MODULE-BASED COURSEWARE FOR TEACHING SECURE WIRELESS SENSOR NETWORKS

Bo Sun, Lawrence Osborne
Department of Computer Science
Lamar University
Beaumont, TX 77713 U.S.A.
{bsun, ljosborne}@my.lamar.edu

T. Andrew Yang
Division of Computing and Mathematics
University of Houston – Clear Lake
Houston, TX 77058 U.S.A.
yang@uhcl.edu

ABSTRACT
Wireless Sensor Networks (WSNs) have become an ideal example software-hardware system for modern teaching and learning. Although there is an increasing trend to incorporate the principles of WSNs and their security issues into the mainstream undergraduate and graduate Computer Science curricula, it is still challenging to teach suitable WSN courses mainly because of the complexities of WSNs and the lack of easy-to-use course modules. Our *goal* is to develop practical course modules that can enable *experimental learning* and technology-based education. Specifically, the modules range from the basic software and hardware of WSNs to commonly used WSN services. To familiarize students with TinyOS programming, a hands-on project, the implementation of a *multihop data collection tree protocol*, is included in the modules. Summative evaluation was conducted when the modules were used in teaching a Computer Science course in WSNs. The evaluation results are discussed in this paper.

KEY WORDS
Wireless Sensor Networks, Module-based Courseware, Security

1. Introduction

A wireless sensor network (WSN) is made of a set of sensor nodes, each of which is a self-contained unit consisting of a battery, a radio module, sensors, and a low-speed on-board processor [27]. Normally the unit that consists of the radio module and the on-board processor is called *mote*. The most essential feature of wireless sensor networks is their being embedded in the real world. Sensors detect the world's physical nature, such as light intensity, temperature, sound, or proximity to objects [19]. A typical WSN consists of nodes collecting data from the environment and transmitting the collected data to a *base station*. An application on the base station analyzes the received data, performs appropriate computation, and displays the results on the user screen.

Combining advanced technologies from multiple Computer Science subject areas, including Computer Networks, Operating Systems, Software Engineering, and Embedded Systems, WSNs have become an ideal example software-hardware system for illustrating fundamental concepts; the principles of WSNs and their security issues have been incorporated into the mainstream undergraduate and graduate Computer Science curricula. Unfortunately, currently offered WSN courses lack a necessary component in science and technology education - *experimental learning* [7], which involves students in hands-on projects and is usually necessary for students to achieve effective learning in engineering and technology courses. Existing textbooks [14, 32] of WSNs also mainly focus on basic concepts and theories and are difficult for students (especially undergraduates) to understand.

In this paper, we present a module-based courseware to teach WSNs and their security issues. Our *goal* is to develop practical course modules that can enable *experimental learning* and technology-based education. Different from most existing WSN courses that are often offered as special research topics, our courseware covers both theoretic and practical aspects of WSNs and is suitable for both undergraduate and graduate students in primarily teaching oriented universities and colleges.

To decide the topics and the content of our course modules, we selected a suite of well-known papers related to WSNs, and investigated whether the ideas presented in those papers have been implemented in TinyOS [23], which is the predominant operating system for supporting wireless sensor networks applications. By including both the concepts and their actual implementations, we have developed course modules that not only cover the concepts of a specific protocol, but also provide students hands-on experience to program sensor nodes to run the protocol. Specifically, we have developed nine modules:

- *Introduction to Wireless Sensor Networks* introduces basic principles and characteristics of WSNs.
- *TinyOS* introduces the predominant open source operating system for WSNs – TinyOS [23].
- *Energy Management* introduces how energy, the most restricted resource in WSNs, is managed in TinyOS.

- *Radio and Medium Access Control (MAC)* introduces unique radio characteristics in WSNs. Based on these characteristics, we introduce one of the most popular MAC protocols - B-MAC [18].
- *Wireless Link Estimation* introduces how to perform wireless link estimation to estimate the quality of wireless links. Wireless link estimation is necessary to indicate the quality of a wireless link in order to implement efficient routing [5].
- *Data Collection and Dissemination* module, based on wireless link estimation, introduces how to perform data collection and dissemination [13].
- *Security of Wireless Sensor Networks* introduces basic security issues and their solutions in WSNs [17].
- *Localization and Secure Localization* introduces localization [21], one of the most widely studied problems in WSNs, and its security issues and solutions [15].
- *Aggregation and Secure Aggregation* introduces data aggregation [11] and its security problems and solutions [22]. Aggregation is one of the most common approaches to save energy in WSNs.

Our design of the course modules is based on the following philosophy: (a) Teaching WSNs should focus on both the fundamental principles and practical skills. Students should have opportunities to practice important concepts learned from WSN classes. (b) The design of the modules should enable easy integration into teaching fundamental Computer Science courses, including Operating Systems, Computer Networks, Wireless Communications, and Embedded Systems.

Therefore, unlike existing WSN courses and textbooks, our designed modules focus on the *system* aspects of WSNs that aim at resolving real world problems and creating practical solutions. To make the designed course modules practical and to cover the state-of-the-art technologies in WSNs, we designed our course contents based on the current release of TinyOS 2.x [12, 23]. Major concepts and algorithms are included to prepare students to understand the design principles of TinyOS.

To enhance students' knowledge with hands-on experience, we also developed a practical project, the design and implementation of a *multihop data collection tree protocol*, to enable students to practice important concepts and protocols of TinyOS. Students can further collect interesting data, for example, ambient light and temperature, based on their developed protocol.

2. Research Assumptions and Objectives

First, we assume that the current research trend of wireless sensor networks make them ideal to be integrated into Computer Science curricula. As a vivid example that students can touch and play with, WSNs provide a perfect combination of both operating systems and computer networks, which are required courses by ABET [1] for Computer Science majors. Our teaching experiences in operating systems and computer networks illustrate that it is usually challenging for students to understand relevant topics in these areas because traditional operating systems (like UNIX and Linux) and the Internet are built upon a huge amount of knowledge and their principles have become very complex. As a newly emerging and relatively simple technology, WSNs can significantly promote students' understanding of basic principles introduced in operating systems and computer networks. In addition, WSN projects involve the development of software applications, which provide a context for students to learn about programming languages and software development for wireless devices.

We also assume that the current research and existing literature in WSNs makes it a daunting challenge for faculty members and students in small and medium sized universities to learn. Research oriented textbooks and a simple collection of WSN papers are not suitable for teaching-oriented institutions. Therefore, a collection of suitable course modules with practical lab projects are necessary for both faculty members and students to grasp the principles of wireless sensor networks.

Motivated by the above concerns, we focus on the following objectives in this paper: (a) Having noticed that students may become overwhelmed by the massive amount of information in WSN research, we therefore present courseware modules which are relatively complete and cover fundamental topics in the areas of WSNs. (b) Our proposed modules avoid complex theories and focus on the introduction of basic principles in WSNs. To realize this, we accompany most protocols with example TinyOS implementations to facilitate *experimental learning* - to provide students with hands-on experience about how to build a wireless sensor network.

3. Related Work

WSN courses have been proposed by instructors in many colleges and universities. However, in most of those courses, only a reading list is presented and students are required to read papers and give research presentations. The final projects in these courses usually require writing a proposal illustrating research ideas. This kind of instruction format may be suitable for research oriented institutes, because in these institutions most students taking WSN courses aim at a Ph.D. degree. However, for most students in small universities or teaching-oriented colleges, reading papers alone may become a daunting task, which makes it challenging for students to grasp the principles of wireless sensor networks and their security issues. Our goal of creating a practical set of WSN course modules suitable for incorporating WSN principles into various courses cannot be achieved by simply compiling a list of the existing WSN papers and research projects.

There are also commercially published textbooks on the subject of wireless sensor networks and their security issues [14]. Unfortunately, like the existing WSN courses, most of those textbooks are research oriented. They cover the important research topics in WSN fields and focus on abstract ideas, typically lacking real world examples of WSN applications. It is therefore difficult for students to grasp the underlying principles of developing WSN applications.

4. A Practical Module-Based Courseware

Following our design philosophy, we first compiled a list of important WSN research topics and then searched highly-selective and premier WSN conferences and journal articles for state-of-the-art solutions. For example, the ACM Conference on Embedded Networked Sensor Systems (*ACM SenSys*) [2] is a premier conference focusing on practical and system related issues in the area of networked and embedded sensors. Many of our course contents were selected from the research results published in *SenSys*. Other important conferences and resources focusing on WSN system issues include USENIX Symposium on Operating Systems Design and Implementation (*OSDI*) [29], USENIX Symposium on Networked Systems Design and Implementation (*NSDI*) [28], TinyOS Extension Proposals (TEPs) and other documentation [24], TinyOS mailing list, and relevant technical reports. Moreover, because University of California Berkeley and Stanford University have dominated the development of TinyOS and their relevant protocols, relevant web sites from these universities also provide useful information and resources.

We adopted *XubunTOS* [30] as the platform for students to develop real world WSN solutions. Built from Xubuntu and TinyOS packages, XubunTOS greatly simplifies the installation of TinyOS by using a live CD. Regarding the specific WSN hardware, *MicaZ* and *TelosB* from CrossBow [31] were used as the wireless motes, *MTS300* was used as sensor boards to collect ambient light and temperature, and *MIB520* was used as the gateway to program motes.

4.1 Module One - Introduction to Wireless Sensor Networks

This module introduces the basic principles and characteristics of WSNs. Specifically, we start with real world examples to explain WSNs and their operational challenges, including energy efficiency, limited storage and computation, low bandwidth and high error rates, etc. *MicaZ* motes and *TelosB* motes are used as hardware platforms to give students more concrete examples.

4.2 Module Two - TinyOS

This module introduces the dominant open source operating system for WSNs - TinyOS. TinyOS differs from traditional operating systems in that it is designed as a lightweight operating system for achieving high efficiency in embedded systems, and therefore has no heavyweight kernel, no process management, and no virtual memory. TinyOS is little more than a non-preemptive scheduler. A quick look at the kernel code is necessary to master important concepts in TinyOS. Our designed module takes sample codes from the TinyOS 2.x kernel [26] to illustrate important ideas. These ideas include how TinyOS is designed to support different hardware platforms (MicaZ, Mica2, TelosB, Iris, etc.), how TinyOS is designed to support different hardware chips (MCU, radio, etc.) for a specific platform, how the data link protocol for ChipCon's CC2420 [3] and CC1000 [4] chips are designed and implemented, how the *timer* system is designed and implemented, and how the serial communication is designed and implemented.

TinyOS source codes provide many example applications, which can serve to illustrate the *nesC* programming language. Usually these applications consist of many *wired* components which can be utilized to illustrate the concept of object oriented design and implementation in software engineering. We use an example application that samples two sensors [9] to illustrate how different components interact with each other through *commands* and *events*. This example can facilitate the understanding of *wire*, *provide* and *use* in TinyOS components. This can also show the concepts of *modules* and *configuration*.

To illustrate networking in TinyOS V2, *message t*, an important abstraction, is used; it facilitates interactions among different link layers in wireless networks. Each link layer can define its own header, footer, and metadata structures. We use *CC1000* and *CC2420* as the example hardware to illustrate this concept.

Several representative WSN applications were adopted as vivid examples to facilitate students to deepen their understanding of TinyOS and WSNs. Important concepts, like *packet forwarder*, are often used in the TinyOS tutorial [25]. However, at this moment there are no specific figures to illustrate how to run WSN applications in a network environment. To fill this gap, we plot different network topologies for each application and use them in our course modules. All these network topologies can be downloaded from WSN class website: http://galaxy.cs.lamar.edu/~bsun/wsn/wsn.html.

Specifically, we use the following applications:

- *BaseStation - Listen – BlinkToRadio* is an application in which one mote periodically sends data to the base station through one-hop wireless communication, which can further transmit data through the Internet for further analysis;

- *Oscilloscope* is also a data collection application for one-hop WSNs, but it provides a GUI interface to facilitate data display in the PC side;
- *MultihopOscilloscope* is a data collection application for multihop WSNs, based on the *Collection Tree Protocol* [5]. It also demonstrates the self-organizing capabilities of WSNs;
- *MViz* is also a data collection application for multihop WSNs, but it provides a GUI interface that can generate the self-organized WSN network topology;
- *Octopus* is the most complex application we use in our course modules. Besides providing data collection functions, it also implements data dissemination functions for multihop WSNs. Considering the fact that WSNs are typically dynamic and network membership is not static, Octopus also provides a more dynamic view of network topology than that of MViz.

4.3 Module Three - Energy Management

Energy is the most stringent resource in WSNs, because sensor nodes are typically battery powered, especially when they are deployed in remote or hard-to-reach locations. To facilitate the application development and to avoid the explicit invocation of power control operations from the applications, TinyOS 2.x has integrated power management into its device drivers. *Virtualized*, *dedicated*, and *shared* are three main resource arbitration models for important resources like timer, bus, memory, and Analog-to-Digital Converter (ADC). We also use CC2420, Atmegal 128 Analog to Digital Converter, and MTS 300 as the example chips to illustrate these important ideas.

4.4 Module Four - Radio and Medium Access Control

WSNs demonstrate unique radio communication patterns, including radio irregularity, anisotropic signal strength, and anisotropic packet loss ratio [33]. All these factors have dramatic impact on higher layer protocols like MAC and their design and implementation. In our course modules, we use detailed experimental data to illustrate the relevant concepts.

We start with the introduction to some basic concepts about wireless communications, including signal, dB, dBm, Received Signal Strength Indicator (RSSI), Link Quality Indicator (LQI), and noise floor. These concepts are frequently used in many WSN papers. We then introduce MAC protocols which coordinate nodes to access shared wireless channels. The essential requirements of MAC protocols are energy efficiency, effective collision avoidance, efficient channel utilization, and scalability. We present a classification of MAC protocols in order for students to have a high-level conception. We focus on the introduction of Time Division Multiple Access (TDMA) and Carrier Sense Multiple Access (CSMA) protocols. Finally we introduce the details of B-MAC, which is a core module implemented in TinyOS. We focus on important concepts including Clear Channel Access (CCA) and Sleep/Wake scheduling using Low Power Listening (LPL), which are unique to WSN design.

4.5 Module Five - Wireless Link Estimation

In this module, we first introduce why wireless link estimation is necessary by showing details of the time varying nature of a wireless channel. The data we use come from the experiments with a real world WSN, which vividly demonstrate the relevant features of a wireless link.

We then introduce state-of-the-art link estimation techniques. First, we focus on ETX – Expected Transmission Count [6], whose purpose is to choose routes with high end-to-end throughout and find paths with the fewest expected number of transmissions. Secondly, we focus on four-bit wireless link estimation [8]. In this estimation, information from the physical layer, the data link layer, and the network layer are all integrated to estimate the quality of a link.

4.6 Module Six - Data Collection and Dissemination

Previously introduced modules lay the foundation for important WSN applications. Data collection and dissemination turn out to be the most basic and important applications promised by WSNs. Many collection and dissemination protocols have been proposed. However, most of the proposals typically show merely basic ideas and analysis of complexity and overhead of the proposed protocol, lacking details about the implementation.

When designing lab projects for the WSN course modules, we select the *Collection Tree Protocol* (CTP) as the data collection protocol, which is a very popular tree-based protocol implemented in TinyOS. CTP has received a lot of discussion in the TinyOS email list. We then introduce a data forwarding scheme which is tailored for extremely low duty-cycle data forwarding applications. Regarding data dissemination protocol, we select *Trickle* [13]. Like CTP, *Trickle* is one of the most commonly used protocols and has been implemented in TinyOS.

4.7 Module Seven - Security of Wireless Sensor Networks

Similar to other types of networks, wireless sensor networks are subject to various types of threats and attacks. Because security has become one of the major concerns to deploy WSNs in the real world, WSN course modules are not complete without modules focusing on WSN security. We start with the introduction of major attack models targeted at WSNs. We then focus on the *TinySec* protocol [10], which aims at providing message integrity and confidentiality at the data link layer.

TinySec has also become an implemented module in TinyOS. Our course module introduces the details with respect to how TinySec is designed and implemented.

4.8 Module Eight - Localization and Secure Localization

Localization is an important service for WSNs, because many WSN applications require location information of the sensor nodes. Furthermore, localization turns out to be a challenging task for WSNs.

We first give a high-level picture of localization and a classification of existing localization protocols - *range-based* and *range-free* localization. We then introduce a Minimum Mean Square Estimation (MMSE) based localization solution, because MMSE based solutions can summarize the basic ideas of most localization schemes. In the course modules, we use simple mathematics to show basic ideas of MMSE.

Based on MMSE, we then introduce the secure localization problem and one popular solution to provide correct location given malicious attacks [15]. We show how to derive a set of consistent location references and illustrate how to retrieve a set of good location references.

4.9 Module Nine - Aggregation and Secure Aggregation

Data aggregation has been proven to be an important method to reduce communication overhead and to save energy for WSNs. To address this need, we first introduce why we need aggregation - to suppress redundant messages and save energy. To illustrate this idea, we use a detailed example - counting how many nodes exist in a large-scale WSN; the example was taken from the Tiny AGgregation Service (TAG) approach [16]. We then move to general aggregation primitives - minimum, average, sum, and maximum - and explain their usage in real world applications.

We further present why we need *secure* aggregation - compromised sensor nodes can manipulate the result of aggregated values without limit, causing false information to be delivered to the base station. We then present the details involved in using a lightweight block cipher to provide an aggregation service which can provide confidentiality, integrity, and authenticity [22]. Our module skips the complex mathematical analysis and focuses on high-level ideas.

5. The Programming Project

A practical project is indispensible in order to enhance students' knowledge with hands-on experience. Therefore, we developed a project aiming at familiarizing students with hands-on programming of TinyOS. In the project, students need to develop a *multihop data collection tree protocol*. The protocol starts with the base station locally broadcasting a *tree construction* message, which includes its own ID and its depth as 0. The detailed information about how to set up the tree is as follows.

Initially, no node belongs to the data collection tree. When a node, say *A*, receives a tree construction message from node *B* the first time (i.e., node *A* has not joined the data collection tree yet), node *A* assigns its depth to be the depth of node *B* plus one, and sets its own parent to be node *B*. After this, node *A* rebroadcasts the tree construction message, with its own ID and depth. If node *A* has already joined the data collection tree when it receives a tree construction message from node *B*, node *A* just simply disregards that message. Figure 1 shows an example data collection tree after it is set up.

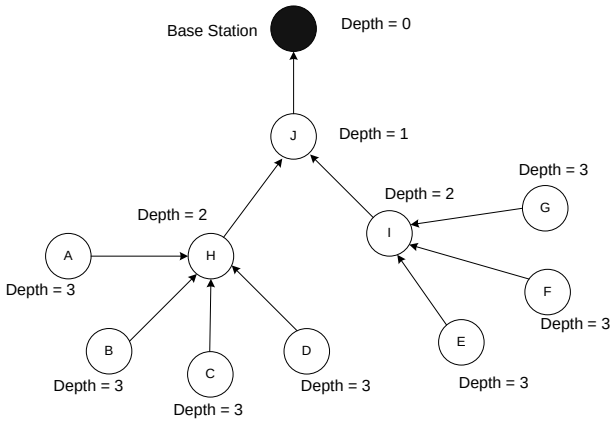

Figure 1. An Example Data Collection Tree

After the tree is set up, each node will start sensing and transmitting its ambient light intensity to the base station every one second. As simple as it is, this project can help students make sense of how sensor nodes work together to form a network and how data are passed from a node to the base station.

In practice, it is difficult to set up multihop communications. This is because each node has a relatively large transmission range by default. Therefore, all nodes deployed in a classroom setting are physically connected to each other. Because of this reason, single-hop communications are often formed in WSN course projects. To enable multihop communications in the project, the *setRFPower()* function provided by TinyOS can be used to change the transmission range of a mote.

6. Student Evaluations

The developed course modules were used in teaching a Computer Science course in Wireless Sensor Networks at the Department of Computer Science at Lamar University in summer 2008 and 2009. Near the end of the semester, we conducted preliminary student surveys that evaluated

the course modules and the project. There were 10 graduate students and one undergraduate student enrolled in that course. Of those 16 students, only two students were female, one from India and one from China. Of the 14 male students, four were domestic (American) students while the other two were international graduate students from India or Nepal. The ages of all the students were between 22 and 25. All of the students majored in Computer Science. Because the class size was small and there existed no control group, the student evaluation was mainly based on a simple class survey and therefore we did not find it necessary to apply statistical packages like SPSS [20] to data analysis. Instead, only the *mean* and the *standard deviation* of the student evaluation scores were computed as a preliminary measure of the effectiveness of the course modules and the project. Moreover, because there was only one female student and the students' ages were very close, the respective impact of age and gender was not analyzed in our evaluation.

Our evaluation questions had three categories: *general questions*, *questions about course modules*, and *questions about the project*. In Table 1, we present the evaluation results based on feedback from those 11 students who took the course. The score is on the scale of 1 to 5, with 5 being *Strongly Agree*. As we can see from Table 1, the feedback from students was very positive.

Table 1. Student Evaluation

Categories	Evaluation Questions	Average Score	Standard Deviation
• General Questions	1) I have learned a great deal from this class.	4.72	0.467
	2) Students have the opportunities to be exposed to practical aspects of wireless sensor networks.	5	0
• Questions about the Project	3) The project is necessary in this class.	5	0
	4) The project in this class is challenging.	4.72	0.467
	5) The project in this class is very interesting.	4.9	0.301
	6) The project in this class requires a lot of skills to finish.	4.5	0.505
	7) The project in this class is time-consuming.	4.5	0.505
	8) The project is a valuable part of this class.	4.72	0.467
	9) The project used in this class helps me improve my ability of problem solving.	4.8	0.405
	10) The project used in this class helps me improve my ability of creative thinking.	4.73	0.467
	11) The project used in this class helps me improve my ability of independent thinking.	5	0
• Questions about the Course Modules	12) The course materials covered in this class are difficult.	4.36	0.501
	13) It is time-consuming to understand the course materials.	3.72	0.905
	14) The course materials used in this class help me improve my ability of problem solving.	4.81	0.405
	15) The course materials used in this class help me improve my ability of creative thinking.	4.81	0.405
	16) The course materials used in this class help me improve my ability of independent thinking.	4.54	0.505
	17) The time spent in this class is worthwhile.	5	0

We also had some short essay evaluation questions which aimed at investigating advantages and disadvantages of this class. Based on students' answers, advantages of this course are as follow:

- Working with hands-on hardware and software is interesting.
- Students like the practical applications introduced in the class.
- The class is very practical.

Disadvantages mentioned in students' answers are as follow. We notice that the main disadvantages provided by students reflect the fact that it is challenging for students to master the fundamental principles of WSNs.

- *nesC* language is difficult to learn.
- Some ideas, especially the hardware aspects of sensor nodes, are complex to understand.

We noticed that students taking this class had big differences in their education background. Based on our interaction with students, we found that those students with insufficient academic preparation in Computer Science thought that the selected papers and projects were difficult and time-consuming for them to understand. It usually took those students much more time to read relevant materials before they could start working on the project. On the other hand, some students were already taking thesis options in the area of wireless sensor networks, and already had some preliminary background in WSNs before they took this course. Those students thought that the project was easy to complete.

7. Implications to Theory and Practice

Pedagogical research has shown that experimental learning and practices are critically important to the

success of Computer Science courses. On the one hand, if only the theories are taught without practice, the course will become a daunting task for most students and may directly conflict with most students' learning styles. On the other hand, if only the practice is provided without the underlying theories, the course will lack its theoretical depths, and few students would appreciate the utility of theoretical materials. Denning indicated that a portrait of computing organized around great principles and practices promote greater understanding of the science and engineering behind information technology [7]. One cannot fully understand a principle or theory without observing it from practice.

A WSN course provides a typical example to integrate scientific principles and technological applications. Therefore, we need to carefully meld WSN theories and practice in order to give students concrete examples about how WSNs work in order to increase students' abilities to absorb the lectured protocols and algorithms. The course modules presented in this paper provide a preliminary illustration of balanced courseware development that offers the instructor and the students an opportunity to experience the interplay between theory and practice.

8. Conclusions and Future Work

In this paper, we present a module-based courseware that we have developed to teach WSNs and their security issues from a practical point of view. Aiming at *experimental learning* which enable effective student learning, we have carefully designed nine course modules by using materials published in representative conference papers and journal articles in the area of WSNs. The selected topics focus on practical aspects of sensor networks. Representative hardware modules from *CrossBow* are used to illustrate important concepts.

Our designed modules avoid complex theories and focus on the introduction of basic principles in WSNs. Specifically, we present nine course modules: *Introduction to Wireless Sensor Networks*, *TinyOS*, *Energy Management*, *Radio and Medium Access Control*, *Wireless Link Estimation*, *Data Collection and Dissemination*, *Security of Wireless Sensor Networks*, *Localization and Secure Localization*, and *Aggregation and Secure Aggregation*. These course modules cover fundamental topics in the area of wireless sensor networks. Distinct from existing WSN textbooks, the protocols presented in our course modules are accompanied by example open source TinyOS implementations. Therefore, students have the opportunities to see a detailed implementation after they learn the concepts from lectures. Based on our teaching experiences, this style of lecturing can significantly facilitate students' understanding of lectured subjects and help them accumulate hands-on experience in building a wireless sensor network.

Integrated in the design of wireless sensor networks are basic principles from traditional Computer Science subjects such as operating systems and computer networks. The WSN course modules that we have developed, therefore, may also be adopted in teaching traditional Computer Science subjects. Studying the open source TinyOS example programs included in the WSN modules not only help students to understand how WSNs operate, but also further enhance their understanding of fundamental concepts in operating systems and computer networks.

We introduce *XubunTOS*, which uses a live Linux CD to greatly simplify the installation of TinyOS and to ease the development environment setup for WSNs. We then present a practical project for students to develop a *multihop data collection tree protocol*. This project can facilitate students' understanding of both sensor hardware and WSN software at various layers of the network protocol stacks. Our project description provides a step-by-step tutorial for students to follow. We have put all the course modules and project in the class website: http://galaxy.cs.lamar.edu/~bsun/wsn/wsn.html. When these course modules are adopted by instructors in other universities, we recommend that those instructors check relevant websites and make corresponding revisions because of constant progress of WSN research.

The development of these modules greatly facilitates students' interests in WSN research. Four students took a master thesis option after they completed this course. This activity can further foster both graduate and undergraduate research environment, encourage student collaboration, and enhance their confidence for future career.

Concerning future extension of this project, first, we plan to further revise the course materials and, if necessary, add more contents. We would also like to make our course modules more practical and comprehensive in the future. This requires us to keep track of the latest research development in wireless sensor networks and continue to integrate relevant topics into the course modules. Further revisions of some interrelated topics, for example, energy-conserving driver design for WSNs, WSN network protocol debugging, WSN data analysis, etc., are inherently complex and depend on more research work from the WSN community. We will add relevant modules when the WSN research community has reached an agreement on certain topics.

Second, we plan to make the assignments and projects more practical and cover more topics. The project presented in this paper only focuses on the network layer of WSNs. We plan to design a practical project for each module to facilitate students' understanding.
Third, we will perform formal evaluation of the course modules when larger data sets are available. We plan to

use the developed modules in future offering of this course to collect more data. We will also obtain students evaluation data from other instructors who are willing to use our proposed modules. When we have enough data, we will apply relevant statistical packages like SPSS [20] to perform formal analysis of the collected data.

Acknowledgements

This work was supported in part by grants from US NSF (DUE-0633445 and DUE-0633469) and the Institute of Space Systems Operations (ISSO).

References

[1] ABET, http://www.abet.org.
[2] ACM SENSYS, from http://sensys.acm.org/.
[3] CC2420 Datasheet, http://focus.ti.com/docs/prod/folders/print/cc2420.html.
[4] CC1000 Datasheet, http://focus.ti.com/docs/prod/folders/print/cc1000.html.
[5] Collection Tree Protocol, http://www.tinyos.net/tinyos-2.x/doc/html/tep123.html.
[6] Couto, D. D., Aguayo, D., Bicket, J., and Morris, R., A High-Throughput Path Metric for Multihop Wireless Routing, *Proc. of the ACM MobiCom*, Sept. 2003, San Diego, CA, 134-146.
[7] Denning, P., Great Principles in Computing Curricula, *Proc. of the 35th SIGCSE*, March 2004, Norfolk, VA, 336-341.
[8] Fonseca, R., Gnawali, O., Jamieson, K., and Levis, P., Four Bit Wireless Link Estimation, *Proc. of the HotNets VI*, Nov. 2007, Atlanta, GA.
[9] Gay, D., Levis, P., and Culler, D., Software Design Patterns for TinyOS, *ACM Transactions on Embedded Computing Systems*, *6*(4), 40-49.
[10] Karlof, C., Sastry, N., and Wagner, D., TinySec: a Link Layer Security Architecture for Wireless Sensor Networks, *Proc. of ACM SenSys*, Nov. 2004, Baltimore, MD, 162-175.
[11] Krishnamachari, B., Estrin, D., and Wicker, S., The Impact of Data Aggregation in Wireless Sensor Networks, *Proc. of 22nd International Conference on Distributed Computing Systems Workshops*, Toronto, Canada, July 2007, 575-578.
[12] Levis, P., T2: A Second Generation OS For Embedded Sensor Networks, *Technical Report TKN-05-007, Telecommunication Networks Group*, Technische Universitat Berlin.
[13] Levis, P., Patel, N., Culler, D., and Shenker, S., Trickle: a Self-Regulating Algorithm for code Propagation and Maintenance in Wireless Sensor Networks, *Proc. of the USENIX NSDI*, March 2004, San Francisco, CA, 15-28.
[14] Liu, D., and Ning, P., *Security for Wireless Sensor Networks* (New York, NY: Springer, 2006).
[15] Liu, D., Ning, P., and Du, W., Attack-Resistant Location Estimation in Sensor Networks, *Proc. of the Fourth IPSN*, April 2005, Los Angeles, CA, 99-106.
[16] Madden, S., Franklin, M., Hellerstein, J., and Hong, W., TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks, *ACM SIGOPS Operating Systems Review*, Vol. 36, Is: SI, 131-146.
[17] Perrig, A., Stankovic, J., and Wagner, D., Security in Wireless Sensor Networks, *Communications of the ACM*, *47*(6), 53-57.
[18] Polastre, J., Hill, J., and Culler, D., Versatile Low Power Media Access for Wireless Sensor Networks, *Proc. of ACM SenSys*, Baltimore, MD, Nov. 2004, 95-107.
[19] Raghavendra, C. S., Sivalingam, K. M., and Znati, T., *Wireless Sensor Networks* (New York, NY: Springer, 2006).
[20] SPSS, http://www.spss.com.
[21] Stoleru, R., He, T., and Stankovic, J. A., Range-Free Localization, in *Secure Localization and Time Synchronization for Wireless Sensor and Ad Hoc Networks, Advances in Information Security Series, Vol. 30* (New York, NY: Springer, 2006).
[22] Sun, B., Li, C. C., Wu, K., and Xiao, Y., A Lightweight Secure Protocol for Wireless Sensor Networks, *Elsevier Computer Communications Journal Special Issue on Wireless Sensor Networks: Performance, Reliability, Security, and Beyond*, 2006, 2556-2568.
[23] TinyOS, http://www.tinyos.net.
[24] TinyOS Documentation, http://www.tinyos.net/tinyos-2.x/doc/
[25] TinyOS Tutorial, http://docs.tinyos.net/index.php/TinyOS_Tutorials.
[26] TinyOS V2, http://tinyos.cvs.sourceforge.net/tinyos/tinyos-2.x/.
[27] Tran, S. P. M. and Yang, T. A., Evaluations of Target Tracking in Wireless Sensor Networks, *Proc. of the 37th SIGCSE technical symposium on Computer science education*, 2006, 97-101.
[28] USENIX NSDI, http://www.usenix.org/event/byname/nsdi.html.
[29] USENIX OSDI, http://www.usenix.org/event/bytopic/osdi.html.
[30] Xubuntu, http://www.xubuntu.org.
[31] CrossBow, http://www.xbow.com.
[32] Zhao, F., and Guibas, L., Wireless Sensor Networks: An Information Processing Approach, The Morgan Kaufmann Series in Networking (San Francisco, CA: Morgan Kaufmann, 2004).
[33] Zhou, G., He T., Krishnamurthy, S., and Stankovic, J. A., Models and Solutions for Radio Irregularity in Wireless Sensor Networks, *ACM Transactions on Sensor Networks*, *2*(2), 221-262.