In Proc. of 12. Sensor Kongress, Nürnberg, Germany, May 2005 AMA.

A Wireless Sensor Network Testbed Supporting Controlled In-building Experiments*

Suet-Fei Li, Vlado Handziski, Andreas Köpke, Martin Kubisch, Adam Wolisz
Telecommunication Networks Group, Technische Universitat Berlin
Sekr. FT 5, Einsteinufer 25
10587 Berlin, Germany
{li, handzisk, koepke, kubisch, wolisz}@tkn.tu-berlin.de

Abstract

Automatic control and monitoring of the building environment is an important application of Wireless Sensor Networks (WSN). We have deployed a large scale WSN consists of 100 nodes and covers 50 rooms in an office building that in currently in-use. The highly realistic experimental setting is rather hostile to the kind of low energy, low cost RF communication WSN relies on. Since we want to experiment with different protocol implementations and evaluate the performance of the hardware and the software, there is a great need for in-network monitoring and debugging support. We argue that such support is best accomplished by a separate reliable wired infrastructure that performs the following duties: supervise the health status of the nodes, reset them if they are in dead-lock states, remotely program the nodes and carry out system diagnosis and performance analysis etc., all without disturbing the wireless data channel. In this paper, we present the physical setup of the testbed, the sensor node hardware and software architecture, the independent USB based infrastructure for debugging, monitoring, control and management, the graphical user interface, and the protocol implemented.

1. Introduction

Automatic control and monitoring of the building environment is an important application of Wireless Sensor Networks (WSN). Modern “smart buildings” need a large network of sensors and actuators to measure and control environmental parameters such as temperature, light intensity, noise level to conserve energy and improve the overall comfort level for the occupants. Additional tasks include security, abnormal conditions detection and emergency situations handlings. Wired installations of sensor networks are already widely deployed. Compared to their hard-wired counterpart, wireless sensor networks offer several significant advantages. According to ARC market research firm [1], the cost of installing a wireless network is roughly 10% that of a hard-wired design. The self-configuring nature of the WSN also means significantly less setup time and a more flexible installation.

Building automation has been widely considered a viable application for WSN and there are commercial products specifically targeting this application [2]. However, there are few published works on how to deploy such a large scale WSN in an office building that is currently in use. WSN is still an emerging technology and is far from mature. In such an early development stage of the technology, one cannot simply install the nodes in their appropriate locations, and count on them working properly without any monitoring and intervention, as would be the case when it has reached maturity. At the “testbed” stage, often an independent infrastructure for debugging, monitoring, control and management is needed to test the proper functioning of the WSN.

The office building environment is extremely hostile to the kind of low energy, low cost RF communication WSN relies on. The hostile factors include the presence of metallic walling and large quantities of electronic equipment, the interference from the WLAN infrastructure and the constant movement of people, etc. Under such realistic and “hostile” settings, we want controlled experiments to evaluate the performance of the hardware and the software. The need for such experiment calls for the installation of a separate wired infrastructure that performs the following duties: supervise the health status of the nodes, reset them if they

* This work has been partially sponsored by the European Commission under the contract IST-2001-34734 – Energy-efficient sensor networks (EYES)

are in dead-lock states, remotely program the nodes and carry out system diagnosis and performance analysis etc., all without disturbing the wireless data channel.

In this paper, we present in detail our WSN testbed deployment for supporting controlled in-building experiments. In particular, we focus on the implementation of the very important but often overlooked wired infrastructure. The rest of the paper is organized as follows: Section 2 illustrates the physical setup of the testbed and describes briefly the sensor node hardware and software. Section 3 presents the independent infrastructure for debugging, monitoring, control and management. Section 4 describes the graphical user interface. Section 5 discusses the protocols implemented in the testbed, namely Content-based publish/subscribe, Consensus protocol and a MAC protocol to prolong sensor lifetime. Section 6 concludes the paper.

2. System Description

Testbed setup

The testbed is deployed in a realistic setting provided by the office building of the Telecommunication Networks Group (TKN) on the TU Berlin campus. The instrumented area covers more then 50 rooms, spread over four floors, using more then 100 nodes. Figure 1 shows the deployment layout for the fourth floor of the building. All the sensor nodes are connected to the wired infrastructure (not included in the figure) via a USB bus interface. In the initial phase of the deployment, the tested only supports in-building monitoring of temperature and light intensity. In the future, actuators will be added and a control algorithm will be developed to achieve full building automation.

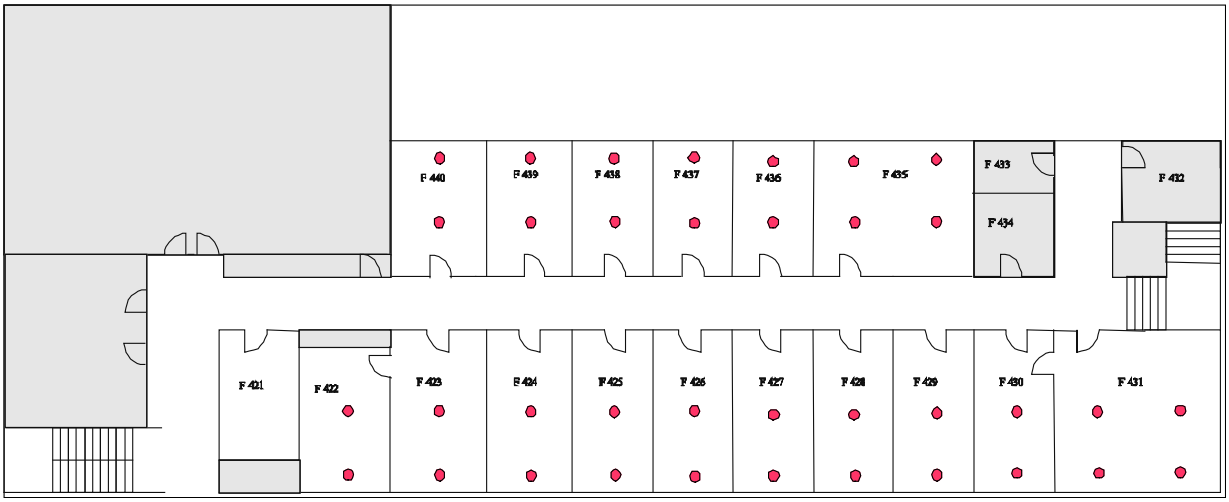

Figure 1. Location of the testbed nodes on the 4th floor of the FT building

Sensor node hardware

We are using the eyesIFXv2 sensor nodes (see Figure 2) developed by Infineon Technologies AG, featuring the Texas Instruments MSP430 microcontroller and the Infineon TDA 5250 radio transceiver. The MSP430 family of microcontrollers is specifically designed for ultra-low-power applications. There are 48 KB of Flash and 10 KB of RAM on board, thus supporting applications with significantly higher memory requirements.

The TDA 5250 radio transceiver supports ASK/FSK modulation with speeds up to 64 kbps. It has a good data valid detection capability and provides robust communication, making it a good choice for demanding RF environments. The sensing capability is provided by three on-board sensors: the internal MSP430 temperature sensor, an external high-precision temperature sensor and a light sensor.

External data interfacing is possible through a USB-slave interface, which enables programming of the microcontroller and in-circuit debugging. This is the interface used to implement the supporting backbone infrastructure. To display status information, an array of 4 LEDs is available. In addition, there is an expansion connector that allows connection of secondary boards with additional analog/digital sensors or actuators. In the future, the expansion connector will be used to add actuation functionalities to the WSN.

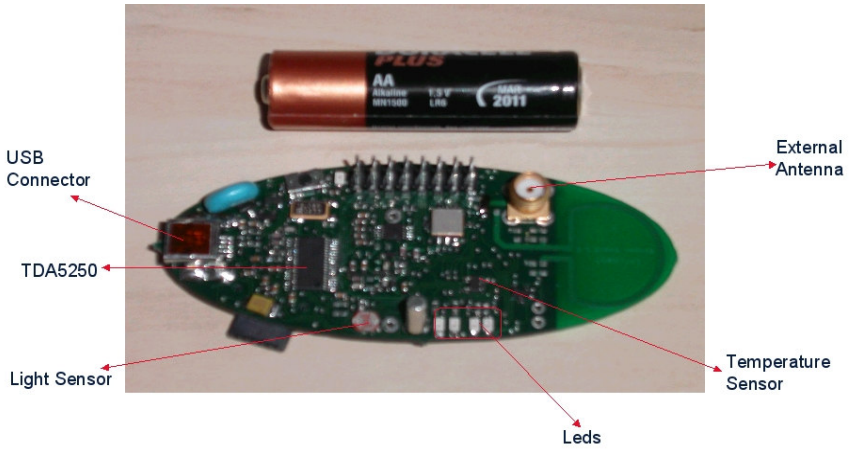

Figure 2. The eyesIFXv2 node by Infineon Technologies

Sensor node software

The operating system (OS) that runs on the eyesIFXv2 is TinyOS [3], the most popular OS for WSNs featuring a component-based model with strong compile-time optimization and race-condition detection. We have ported TinyOS to the TI MSP430 family of microcontrollers, enabling full use of the existing TinyOS services on eyesIFXv2. A hardware abstraction layer was developed to abstract the capabilities of the hardware and translate them into higher-level OS services available to the application developers [4]. Similarly, we have abstracted the capabilities of the Infineon TDA 5250 radio transceiver and integrated them into a flexible execution environment that enables easy testing of different MAC and Link Layer protocols.

3. Infrastructure Support for Debugging, Monitoring, Control and Management

The necessity to include an infrastructure that is completely independent of the WSN wireless data channel has been motivated in the introduction section of this paper. Basically there are two major roles for such infrastructure: the first is the remote power cycling/reset and programming of the nodes; the second is system states monitoring, debugging and diagnosis to validate the correct operations of the WSN. Figure 3 depicts the topology of the infrastructure. Up to four eyesIFXv2 nodes are connected to a USB hub through the on-board USB interface. A USB interface is used due to its high bandwidth, communication characteristics and well-supported plug-and-play capabilities. The USB hub is present to perform power cycling operations because the standard USB controller does not support power down/up of a specific device.

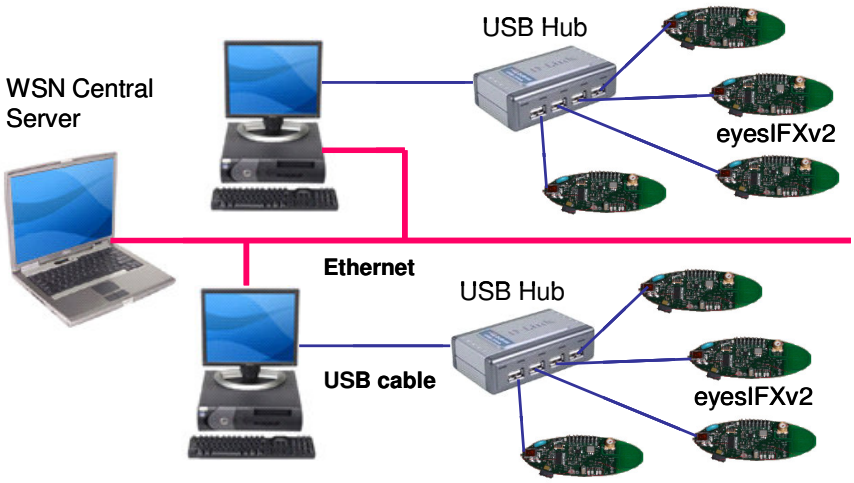

Figure 3. USB tree infrastructure for system monitoring, control and management.

The hub is then connected to a PC in the existing building LAN infrastructure, which then communicates to the global WSN server via Ethernet. The server collects data from all the eyesIFXv2 nodes, keeps the overall system status and performs the necessary control and analysis.

In the current version of the testbed deployment, due to time and resource constraints, PCs in the existing infrastructures are used as USB relays. Ideally we would want the testbed to be stand-alone and completely separate from any existing infrastructure. In the next phase of the deployment, we will replace these PC with custom USB to LAN or WLAN bridges.

The capability to remotely program the nodes enables us to update the application on the fly and experiment with different protocols. Hence we can implement different applications on different nodes based on run time information.

System states monitoring, debugging and validation

To validate the correct functioning of our initial deployment, it is desirable and even necessary to pass debug and monitoring messages in the network. These types of messages should not be mixed with the regular data packets such that they may interfere with normal operations. By moving them to the wired infrastructure channel, we can perform system state checks and background monitoring without burdening the wireless network.

With only the wireless channel, it is quite difficult to gauge the performance of the WSN as well as debug the problems. The WSN system acts like a black box, and we can only deduce the internal states by analyzing output traces. With the aid of the infrastructure, we are able to “look inside” and have access to the system internal states. Furthermore, the reliable USB based infrastructure enables us to establish a “ground-truth” against which the application output can be compared and consequently system performance can be measured.

4. Graphical User Interface (GUI)

A GUI is provided to visualize both the wireless data channel communication and the infrastructure backbone activities. The serial forwarder application, which provides a relay between the eyesIFXv2 USB interface and the TCP/IP socket connection, is used as a gateway between the WSN and the Ethernet. In the case of wireless data channel communication, data messages are collected by the base node that is connected to a PC control station and then exported. In the case of backbone infrastructure, each node starts a serial forwarder on the corresponding relay PC, which is then piped through SSH tunnels to the central

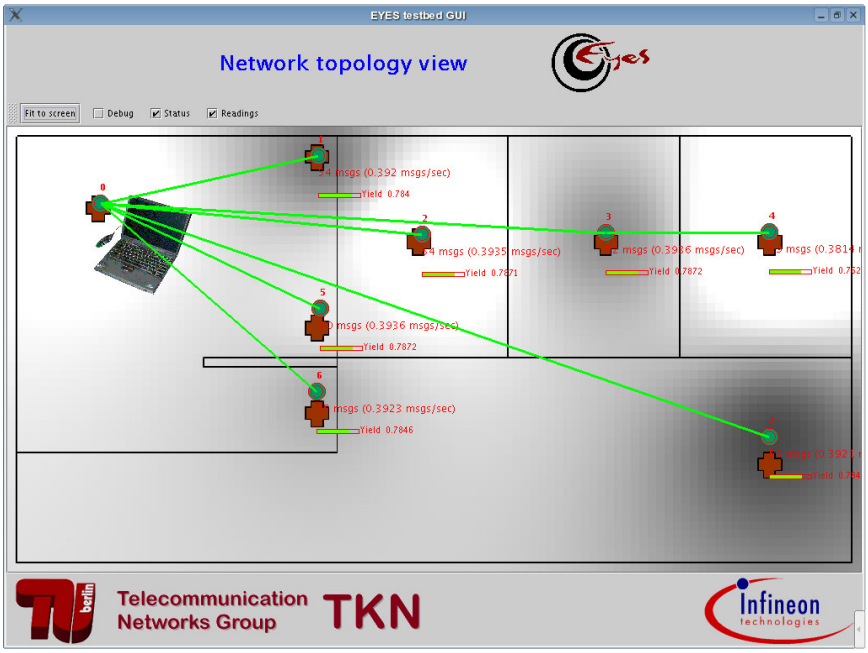

Figure 4 Java based graphical user interface for visualization and control of WSN

server, which gathers and processes all the data streams.

By using the serial forwarder gateway, data from the WSN can be made available both locally and (over the Internet) to any remote site. The GUI (Figure 4) serves two purposes. Firstly, it provides a graphical interface for the user to explore the various services offered by the WSN such as temperature and light intensity queries of the building. Secondly, it ensures the validation of the proper operation of the network via augmented visualization of the network topology, the sensor readings, the node health status and the service protocol state. It also provides a simple interface for power cycling and remote programming of specified nodes.

5. Protocol development

A major goal of testbed deployment is to aid the development and validation of novel application layer, network layer and MAC algorithms. We have implemented the content-based publish/subscribe, the consensus protocol and a MAC protocol to prolong sensor lifetime.

Content-based publish/subscribe [5]

Content-based publish/subscribe (CBPS) is a data-centric communication paradigm. A CBPS system consists of publishers, a set of nodes which produce information to the network; and subscribers, another set of nodes which consume this information in the network based on a set of subscription criteria [6]. The CBPS middleware implemented in the testbed has the following technical capabilities: data-centric routing, in-network filtering of data and event-based notification with identity and synchronization decoupling. The adaptive routing protocol is robust and able to deliver services in the face of transient failures of the nodes or the communication paths.

The correct operation of CBPS will be directly validated via the graphical user interface. The GUI's network topology view will be augmented with information about the internal state of the semantic addressing protocol, which will enable good insight in the inner-workings of the algorithm. The user will interact with the WSN services via an interface that enables graphical specification of the subscription filters. After issuing the subscription request, the application will pass on the request to the service that will spread the subscription throughout the network, thereby establishing a routing structure for the notifications. On the reception of a matching subscription message, the publishers issue notifications that are routed back to the subscriber). The received data is then returned to the application and is adequately visualized.

Consensus [7]

Consensus provides the important basic functionality of a distributed system, e.g. a sensor network. It enables coherent functions of the actors in a sensor network (door locks, blinds, lighting, heating etc.) by enabling a common view on the action that should be performed. Furthermore, it is at the heart of any group communication protocol that enables actors to rely on certain semantics of the communication. It is also the enabling technology for in-network reprogramming of all nodes, either partially or completely. In the latter case, it becomes vitally important if basic protocols like the MAC protocol are exchanged. If a single node fails to exchange the MAC protocol, it is henceforth unreachable. Consensus is necessary to ensure that all nodes can switch to the new protocol, or that none does until additional measures are taken.

The correct operation of the protocol can be directly verified using the graphical user interface. In the first stage, the new firmware will be distributed in the network; then the two-phase commit protocol runs to ensure that the firmware can be run everywhere. The phases of the two-phase commit protocol will be visualized, together with the state of each node. After all nodes switch to the new firmware, all of them must reappear on the GUI. There is no restriction on the distance of the nodes to the root node.

MAC protocol to prolong sensor lifetime

One of the main constrains in WSNs is the power consumption of the nodes. In real deployments a sensor will not have a wired connection, thus leaving its power supply to batteries or simple harvesting techniques. Hence, the sensor to has to work as efficiently as possible with its power reserve, i.e., reduce the energy usage. A major power sink in wireless sensors is the radio transceiver. Switching off the radio transceiver can reduce the power consumption by a factor of 1000 [8], i.e., the sensor can significantly reduce the power drainage to prolong its lifetime. While many popular MAC protocols rely on idle listening (carrier sense multiple access -CSMA), it is a waste of energy in sensor networks with low duty cycles.

A better approach is to use timed protocols (time division multiple access - TDMA), thus allow the radio transceiver to switch off for the duration when no data must be exchanged. But implementing a timed protocol has two constraints: 1. At least between the neighboring sensors, time synchronization must be achieved. 2. Collisions on the wireless medium between sensors in a larger area must be avoided. As there are already higher layer protocols to provide sensor synchronization, the second problem is more severe. In wireless networks the distance at which a sensor can disturbed an ongoing reception is larger than the distance within which two sensors can communicate; it requires a protocol that takes care of at least a two hop protection around a receiver. We have developed the JamTDMA protocol to provide such functionality. In addition, JamTDMA protocol is able to provide periodic times assigned to only two sensors, thus reduces power consumption as little further information exchange is necessary. With the wireless sensor testbed we are able to test and validate the JamTDMA protocol and determine the power reduction achieved.

6. Conclusion

Wireless sensor networks (WSN) research is currently a hot topic in both academia and industry. New application scenarios, MAC, routing and physical layer algorithms are being developed by the research community. The ultimate test for the validity of these research activities is to implement the various scenarios and algorithms on a testbed and deploy it in a realistic setting. In this paper, we have presented a large-scale WSN deployment in an office building that is currently in-use. Our experience has shown that major “real-life” deployment is far from trivial and we have encountered many problems that do not occur in smaller test scenarios. Therefore, sufficient in-network monitoring and debugging support is crucial for the success for testbed deployment. Such supports should also be incorporated into the sensor node software and hardware architectures.

Acknowleadgement

We would like to thank Infineon Technologies AG for providing us the the eyesIFXv2 nodes used in the testbed as well as other supports.

References

1. ARC Advisory Group, **http://www.arcweb.com/**
2. The Millennial Net Sensor Networking Platform for Building Automation, http://www.millennial.net/content.cfm?section=2.06
3. TinyOS, http://www.**tinyos**.net/
4. Vlado Handziski, Joseph Polastre, Jan-Hinrich Hauer, Cory Sharp, Adam Wolisz, and David Culler, "Flexible Hardware Abstraction for Wireless Sensor Networks", In *Proc. of 2nd European Workshop on Wireless Sensor Networks (EWSN 2005)*, Istanbul, Turkey, February 2005
5. V. Handziski, A. Köpke, H. Karl, C. Frank, and W. Drytkiewicz, "Improving the Energy Efficiency of Directed Diffusion Using Passive Clustering", In H. Karl, A. Willig, and A. Wolisz, editors, *Proc. of 1st European Workshop on Wireless Sensor Networks (EWSN)*, Volume 2920 of *LNCS*, Berlin, Germany, January 2004 Springer.
6. Improving Reliability of Data Dissemination in Sensor Networks Through Epidemic Algorithms, http://www.ug.bcc.bilkent.edu.tr/~ozang/wsn/analysis.htm
7. A. Köpke, H.Karl, and A. Wolisz, "Consensus using aggregation - a wireless sensor network specific solution", Technical Report TKN-04-004, Telecommunication Networks Group, Technische Universität Berlin, April 2004.
8. Laura Marie Feeney and Martin Nilsson, Investigating the Energy Consumption of a Wireless Network Interface in an Ad Hoc Networking Environment, *INFOCOM 2001*, Anchorage, USA, April 2001.