Computer Science • 13 (3) 2012 http://dx.doi.org/10.7494/csci.2012.13.3.101

Radosław Kapłoniak
Łukasz Kwiatkowski
Tomasz Szydło

ENVIRONMENT EMULATION FOR WSN TESTBED

Abstract *The development of applications for wireless sensor networks is a challenging task. For this reason, several testbed platforms have been created. They simplify the manageability of nodes by offering easy ways of programming and debugging sensor nodes. These platforms, sometimes composed of dozens of sensors, provide a convenient way for carrying out research on medium access control and data exchange between nodes. In this article, we propose the extension of the WSN testbed, which could be used for evaluating and testing the functionality of sensor networks applications by emulating a real-world environment.*

Keywords sensor networks, testbed platform, emulation

1. Introduction

The writing of applications for wireless sensor networks(WSN) is a very demanding task because it requires programming a number of nodes and then testing them using several use case scenarios in order to verify their functionality. Establishing a sensor network for testing purposes in the real environment may cause several problems that have to be solved. Programming several sensor nodes is the time consuming because every sensor has to be connected to the programmer. Some technologies allow the use of over-the-air programming technique, but usually this is not a robust solution. Debugging large number of nodes is also challenging task because all the debugging data has to be collected and analysed. Sending this data wirelessly may influence the communication protocol and introduce disturbances to the radio spectrum. Additionally there is a significant increase in the overhead relating to the maintenance of the testbed equipment, not to mention securing equipment against the threat of damage or theft.

To overcome these limitations there are several WSN testbed platforms [5, 8] that simplifies the node programming process and debugging. In these testbeds, sensor nodes are deployed in a particular place and infrastructure services are provided. These services and additional hardware infrastructure are used for the manageability of nodes such as programming and debugging realized out-of-band. In typical testing facilities it is difficult to repeat some of the tests, because every time environmental conditions would be different. This concerns not only wireless spectrum and its influence on medium access control protocols but also on the values red from sensors such as temperature or humidity.

Wireless sensor networks have increasing number of applicability fields. They are used mainly for environmental sensing, industrial monitoring, on-site tracking, assisted living and controlling. One of the usage domain of WSN that is gaining momentum is Internet of Things [1]. Testing and evaluating such solutions require combining wireless sensor nodes with real-world objects. In the paper we propose a concept of environment emulation where there will be the possibility to combine real wireless sensor nodes with virtual objects that may interact among themselves. In our concept such solutions might be evaluated if only the model of real objects or environment is provided. Environment emulation is limited to the interaction between sensor nodes and the information gathered by set of sensors installed on the nodes and does not influence wireless connectivity between nodes.

This paper shows the preliminary results of developing components that might be used for environment emulation in the WSN testbed. The structure of the paper is as follows: Section 2 shows motivating scenario that has driven our research. Section 3 discusses other testbeds and then Section 4 shows the concept of environment emulation. Section 5 describes the architecture of said system while in Section 6 implementation details are presented. Finally Section 7 presents the summary and further research.

2. Motivating scenarios

Sensor networks have still growing number of applicability fields. Most obvious areas of usage are: (i)Smart homes; (ii) Ambient assisted living; (iii) Heating, Ventilation, Air Conditioning (HVAC). A typical use case that encompassed all these applicability domains might look like:

Bob is coming back after work to his intelligent home. He opens the door, presses the light switch and goes to the wardrobe to change clothes. Then he goes to the kitchen to eat supper, but in the meantime he increases the temperature in the home using a dial knob located in the corridor. He eats supper and then he moves to the living room and watches a movie on TV sitting comfortably on a couch. When the movie finishes, he goes to the bedroom to sleep.

During that time, when Bob is in his home, a wireless sensor network deployed in his home can control all the available appliances. The lights in the house might be switched off at a predetermined time and data range, or after occupants have left the room. The brightness of the lights might be controlled accordingly to the level of available ambient light. The temperature in the home might be decreased once Bob is sleeping and can automatically adjust to his working hours in order to increase it couple of minutes before he is return from work.

The development of wireless sensor network applications requires continual testing of it in the real world environment and scenarios. It would be desirable to have a testing platform where such scenarios could be defined in order to rerun it several times to verify functionality during development. In this paper, we would like to introduce the concept of an extension to the WSN testbed platform that enables the possibility of defining such scenarios.

3. Related work

Several WSN testbeds that have been designed differ in scope, architecture and hardware features. All of them provide some system services that simplify programming and debugging sensor nodes. Most of the testbeds addresses only single domain of devices, enabling possibility to use only a particular type of devices, but the key factor that differentiates the testing platform is its architecture. Smaller platforms such as MIRAGE [4] or Vinelab follows two-tier structure that consist of a server tier and directly attached sensor nodes. Larger testing facilities follow a three-tier architecture where additional intermediate layer is introduced. That layer consists of super node devices that intermediate in the communication between sensor nodes and a server. Examples of such testbeds are MoteLab [10], TutorNet, TWIST [6] and others. Existing testbeds vary in the hardware features that they provide. The key factor is underlying microcontroller type (e.g. MSP430 or ATmega), OS (e.g. TinyOS or Contiki) and medium access (e.g. 802.15.4). Secondary factor is type of sensor nodes (e.g. TelosB or Zigduino or MicaZ) as they differ mostly in microcontrollers and OS types and eventually simple temperature, humidity and light sensors.

Some testbeds allow for node level energy measurement and node mobility. In such testbeds (e.g. WISEBED [3]), nodes are attached to movable objects with controllable trajectory. As of time of writing, there is only one testbed w-iLab.t [2] that supports the emulation of sensor events by attaching digital-to-analog converters to the sensor nodes. The testbed provides additional modules, which trigger sensor readings. This allows replaying of sensor events obtained from real-world environment, but does not allow the inclusion of scenarios with virtual objects or environments. A full survey on testbed facilities for WSN can be found in [5, 8].

4. Concept of environment emulation

In the typical sensor node, the interaction with real environment is done by input and output pins of the sensor node. Environmental information is acquired by means of sensors, while changes to the environment are performed by actuators. Environmental sensors convert measured physical values to a number value that is read by the sensor node. Actuators connected to the sensor node convert electrical values to various actions e.g. movement or light controlling. There are several communication protocols between environmental sensors/actuators and sensor nodes such as I2C, RS485, etc. Nevertheless, the most commonly used are simple analog-to-digital(ADC) and digital-to-analog (DAC) converters. In both cases, information processed by the sensor node is the same.

The value acquired from an environmental sensor does not provide semantic information of what has been measured – this information is hardcoded in a program that is deployed in the sensor node. Accordingly, the semantic meaning of the value that drives an action is hardcoded in a program executed on a sensor node. This led us to the conclusion that any environmental sensor can be substituted by the electronic module providing any values by means of voltage level that can be read by the sensor node. Similarly, any effectors can be substituted by the electronic module that can read value provided by the sensor node. Typically, sensor network is designed for specific purpose, thus measured values are correlated to each other because sensor nodes are located in the same environment. This is depicted in the Fig. 1a.

In our concept, we propose a solution where sensor network testbed would be located in a fixed and secured place, while real environment would be emulated by additional infrastructure. To emulate a real environment, a model of that environment and electronic modules wired to the sensor nodes have to be provided. This concept is depicted in Fig. 1b. Electronic modules might be used for different natural environment modelling because their aim is to read and write digital values set by the model of the natural environment.

The introduced concept can also be used for emulating real objects and providing information about its state in the same way. In that case, model of natural environment can be substituted by any model of virtually cooperative objects.

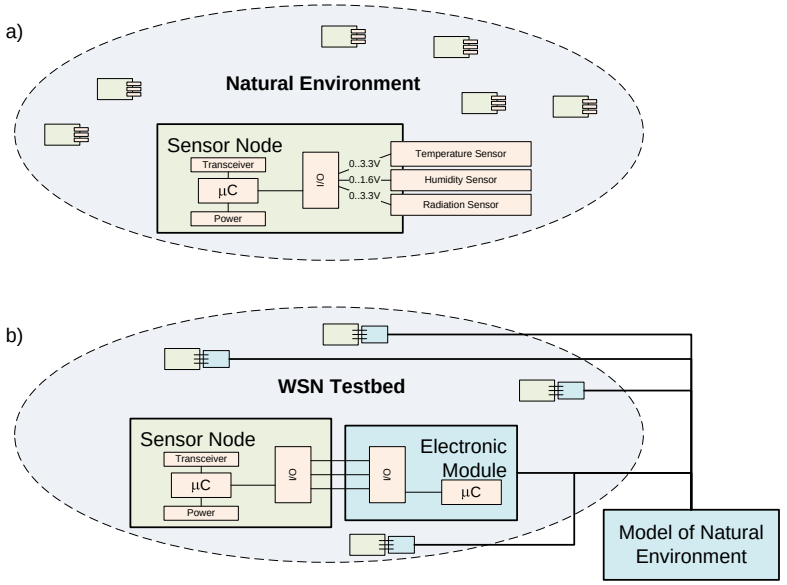

Figure 1. Concept of environment emulation.

5. Environment emulation architecture for WSN testbed

Typical wireless sensor network testbeds are composed of sensor nodes located in a fixed place and the additional hardware infrastructure that provides manageability. Modern testbeds exhibit a three-tier architecture in which groups of sensor nodes are connected via wired connections to an intermediate gateway or super-node device as depicted in Fig. 2. This reduces the need for each individual sensor node to provide an additional network interface card for its manageability. Instead of that, sensor nodes might be connected via a cheap USB cables to the super-nodes that are then connected via network connection to the server tier. Additional advantage of such an infrastructure is that super-nodes, which have usually comparatively large computing power, can be also used for experiments e.g. as a gateway for wireless sensor network to the Internet.

System is composed of the three elements: (i)Environment Emulation Server with a pluggable emulation modules, (ii)distributed software executed on a server and super nodes, and (iii)hardware modules connected to super-nodes and wired to sensor nodes. The main element of the system is a server with dedicated software. Emulation software enables possibility to implement different functional emulation modules for various applications. Server is connected to the existing network infrastructure and communicates with software installed on super-nodes. There are two electronic modules: *master* and *slave*. *Master* module is connected to the super-node and supports up to eight *slave* modules that can be connected to. Output (DAC converters) and input

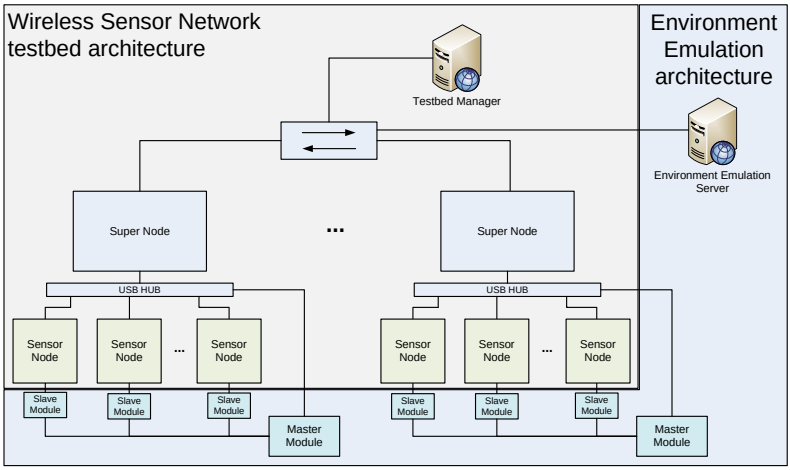

Figure 2. Architecture of WSN testbed with environment emulation tier.

(ADC converters) pins of *slave* modules are directly wired to the sensor nodes in order to read and write input and output pins as it was described in the previous section. All the electronic modules are accessible through the abstraction layer provided by the distributed software executed on the testbed infrastructure. Described elements of environment emulation system could be added to the typical wireless sensor network testbed reusing most of the existing hardware.

6. Implementation details

In the previous section architecture of environment emulation tier has been presented. Main elements of the infrastructure are the electronic modules that have to be connected to the existing elements of WSN testbed and the corresponding software components that control the environment emulation.

6.1. Hardware modules

Hardware layer consists of *master* and *slave* modules. Both are based on ARM Cortex-M3 microcontrollers supplied by STMicroelectronics. The main advantage of used chips is a rich set of SoC peripherals such as three fully hardware UARTs with DMA and IRQ support. In our prototype STM32F103RBT6 microcontrollers have been used. Chips are clocked by external 12MHz oscillator which is multiplied by internal PLL to 72MHz.

Master module communicates with super-node by USB-UART converter (FTDI232) and with *slave* module via RS485-UART converter (MAX485). The FTDI232 chip has been chosen because of availability of OS drivers for many platforms (including Windows and Linux). The RS485 standard is highly recommended

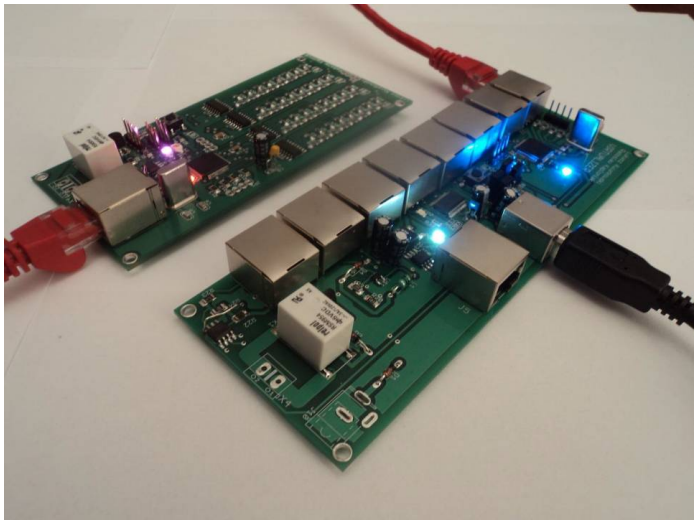

Figure 3. Prototype of *master* and *slave* modules.

for industrial installations as it can be used over long distances and in electrically noisy environments due to differential lines. *Master* module receives AT commands from super-node, queues them into one of eight queues (each for one *slave* module) and successively transmit commands to destination module (using round-robin algorithm). *Master* module receives also results and events from *slave* modules, buffers them and finally transmits to the super-node. The entire communication is based on hardware interrupts allowing for the effective use of CPU. *Master* board contains also an LDO regulator and a step-up converter that are used to supply power to the connected *slave* modules. This module can also reset super-node using built-in relay in the case of testbed platform maintenance. The PCB contains also in-system programming for easy maintenance.

Slave modules have the same microcontroller as *master* module. It uses internal ADC (12-bit) to read values from sensor nodes and external DAC (8-bit) which can generate voltages between 0-3.3V or 0-5V (selectable). The module can reset sensor node, which can be useful while remote programming sensor nodes.

Both modules, as presented in Fig. 3, contain pair of status LEDs used for debugging and controlling purposes. Modules are connected via UTP cable with a standard RJ45 plug, so there is no need to use a custom cables. One cable contains RS485 lane and three supply lanes (3.3V, 5V, 9V).

6.2. Software

Environment emulation infrastructure is managed and controlled by the prototype software that has pluggable architecture and is implemented using OSGi [9]. The system is designed in an event-driven way; thus, the information about changes observed

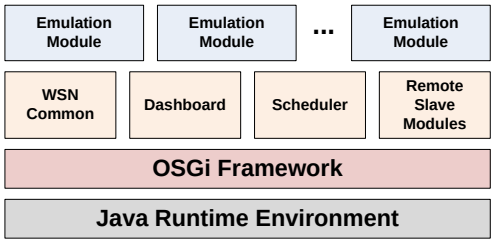

Figure 4. Layered architecture of software components.

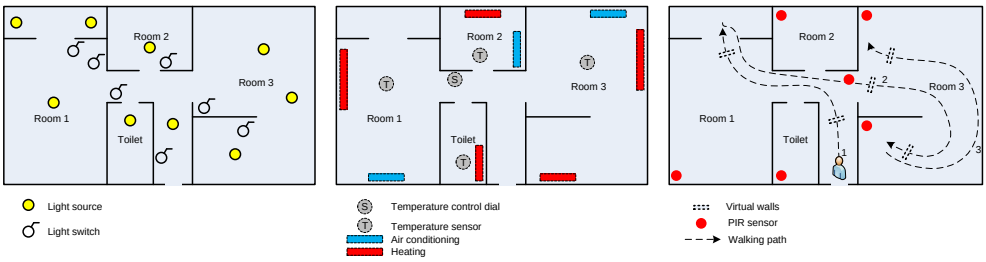

Figure 5. Environment emulation modules.

by an electronic module is published by the Event Admin service available in OSGi. The EventAdmin service is part of the OSGi Compendium Specification and is intended to provide publish/subscribe functionality. The system as depicted in Fig. 4 is composed of the various software modules:

- WSN Common – contains all the necessary interfaces that are common for the system;
- Scheduler – is a main component for a system that discovers registered emulation modules and maintains the test cases;
- Dashboard – is a common GUI frame that any simulation module can provide a graphical representation of its state;
- Remote *Slave* Modules – proxy for remote *slave* modules that are wired to the sensor nodes;
- Emulation modules – software components implementing emulation modules.

For remote communications between the server and the electronic modules, ICE [7] middleware has been used. It simplifies remote communication providing object-oriented remote procedure call. ICE provides two-way communication, so the system supports action-reaction model for virtualised environments based on asynchronous events generated by a hardware layer. The user is able to define his own implementation of callback method for each event (such as rising edge on input). All of the electronic *slave* modules are accessible as a service registered in OSGi environment.

Listing 1: Example testing scenario

```
[t=100]
Dashboard.println("Time=100");
LightModule.bindLight("KitchenL", 1, 1, 0);
LightModule.bindSwitch("KitchenS", 1, 1, 1);

[t=2000 every 2000]
Dashboard.println("Every 2000ms");
Dashboard.println(LightModule.getSwitchState("KitchenS"));
LightModule.setSwitchState("KitchenS",
        !LightModule.getSwitchState("KitchenS") );
```

Currently, there are three emulation modules implemented, as presented in Fig. 5 i.e. light sources emulation, HVAC modelling and movement simulation module. In the emulation environment, virtual light sources and virtual switches might be located in different places. The state of the virtual lights is controlled by the output of a sensor node, while pressing the virtual switch will cause raising a high state on one of the sensor node inputs. In HVAC modelling, it is possible to define a shape and an area of virtual premises and a location of emulated heaters and air-conditioners. State and the power of the heaters as well as air-conditioners are controlled by the sensor node outputs. Outputs of the virtual thermometers are provided to sensor node inputs. Evaluating solutions for home automation or any other application that requires interaction with environment would not be possible without simulation of an object movement among sensors. There is a possibility of defining paths of movement and location of virtual passive infrared sensors within virtual spaces. The information from these virtual sensors controls the state of some sensor nodes inputs.

The system is designed in a way that the testing scenario is defined in a text file as a set of pairs that consist of timestamp and a Java language set of operations to be performed at that time. An example of such a testing scenario is presented in Listing 1. Scheduler module processes the Java operations in a particular order according to the timestamps. Java code included in the testing scenario is compiled in a run-time by `Janino` library. All the emulation modules are available from testing scenario.

7. Conclusions

Wireless Sensor Network platform at the Department of Computer Science at AGH is still work in progress. In the paper, we have presented a concept and architecture of its unique features that encompass environment emulation. This functionality makes it possible to define testing scenarios reflecting typical user behaviours or an I/O patterns observed by the sensor nodes. Having defined such scenarios, it will be possible to compare different ideas, protocols and applications in the same environmental conditions.

Acknowledgements

The work reported in this paper was supported by the AGH University of Science and Technology grant no. 15.11.120.076.

References

[1] Atzori L., Iera A., Morabito G.: The internet of things: A survey. *Comput. Netw.*, 54(15):2787–2805, October 2010.

[2] Bouckaert S., Vandenberghe W., Jooris B., Moerman I., Demeester P.: The w-ilab.t testbed. In Magedanz T., Gavras A., Nguyen H.-T., Chase J. S., editors, *TRIDENTCOM*, volume 46 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 145–154. Springer, 2010.

[3] Chatzigiannakis I., Fischer S., Koninis C., Mylonas G., Pfisterer D.: Wisebed: An open large-scale wireless sensor network testbed. *Sensor Applications Experimentation and Logistics*, 29:68–87, 2010.

[4] Chun B. N., Buonadonna P., AuYoung A., Ng C., Parkes D. C., Shneidman J., Snoeren A. C., Vahdat A.: Mirage: a microeconomic resource allocation system for sensornet testbeds. In *Proceedings of the 2nd IEEE workshop on Embedded Networked Sensors*, EmNets '05, pp. 19–28, Washington, DC, USA, 2005. IEEE Computer Society.

[5] Gluhak A., Krco S., Nati M., Pfisterer D., Mitton N., Razafindralambo T.: A survey on facilities for experimental internet of things research. *Communications Magazine, IEEE*, 49(11):58–67, november 2011.

[6] Handziski V., Köpke A., Willig A., Wolisz A.: Twist: a scalable and reconfigurable testbed for wireless indoor experiments with sensor networks. In *Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality*, REALMAN '06, pp. 63–70, New York, NY, USA, 2006. ACM.

[7] Henning M.: A new approach to object-oriented middleware. *IEEE Internet Computing*, 8(1):66–75, 2004.

[8] Imran M., Said A. M., Hasbullah H.: A survey of simulators, emulators and testbeds for wireless sensor networks. In *International Symposium on Information Technology, ITSim*, volume 2, pp. 897–902, 2010.

[9] The OSGi Alliance.: OSGi service platform core specification, release 4.1. `http://www.osgi.org/Specifications`, 2007.

[10] Werner-Allen G., Swieskowski P., Welsh M.: Motelab: a wireless sensor network testbed. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, IPSN '05, Piscataway, NJ, USA, 2005. IEEE Press.

Affiliations

Radosław Kapłoniak
AGH University of Science and Technology, Faculty of Electrical Engineering, Automatics, Computer Science and Electronics, Department of Computer Science, al. A. Mickiewicza 30, 30-059 Krakow, kradosla@agh.edu.pl

Łukasz Kwiatkowski
AGH University of Science and Technology, Faculty of Electrical Engineering, Automatics, Computer Science and Electronics, Department of Computer Science, al. A. Mickiewicza 30, 30-059 Krakow, kwlukasz@agh.edu.pl

Tomasz Szydło
AGH University of Science and Technology, Faculty of Electrical Engineering, Automatics, Computer Science and Electronics, Department of Computer Science, al. A. Mickiewicza 30, 30-059 Krakow, tszydlo@agh.edu.pl

Received: 9.03.2012
Revised: 22.05.2012
Accepted: 9.07.2012