engineering, scientific, and commercial application domains lie in its lack of reliability, flexibility, scalability, interoperability, and in its extreme difficulties in long-term deployment, operation, and maintenance especially by non-engineering application domain practitioners. The system presented in this paper represents a step forward towards addressing these challenging issues.

The rest of the paper is organized as follows. In Section 2, the overall system architecture is described. Then, in Section 3, the design of hardware and software components is presented in details. Some sample experimental deployment and measurement results are presented in Section 4 to demonstrate the usefulness of the design. Finally, the paper is concluded with a summary in Section 5.

2. Overall System Architecture

Building a wireless sensor network system requires development and integration of many hardware and software components. Figure 1 shows the overall system architecture of an environmental monitoring wireless sensor network system that we have developed. The system includes an in-situ base station and a number of distributed wireless sensor nodes. Each sensor node is a combination of sensors, microcontroller (uC), and a ZigBee radio transceiver,

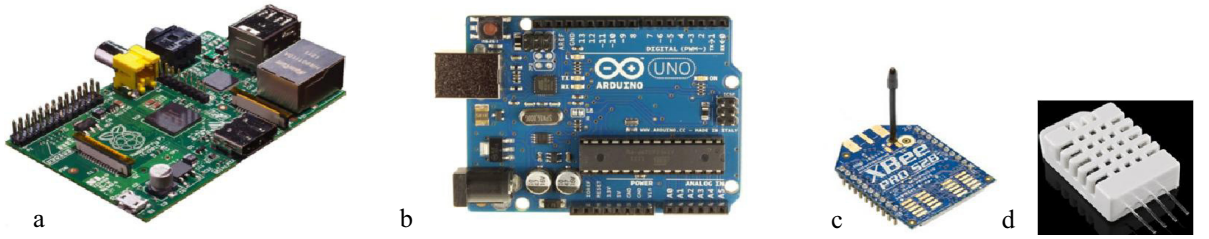


Fig. 2. (a) Raspberry Pi Model B; (b) Arduino UNO R3; (c) XBee Pro S2B; (d) RHT03 Temperature and humidity sensor.

i.e., the XBee module. In addition, there is a user application program on each sensor node, which handles sampling data from sensors in a certain well defined manner and communication with the base station.

The XBee module on the base station is configured as coordinator and the XBee modules on the sensor nodes are configured as routers as discussed in Section 3. Then, the XBee modules work together to form a mesh network topology using ZigBee networking protocols. To support remote online configuration and management of sensor nodes, we also implemented a gateway application in the base station. It is used to control the behavior of the distributed sensor nodes and to query all or a selected set of sensor nodes to retrieve data. The base station also includes a relational database management system (RDBMS) MySQL for data storage and management. To access the sensor nodes and the data from the outside world, a web application is developed on the base station using the Apache HTTP web server.

In our system architecture, we have combined the gateway node of wireless sensor network, database server, and web server in one single-board computer (SBC) hardware platform, which helps to reduce the cost and complexity of deployment. A web application is developed to provide users a convenient web interface to the system. Users can interact with the web application within the local area network or from any terminal on the Internet to access the sensor data or perform remote configuration and management of deployed sensor nodes. As compared to the large-scale TEO environmental system[1] that we have developed earlier, the system design presented in this paper is well suited for small-scale environmental monitoring and data collection applications.

3. Design of Hardware and Software Components

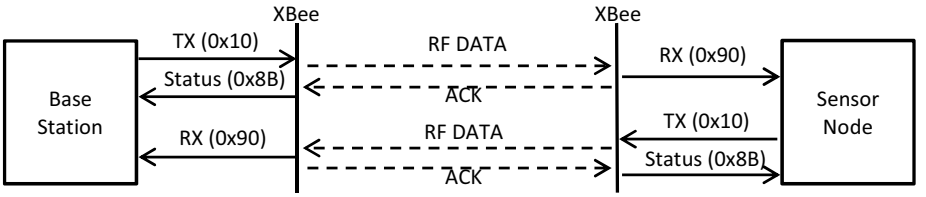
3.1. Design of Sensor Node

In this research, we developed networked sensor nodes using Arduino and Digi XBee moduels. Arduino is a widely used open-source single-board microcontroller development platform with flexible, easy-to-use hardware and software components. Arduino Uno R3 is based on Atmel Atmega328 microcontroller and has a clock speed of 16 MHz. It has 6 analog inputs and 14 digital I/O pins, so it is possible to connect a number of sensors to a single Arduino board. Arduino-compatible custom sensor expansion board, known as shield, can be developed to directly plug into the standardized pin-headers of the Arduino UNO board.

For wireless communication and multi-hop mesh networking, we used the commercially available ZigBee module, XBee Pro S2B from Digi[11], which operates at 2.4 GHz ISM band. Indoor communication range of the XBee moudle is 90 m whereas outdoor range is nearly 2 miles. With low power consumption and data rates up to 250 kbps, ZigBee devices are particularly suitable for fast prototyping for wireless sensor network applications. It is possible to build a simple star-structured network or complex mesh network using these devices. To exploit some of the advanced features of ZigBee we need to develop application accordingly both at the gateway and sensor nodes. To keep the initial development simple we used just one type of sensor on the sensor nodes; the sensor that we used is a low-cost humidity and temperature sensor RHT03. The hardware components used in this design are shown in Fig. 2.

The XBee module encapsulates 802.15.4 RF transceivers and ZigBee protocol stacks, and it can be easily integrated into microcontroller or microprocessor systems such as Arduino and Raspberry Pi through UART serial communication interface. The XBee module can be configured into three types of devices: coordinator, router, and end device. Coordinator has the capability to control the entire network. Router can relay messages in a tree or mesh network

a



b

TX Packet from Base Station to Sensor Node																				
Frame Filds	Start	Length		API ID	Frame ID	64 - bit Destination Address								16 - bit Network Address		Broadcast Radius	Options	Payload		Checksum
		MSB	LSB			DH				DL								PL1	PL2	
Byte Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Byte Value	0x7E	0x00	0x10	0x10	0x01	0x00	0x13	0xA2	0x00	0x40	0x7A	0xAC	0x8A	0xFF	0xFE	0x00	0x00	0x31	0x31	0x68

Fig. 3. (a) Communication scheme between base station and sensor node; (b) frame structure of the TX Packet.

topologies. End device can only communicate with the coordinator or the router. There can be only one coordinator in a network; the number of router or end device is not limited.

The XBee module has support for transparent (AT) and application programming interface (API) modes. With the AT mode, the XBee module behaves as a serial line replacement. With the API mode, all data entering and leaving the module is contained in frames that define operations or events within the module. The API mode is required when the network needs to be formed into a multi-node mesh or tree topology. In this work, we designed a simple frame-based communication scheme in the API mode to implement two functions in mesh network: 1) collect data from sensor nodes to base station, and 2) send configuration commands from base station to sensor nodes.

In the XBee mesh network, the coordinator node can query sensor nodes using either multicast or unicast communication mode. The communication scheme based on uni-cast is illustrated in Fig. 3a, and the frame structure of one of the API frames that we employed is shown in Fig. 3b. With this scheme, a unitcast frame can be directly sent from base station to a specific sensor node using the 64-bit unique device address of the XBee module on sensor node. More specifically, as shown in Fig. 3a, base station first sends a TX frame to a sensor node. Inside this TX frame the 64-bit device address of the receiver is embedded. So, the XBee module connected to sensor node receives the RF Data containing the TX frame and sends ACK to base station; meanwhile, it also sends a RX frame to the sensor node. Sensor node application processes the RX frame to get the payload information, and then, based on that information, prepares another TX frame containing the sensor data and the 64-bit address of the base station XBee module. The second TX frame is then transmitted back to base station.

3.2. Design of Base Station

For the base station, we used a low power credit-card-sized single-board computer Raspberry Pi Model B. The CPU on the board is an ARM processor with 700 MHz clock speed. CPU performance can be compared to a Pentium II 300 MHz processor and the GPU performance is similar to the original Xbox. It has a variety of interfacing peripherals, including USB port, HDMI port, 512MB RAM, SD Card storage and interestingly 8 GPIO port for expansion. Monitor, keyboard, and mouse can be connected to Raspberry Pi through HDMI and USB connectors and it can be used like a desktop computer. It supports a number of operating systems including a Debian-based Linux distro, Raspbian, which is used in our design. Raspberry Pi can be connected to a local area network through Ethernet cable or USB Wi-Fi adapter, and then it can be accessed through SSH remote login.

Functional building blocks of the base station, including gateway application, database, and web application, are shown in Fig. 4. Raspberry Pi is connected to the XBee coordinator module through a USB cable and a USB-to-UART serial converter circuit. The gateway application is the intermediate layer between sensor network and database. It sends out configuration and data collection commands to sensor nodes and inserts the data received from sensor nodes into the MySQL database. The gateway application is programmed in Python that comes built-in with Raspbian. Some of the required Python packages include PySerial 2.7, MySQL-python 1.2.5, Advanced Python Schedule (APScheduler 2.1.2), and XBee 2.1.0. The web application is built with Apache HTTP web server, and

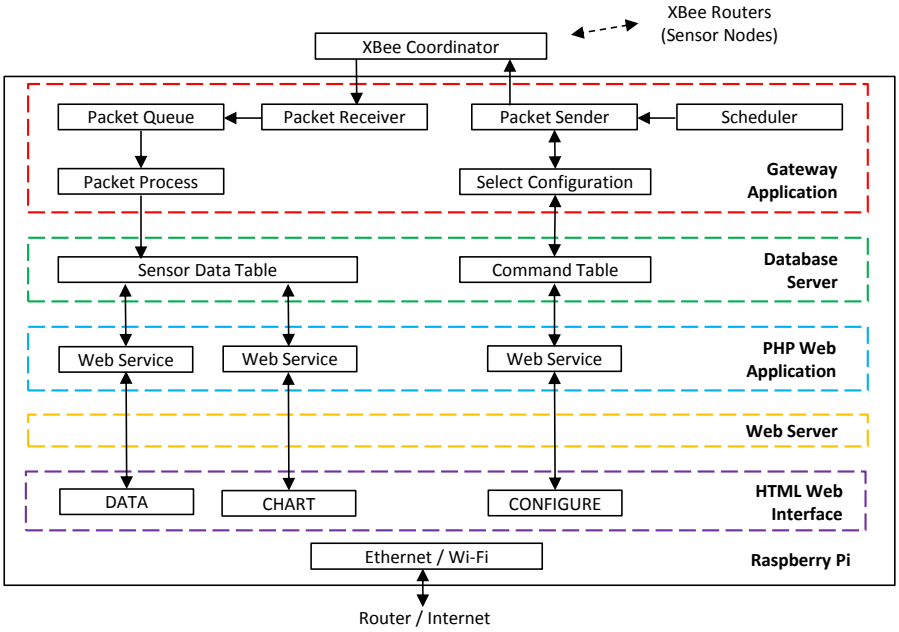


Fig. 4. Functional block diagram of base station.

the server-side web services are programmed in PHP. Users can access the HTML web interface through Ethernet or Wi-Fi connection within the local area network or from anywhere on the Internet when configured appropriately on the router.

3.3. Design of Web Interface

The client-side web interface is implemented with HTML, CSS, JavaScript, Ajax, jQuery, and Flot. HTML and CSS is used in combination to mark up and style the web page. JavaScript is used for client-side scripting to enable dynamic display and interactive user interface. jQuery is a widely-used JavaScript library that greatly simplifies JavaScript programming. Ajax, an acronym for Asynchronous JavaScript and XML, is a group of interrelated web development techniques used on the client side to create asynchronous web applications[8]. With Ajax, client-side web applications can exchange data with a server asynchronously in the background without interfering with the display and behavior of the existing page.

Flot is a JavaScript plotting library for jQuery, with a focus on simple usage, attractive looks and interactive features[7]. Generally, it supports all browsers that support the HTML5 canvas tag. In our design, Flot is used to visualize sensor data in both static and dynamic real-time graphical displays. In the real-time display mode, instead of refreshing and redrawing entire web page, Flot provides the capability to only update the chart with new data that are fetched periodically from the server. Ajax and jQuery are used in our design to feed the Flot charting functions with continuous flow of data from the MySQL database on the server via the web services written in PHP. The data is serialized into the JSON (JavaScript Object Notation) format to be communicated between server and client.

In the current implementation, we have designed three front-end utilities for experimentation and demonstration purposes: real-time display in chart (CHART), data access (DATA), and system configuration (CONFIGURE), as shown in Fig. 4. With the data access utility, users can download and view sensor data stored in the database conveniently. Users can also monitor the sensor data updates in real-time using the real-time display utility. With the system configuration utility, authorized users can configure the sensor network system with a number of global and node-level settings, including measurement period, which sensor nodes to measure, and which sensor to measure. The configuration requests generated by users on the web interface are sent asynchronously to web service on the server and the requests are inserted into a command table in the MySQL database. The gateway application monitors the

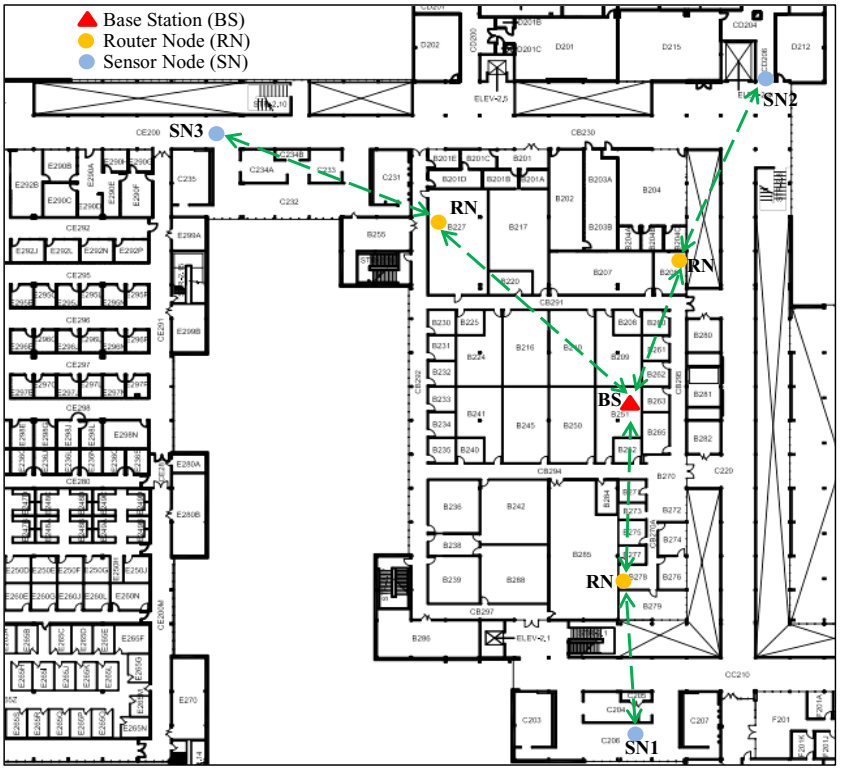


Fig. 5. Deployment structure of a wireless sensor network at the UNT Discovery Park office building.

command table periodically for any configuration updates to be sent to sensor nodes. The command table is used to share data between gateway application and web application. Such a design decouples gateway application and web application and greatly simplifies the inter-process data sharing problem. In general, environmental monitoring applications are tolerant to delays in the order of seconds or even minutes, so that the amount of delay introduced by such a method is acceptable in our targeted application scenarios.

The sensor nodes and base station can work independent of web application. Authorized users can directly access the data and command tables in the database and remotely configure sensor nodes from Raspberry Pi by logging into its Linux system. On the other hand, the web application can be updated without interfering or interrupting sensor nodes and the gateway application residing inside the Raspberry Pi. Such a design makes it very convenient to perform application-specific customization and revision of the system.

4. Experimental Results

We have deployed the system at the Department of Electrical Engineering office area of the UNT Discovery Park facility for development and testing purposes. The deployment includes a base station, 3 router nodes, and 3 sensor nodes, as shown in Fig. 5. The XBee modules on router and sensor nodes are all configured with the router device type. The XBee module on base station is configured with the coordinator device type. As a result, XBee modules can form into a mesh network topology using the ZigBee networking protocols. The coordinator device can support a maximum of 10 child nodes, and a router device can support a maximum of 12 child nodes. End devices are not capable of routing. So there should be enough routers before planning a large scale deployment. Theoretically, a coordinator device can support a network of up to 65,536 nodes, which is only limited by the 16-bit network addresses of individual nodes.

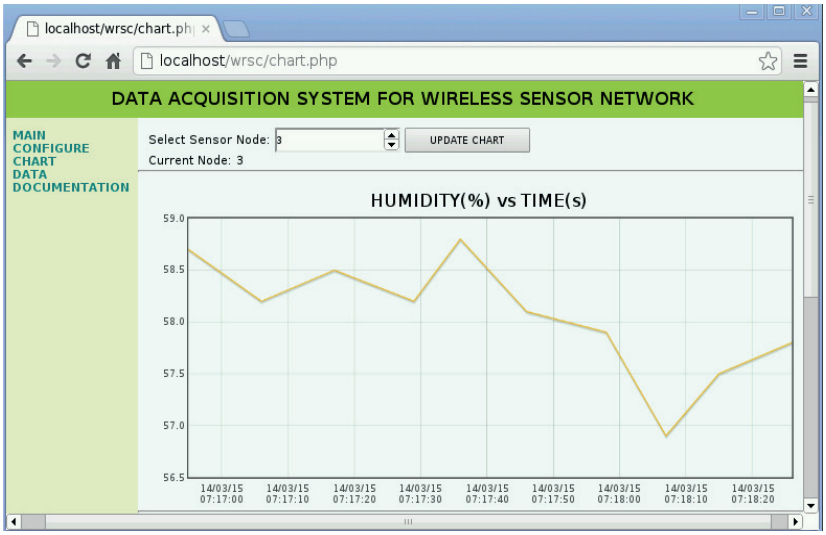


Fig. 6. Web interface display of the wireless sensor network system.

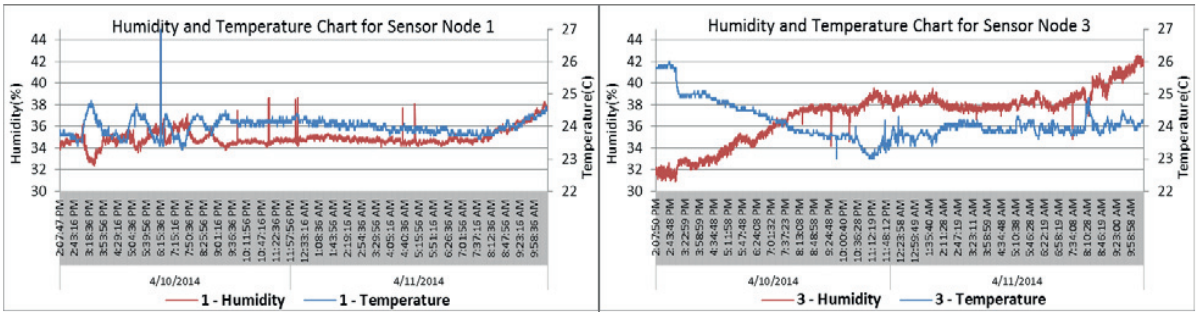


Fig. 7. Sample humidity and temperature data from Sensor Node 1 and Sensor Node 3.

The power level was set to the maximum setting with boost mode enabled for all XBee Pro S2B modules to provide a maximum transmit power of 18 dBm [12]. The boost mode improves sensitivity by 1 dB and increases output power by 2 dB, which improves the link margin and range [12]. We experimented with both multicast and unicast based communication schemes; both worked successfully without any packet loss [13]. However, further experiments and studies are needed to determine the coverage of each XBee radio transceiver in the selected deployment environment, which is useful in designing the optimum deployment structure.

A screen snapshot of the web interface is shown in Fig. 6. The sensor data can be conveniently downloaded from the web interface. The RHT03 relative humidity and temperature sensor is connected with the sensor nodes to collect measurement data. In Fig. 7, sample measurement results collected at Senor Node 1 and Sensor Node 3 are presented. While it is beyond the scope of this paper, the data can be used for post processing and analysis to serve the needs of various environmental applications.

5. Summary and Conclusions

In this paper, we have presented a wireless sensor network system designed with Arduino, Raspberry Pi, XBee, and a number of open-source software packages. The system has a number of attractive features, including low-cost, compact, scalable, easy to customize, easy to deploy, and easy to maintain. One major advantage of the design lies

in the integration of the gateway node of wireless sensor network, database server, and web server into one single compact, low-power, credit-card-sized computer Raspberry Pi, which can be easily configured to run headless (i.e., without monitor, keyboard, and mouse). Such a design is useful in many environmental monitoring and data collection applications. The XBee module from Digi encapsulates the 802.15.4 radio transceiver and the ZigBee protocol stack; it is capable of forming a complex mesh network structure on its own without intervention from user application program running on the microcontroller or microprocessor platform. As a result, it has significantly reduced the complexity of wireless sensor network system development. The detailed design and measurement results presented in this paper clearly demonstrate the usefulness of such a system.

As future work, the system design presented in this paper can be expanded in a number of different aspects. For example, additional sensing modalities can be integrated on sensor nodes to meet the needs of various monitoring applications. Also, the web interface can be further developed to implement more functionality in data visualization, management, and analysis among many others to provide better user interface and better user experience. Considering the limited storage space on the Raspberry Pi, it is also useful to integrate a second database server on the Internet or on the cloud storage service, and then upload and/or synchronize the data tables between the two data storage servers. A cloud storage is also useful in the scenarios where the base station (i.e., Raspberry Pi) resides in a private IP intranet, and it is not directly accessible from Internet.

References

1. J. Yang, C. Zhang, X. Li, Y. Huang, S. Fu, M.F. Acevedo. Integration of wireless sensor networks in environmental monitoring cyber infrastructure. *Wireless Networks, Springer/ACM*, Volume 16, Issue 4, pp. 1091-1108, May 2010.
2. G. Werner-Allen, P. Swieskowski, and M. Welsh. MoteLab: A wireless sensor network testbed. *Fourth International Symposium on Information Processing in Sensor Netoworks*, pp. 483-488, Boise, ID, USA, April 2005.
3. M. Doddavenkatappa, M.C. Chan, and A.L. Ananda. Indriya: A Low-Cost, 3D Wireless Sensor Network Testbed. *TRIDENTCOM*, 2011.
4. F.C. Delicato, P.F. Pires, L. Pirmez, L. Carmo. A flexible web service based architecture for wireless sensor networks. *The 23rd International Conference on Distributed Computing Systems Workshops*, pp. 730-735, May 2003.
5. S. Hussain, N. Schofield, and A.W. Matin. Design of a Web-based Application for Wireless Sensor Networks. *The 17th International Workshop on Database and Expert Systems Applications*, pp. 319-326, 2006.
6. X. Wei, J. Liu, G. Zhang. Applications of web technology in wireless sensor network. *The 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT)*, pp. 227-230, 2010.
7. Flot: Attractive JavaScript plotting for jQuery, available at `http://www.flotcharts.org/`
8. J.J. Garrett, Ajax: A New Approach to Web Applications, Adaptive Path, February 18, 2005. (see also `http://en.wikipedia.org/wiki/Ajax_(programming)`)
9. IEEE 802.15 WPAN Task Group 4 (TG4), available at `http://www.ieee802.org/15/pub/TG4.html`
10. ZigBee Alliance, available at `http://www.zigbee.org/`
11. Digi International Inc., available at `http://www.digi.com/`
12. XCTU: Next generation configuration platform for XBee, Digi International Inc., available at `http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/xctu`
13. S.M. Ferdoush. A low-cost wireless sensor network system using Raspberry Pi and Arduino for environmental monitoring applications. Master of Science Thesis, University of North Texas, 2014.