Research Article

OPCPP: An Online Plug-Configure-Play Experiment Platform for WSN

Wei Lin,1 XueHai Hong,1 Chaonong Xu,2 YongJun Xu,1 and DengChang Zhao1

1 Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China
2 Department of Computer Science and Technology, China University of Petroleum, Beijing 102249, China

Correspondence should be addressed to XueHai Hong; hxh@ict.ac.cn

Received 28 December 2012; Accepted 11 March 2013

Academic Editor: Hongsong Zhu

Copyright © 2013 Wei Lin et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Wireless sensor network (WSN) experiment platforms are used for teaching, research, and development of WSN. However, existing WSN experiment platforms generally have the following disadvantages: tedious manual operations, invasive measurement method, poor sensor self-awareness, and low resource utilization rate. To address the above problems, this paper proposes OPCPP, an online Plug-Configure-Play experiment platform for WSN. It has four prominent strongpoints: in-application programming in batch, noninvasive measurement method, sensor self-awareness, and remote operation. OPCPP has been used in eight colleges in China for teaching course of "internet of things" till now. We also develop a sensor-aware ZigBee-based smart home system prototype based on OPCPP.

1. Introduction

Recently, wireless sensor network has a great research progress because of the advances in microelectromechanical systems and wireless communication technologies. To have a deep understanding and application of WSN, WSN experiment platform is essential for development, research, and teaching of WSN. However, the existing WSN experiment platforms seldom have good usability and are mainly limited in the following aspects.

(1) Tedious Manual Operations. WSN experiments often involve multiple nodes. Therefore, when users conduct an experiment on current experiment platform, they often spend a lot of time in repeating tedious operations for every node, which results in low efficiency. For example, for having an experiment of wireless routing protocol, users have to download object code into nodes in WSN experiment platform one by one with a hardware emulator, such as CC Debugger in TI [1]. Obviously this inevitably involves repeated manual plug-and-program operations which are time consuming and tedious.

(2) Invasive Measurement Method. In existing experiment platforms, the traditional method for collecting experimental data is as follows: node transfers its experimental data to the sink node via wireless communication, and then the sink node delivers all data to PC through serial port or ethernet port for further process. Since a large number of experimental data collected via wireless channel could fail the wireless network, this method is obviously not suitable for experiments such as time synchronization experiment [2], routing algorithm experiment [3], and other experiments where a large number of experimental data has to be collected for analyzing protocol performance.

(3) Poor Sensor Self-Awareness. The existing experiment platforms generally support the experiment of several common sensors, such as temperature, humidity, and accelerometer. Although the driver development of these sensors is similar,
it is still difficult to a beginner. An easy-to-use experiment platform should have the ability of self-adaption for new sensor.

(4) Low Resource Utilization Rate. Existing platforms generally do not support remote operation. Users can use the experiment equipment only when they are in the lab. Thus, low resource utilization rate is inevitable.

In order to address the above-mentioned problems, we design OPCPP, an online Plug-Configure-Play WSN experiment platform. Contrary to the above four shortcomings of traditional WSN experiment platforms, the new experiment platform has four strongpoints as follows.

(1) In-Application Programming in Batch. Users can update the code of any node in OPCPP without hardware emulator, and they can even update codes of all nodes in batch simultaneously with our software tool. This will significantly reduce the tedious and repeated manual code downloading operations.

(2) Noninvasive Measurement Method. Two data channels, wireline or wireless, are built in OPCPP, and users can assign either of the two channels for data transmission. A typical application scenario is that the test data is transmitted via wireline while the data of protocol execution process, such as Transport Protocol Data Unit (TPDU), is transmitted wirelessly. With this assignment, the test data flows do not interfere with TPDU, so the performance of wireless protocol can be accurately measured by analyzing test data post facto. The feature is especially important for research in wireless network.

(3) Sensor Self-Awareness. The concept of Plug-Configure-Play is similar to the concept of Plug-and-Play. It means that when a node plugs a new sensor, user only needs to specify the usage method of the sensor with simple operation on configuration software running on PC, and the sensor can then work without any driver developing. Thanks to this capability, it seems that node has the capability of sensor self-awareness only if the electric interface of sensor is compatible with node.

(4) Remote Operation. OPCPP can be connected to a local network or Internet. Users can remotely access any node in OPCPP and get any data they need with a browser such as IE. Users can also download program of nodes remotely. It is convenient for user to conduct an experiment whenever and wherever. It will enhance the resource utilization rate significantly and improve the availability of OPCPP.

The remainder of this paper is organized as follows. In Section 2 related works are introduced. In Section 3, architecture of OPCPP is presented. Section 4 gives the implementation of the four strongpoints of OPCPP. OPCPP is shown in Section 5, and the last section is conclusions and future works.

2. Related Works

In the section, we show briefly the existing WSN experiment platforms. Many platforms have been designed for one or several specific emphasis. Some existing platforms focus on teaching of WSN. Ying-xin et al. [4] researched and implemented a teaching experiment platform based on MSP430 and CC2420 chipsets. This platform is mainly a simple temperature and humidity data acquisition and presentation system for teaching WSN. Sun and Andrew Yang [5] presented a suite of practical WSN applications based on TinyOS. It focuses on using WSN data collection and dissemination as examples to provide technology-based education and training. An-Feng et al. [6] proposed WSN routing experimental teaching platform based on Omnet++. It focuses on the
I plug a led, a smoke detector, a temperature sensor on Node A
The led is an output device and connects to Pin1, Pin2, ... of Node A...
The smoke detector is an input device and connects to Pin3, Pin4, ... of Node A...
The temperature sensor is an IIC device and connects to Pin7, Pin8, Pin9, ... of Node A...

**Figure 4: Sensor self-awareness and Plug-Configure-Play.**

**Figure 5: Structure of sensor node.**

teaching and experiment of routing algorithm. Hloupisa et al. [7] proposed an open source development platform for WSN, which is designed for simple WSN teaching.

Some existing platforms focus on providing development platform of WSN. Girod et al. [8] proposed Emstar: a software environment for developing and deploying heterogeneous sensor-actuator networks. It is a software environment for developing and deploying complex WSN applications. Kuo et al. [9] presented an NTP experimental platform for heterogeneous wireless sensor networks. They mainly considered providing an open platform to test and to showcase WSN systems and applications. Handziski et al. [10] proposed a scalable and reconfigurable testbed for wireless indoor experiments with sensor networks. It is mainly designed for indoor deployment of wireless sensor networks. Ertin et al. [11] proposed a testbed for sensing at scale; its design focuses on sensing and scaling. Gireesan Namboothiri et al. [12] designed a testbed for distributed target tracking with directional sensors.

Some platforms mainly consider WSN research. Schiller et al. [13] proposed ScatterWeb, which is a wireless sensor net platform for research and teaching. This platform focuses on the design for low-power consumption. Werner-Allen et al. [14] proposed Motelab: a web-based wireless sensor network testbed. It is a powerful tool mainly for sensor network research. Kuzu et al. [15] introduced a multipurpose WSN platform for research and training in data fusion and multifeature target tracking. Sheu et al. [16] designed and implemented a testbed to realize various experiments in heterogeneous wireless sensor networks.

Compared to the existing related research, we focus on the experiment efficiency, availability, and scalability of WSN experiment platform.

### 3. Architecture of OPCPP

In this section, we introduce the architecture of OPCPP. In order to improve flexibility and scalability of the platform, we design a layered architecture. As shown in Figure 1, the architecture of OPCPP consists of four layers: sensor network layer, smart gateway layer, network communication layer, and application layer.

3.1. **Sensor Networks Layer.** This layer consists of a number of nodes which carry sensors. As shown in Figure 2, the nodes can compose wireless networks of different topology which
are important for collecting the sensor data and controlling the sensor action.

As mentioned above, in order to provide the capability of invasive measurement method, each node of OPCPP has both wireless communication channel and wireline communication channel, as shown in Figure 3. Node can collect sensor data and transmit sensor data to sink node in wireless channel while background data such as the status of node and test data can be transmitted via wireline channel.

3.2. Smart Gateway Layer. This layer is mainly a Plug-Configure-Play service-oriented gateway which is proposed by our research partners [17]. The smart gateway layer consists of WSN manager, protocol translator, and database. WSN manager supervises the sensor network and provides simple application interfaces. Protocol translator executes the protocol conversion between ZigBee and IP-based protocol for convenient access from external network. Database stores the information of sensors on the nodes, test data, and association rules, which is a necessary component for sensor self-awareness.

A creativeness in this layer is the so-called “Plug-Configure-Play.” As shown in Figure 4, when user plugs a sensor or an actuator, such as a LED, a smoke detector, or a temperature sensor on the node, user configures the usage information of sensors, such as the output pin, input pin, and enable level, in the first step, then this information will be saved in the database. WSN manager sends the above information to node carrying the sensor. The node can then play the device based on the information. In this manner, new sensor can be supported without driver development.

3.3. Network Communication Layer. This layer provides multiple ways for application layer to access gateway layer, such as Internet, 3G, Wi-Fi, and LAN.

3.4. Application Layer. The application layer includes three components: experiment demo system, program download module, and sensor manager. Experiment demo system dynamically shows collected sensor data and controls the actions of sensor. This system is suitable for teaching users to understand the concept of WSN and conduct experiments. Program download module provides an interface to download programs to nodes via local network or Internet. This will significantly reduce the repeated manual operations and improve the experiment efficiency. Sensor manager is used for configuring the information of sensors on nodes in order to conduct different sensor experiments flexibly.

4. The Implementation of OPCPP

In this section, we give the implementation of OPCPP by introducing our solution to the main creativeness of OPCPP.

4.1. In-Application Programming. The node of OPCPP consists of a CC2530 module and a Serial2Ethernet module as
shown in Figure 5. CC2530 [18] is a system-on-chip (SoC) solution for IEEE 802.15.4, ZigBee, and RF4CE applications. It enables robust network nodes to be built with very low total bill-of-material costs. The CC2530 combines the excellent performance of a leading RF transceiver with an industry-standard enhanced 8051 MCU, in-system programmable flash memory, 8-KB RAM, and many other powerful features.

In order to implement the function of in-application programming, the program image of CC2530 is divided into two sections. As shown in Figure 5, the first section is for BootLoader and the second section for application code which stores user program. BootLoader is used for reprogramming the application code section. BootLoader receives the packets of application code which users want to download from serial port (serial port can be remotely controlled via the under-mentioned Serial2Ethernet module) and then writes them into the specified flash area in the application code section. After downloading the application code, BootLoader will jump to the application code section for running the application code.

4.2. Noninvasive Measurement Method. Node provides wire-line channel to support noninvasive measurement method. We use a Serial2Ethernet module as shown in Figure 6, the Serial2Ethernet module realizes conversion between serial communication and ethernet communication. Each Serial2Ethernet module has a configured IP address which can be used for identifying nodes. With ethernet cable, each Serial2Ethernet module connects node to smart gateway with ethernet. So, user can do experiment of WSN and acquire test data without influencing the execution of wireless protocol.

4.3. Sensor Self-Awareness. Critical idea for sensor self-awareness is to migrate sensor driver from node in sensor network layer to smart gateway layer. When a new sensor is plugged on a physical node, user will configure the sensor in the user interface of smart gateway layer. The information includes the type of a sensor, sensor name, the GPIO pins of node connected with the sensor, and so forth. Then the database in smart gateway will store the information.

As shown in Figure 7, the process of “play” in Plug-Configure-Play is as follows: (1) when a node starts to work, it will request smart gateway to send its configuration information (the information of virtual nodes on the node). (2) Smart gateway sends the configuration information of the request node. (3) After the node receives the configuration information packet, it will analyze the packet based on the protocol defined beforehand. The configuration packet includes the type of sensors, virtual address, the IO pins which sensors connect to, and so forth. The node sets its GPIO pins and records the information of sensors which it connects to. Then sensors on the node can be driven normally. (4) When application layer calls service interface to control a sensor, gateway sends commands to the node which the sensor belongs to. (5) The node receives the command and calls the interface of sensor driver. (6) Sensor returns data to the node or operates according to the command. (7) The node returns the sensor data or sensor event to gateway.

4.4. Remote Operation. To support remote operation of OPCPP, it should be connected to a local network or Internet.
As above mentioned, every node has a Serial2Ethernet module. Every Serial2Ethernet module implements a lightweight TCP/IP protocol stack. Thus, smart gateway and users can establish a TCP/IP connection to every node. With the TCP/IP connection, smart gateway and users can remotely access every node, update program, and get data of every node. As shown in Figure 8.

5. OPCPP Show

The prototype of OPCPP is shown in Figure 9 as follows. Each node consists of a CC2530 module and Serial2Ethernet module. Each node has 21 GPIO pins, VCC pin, and GND pin. The pins are used for connecting to sensors.

As shown in Figure 10, various sensors are plugged on each node. For example, Node B has five sensors: temperature, humidity, light, buzzer, and ultrasound ranging. Node C has two sensors: accelerometer and angular rate sensor.

When sensors are plugged on a node, we configure the sensor information of the node as shown in Figure 11. The information includes node address, node name, and GPIO pins used by sensors.

After configured, the sensors can work. Node will collect the data of sensors and transmit data to gateway, and then experiment demo system shows these sensor data in graphic manner. For example, the four sensors including temperature, humidity, light strength, and ultrasound ranging are on Node B. After configured as shown in Figure 11, the four sensors on Node B can work. The data of the four sensors will be collected and shown in the experiment demo system. Figure 12 is graphic interface of the four sensor data. Users can watch the real-time changes of the four sensor data.

OPCPP provides an online program download interface for users’ development and research of WSN. Users can also easily establish a TCP/IP connection to the experiment node and get test data using popular software such as network debugger assistant. For example, as shown in Figure 13, we download the code to all the nodes in OPCPP simultaneously. The IP of Node 2 and Node 3 are, respectively, 192.168.1.22 and 192.168.1.23. After programming, we establish a TCP/IP connection to Node 2 and Node 3 with a network assistant and get the test data of Node 2 and Node 3.

After our experiment statistics, the average time of programming a node using a hardware emulator is about 2.3 minutes. The average time of programming a node in OPCPP is about 1.4 minutes. The average time of programming all nodes (10 nodes) in OPCPP simultaneously is about 1.45 minutes. The time to complete an experiment using a...
hardware emulator is proportional to the number of nodes it needs. However, the time to complete an experiment using OPCPP is almost a constant because the process speed of PC is far faster than the nodes and the programming of nodes can be concurrent. Thus, it will greatly reduce the experiment time especially when the WSN experiment involves many nodes.

6. Conclusions and Future Works

Current experiment platform cannot satisfy the need of teaching, research, and development of WSN. In order to provide a simple, flexible, and scalable platform, this paper proposes OPCPP, an online Plug-Configure-Play experiment platform for WSN. It has four prominent strongpoints: in-application programming in batch, noninvasive measurement method, sensor self-awareness, and remote operation. With OPCPP, users can conveniently do application development and research for algorithms and protocols in WSN. It also improves the utilization rate of hardware resources. We also quickly developed a ZigBee-based smart home system prototype based on OPCPP.

For future works, we will improve OPCPP to provide more simple interfaces. We will attempt to combine it with cloud computing technology to build a virtual WSN lab for users. We will also research for algorithms and protocols of wireless network with OPCPP.

Acknowledgment

This work was supported by the National Science and Technology Major Project of the Ministry of Science and Technology of China under the Contract no. 2011ZX03005-005.

References

[1] http://www.ti.com/lit/ug/swru197e/swru197e.pdf.
[2] Z. An, H. Zhu, X. Li, C. Xu, Y. Xu, and X. Li, “Nonidentical linear pulse-coupled oscillators model with application to time synchronization in wireless sensor networks,” IEEE Transactions on Industrial Electronics, vol. 58, no. 6, pp. 2205–2215, 2011.
[3] J. Zhang and L. Gao, “Clustering routing algorithm for WSN based on improved ant colony algorithm,” in Proceedings of the 2nd Annual Conference on Electrical and Control Engineering (ICECE ’11), pp. 2924–2928, Yichang, China, 2011.
[4] Z. Ying-xin, X. Ping-ping, X. Gui-bin, and S. Tie-cheng, “Design and realization of teaching experiment based on wireless sensor networks,” Journal of Electrical & Electronic Education, no. 5, pp. 48–53, 2008.
[5] B. Sun and T. Andrew Yang, “WSNED: a suite of data collection and dissemination applications for Wireless Sensor Network Education,” Journal of Computing Sciences in Colleges, vol. 25, no. 4, pp. 104–111, 2010.
[6] L. An-Feng, R. Ju, and C. Zhi-Gang, “WSN routing experimental teaching platform based on omnet++,” Computer Engineering, vol. 38, no. 11, pp. 258–261, 2012.
[7] G. Hloupis, I. Stavarakas, K. Moutzouris, A. Alexandridis, and D. Triantis, “WSN open source development platform: application to green learning,” Procedia Engineering, vol. 25, pp. 1049–1052, 2011.
[8] L. Girod, N. Ramanathan, J. Elson, T. Stathopoulos, M. Lukac, and D. Estrin, “Emstar: a software environment for developing and deploying heterogeneous sensor-actuator networks,” ACM Transactions on Sensor Networks, vol. 3, no. 3, article 13, Article ID 1267061, 2007.

[9] S.-P. Kuo, C.-Y. Lin, Y.-F. Lee, and W. Hua-Fang, “The NTP experimental platform for heterogeneous wireless sensor networks,” in Proceedings of the 4th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities, 2008.

[10] V. Handziski, A. Köpke, A. Willig, and A. Wolisz, “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, May 2006.

[11] E. Ertin, A. Arora, R. Ramnath et al., “Kansei: a testbed for sensing at scale,” in Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN ’06), pp. 399–406, April 2006.

[12] P. Gireesan Namboothiri, A. Anand Kumar, K. M. Sivalingam, and S. A. V. SatyaMurty, “A testbed for distributed target tracking with directional sensors,” in Proceedings of the IFIP Wireless Days Conference, pp. 10–12, Niagara Falls, Canada, October 2011.

[13] J. Schiller, A. Liers, and H. Ritter, “ScatterWeb: a wireless sensornet platform for research and teaching,” Computer Communications, vol. 28, no. 13, pp. 1545–1551, 2005.

[14] G. Werner-Allen, P. Swieskowski, and M. Welsh, “MoteLab: a wireless sensor network testbed,” in Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN ’05), pp. 483–488, April 2005.

[15] A. Kuzu, S. Erboral, S. Bogosyan, and M. Gokasan, “Multipurpose wireless sensor network platform for research and training in data fusion, and multi-feature target tracking,” in Proceedings of the 2nd Conference on Human System Interactions (HSI ’09), pp. 184–189, May 2009.

[16] J. P. Sheu, C. J. Chang, C. Y. Sun, and W. K. Hu, “WSNTB: a testbed for heterogeneous wireless sensor networks,” in Proceedings of the 1st IEEE International Conference on Ubi-Media Computing and Workshops (U-Media ’08), pp. 338–343, August 2008.

[17] L. Wu, Y. Xu, C. Xu, and F. Wang, “Plug-configure-play service-oriented gateway for fast and easy sensor network application development,” in Proceedings of the 2nd International Conference on Sensor Networks (SENSORNETS ’13), 2013.

[18] http://www.ti.com.cn/product/cc2530.
Submit your manuscripts at http://www.hindawi.com