UiTiOt-Vlab: A Low Cost Physical IoTs Testbed Based on Over-The-Air Programming Approach

Hung Le-Viet, Dang Huynh-Van, and Quan Le-Trung(✉)

Department of Computer Networks, University of Information Technology, Viet Nam National University, Ho Chi Minh City, Vietnam
14520339@gm.uit.edu.vn, {danghv,quanlt}@uit.edu.vn

Abstract. In the plethora of technologies in wireless networks and embedded systems, Internet of Things (IoTs) technology plays a vital role, be applicable in many different domains, such as agriculture, industry, education, transportation, just to name a few. The deployment of IoTs applications in these application domains normally requires complicated phases not only in the analysis, design, implementation but also in the actual operations, e.g., to control the devices and application remotely via the wireless protocols. Therefore, the development of a low cost physical IoTs-based reconfigurable testbed supporting well-known hardware and software platforms is significant and will be presented in this paper. By enabling over-the-air programming (OTAP) with a proper enhancement on Arduino ESP266 device, the proposed testbed system can be used for testing real-time IoTs application, as well as for (re)programming to change or adapt the operations of IoTs devices and IoTs applications remotely, through the wireless network protocols. In addition, the testbed system also provides a virtual laboratory for studying IoTs technology, be a cost-effective approach for students to share IoTs devices and practice IoTs programming remotely.

Keywords: OTA (re)programming · Arduino ESP8266 · OTA upgrade firmware · IoTs reconfiguration

1 Introduction

The Internet of Things (IoTs) connects everything through the Internet, bringing many benefits to users such as more convenient services, utilities for users; better resource allocation; better control of devices and applications remotely via the wireless network protocols. Today, IoTs technologies are indispensable for users in ambient assisted living, transportation, and healthcare [1]. More and more IoTs products have been launched to serve consumers in many different fields such as agriculture [2], industry [3], transportation [4], etc. The release of the breakthrough technologies as 5G technology [5] providing faster speeds, reducing the power consumption and improving the response time and bandwidth. Thus, the appearance of IoTs/5G technologies is necessary for billions of IoTs devices to communicate easily and effectively. These improvements are expected to not only improve the user experience but also pave the way for further improvements in the development of IoTs applications and solutions. In

T. Q. Duong et al. (Eds.): INISCOM 2019, LNICST 293, pp. 181–192, 2019.
https://doi.org/10.1007/978-3-030-30149-1_15

this approach, IoTs testbeds have been developed in order to evaluate IoTs applications and protocols before deploying such IoTs applications and protocols into the real world.

At present, some IoTs-based systems have been integrated with the reconfiguration and reprogramming features to manage IoTs devices and applications remotely. This approach allows the users to reconfigure behaviors of IoTs devices to adapt upon the dynamic changes or unpredictable events in the environment. The remote reprogramming of IoTs devices and IoTs applications through wireless network protocols, aka Over-The-Air-Programming (OTAP), is a potential research topic which has been attracted the attention of many famous research centers in the world. However, most of the existing testbeds are expensive and not to be popular with beginners. In this paper, we propose UiTiOt-VLab, which uses cost-effective Arduino-based IoTs devices based on the OTA programing approach.

The UiTiOt-VLab has been deployed on the IoTs lab and integrated into the cloud infrastructure at the University of Information Technology - VNUHCM. Because the UiTiOt-VLab can be remotely accessed and controlled through a web interface, different users can use this testbed for doing their own researches and experiments in the areas of OTAP/reprogramming, reconfiguration/re-adaptation of both the operations in IoTs devices and the behaviors in IoTs applications.

The paper structure is organized as follows. Section 1 introduces the topic and shares researches and systems deployed in the world as well as contributions in our research. Section 2 lists similar works implemented in the world, summarizing the nature, purpose, and mode of operation of each project. Section 3 focuses on clarifying the process of system deployment. This section contains 4 sub-sections, each of which presents the core work when building the system. Part 3 is the result of our deployment in a realistic scenario. Finally, Sect. 4 concludes the research paper with the conclusions and future work.

2 Related Work

Nowadays, there are many existing IoT device management systems which released in the modern industry and bring positive effects.

PlanetLab [6] is a global coverage network for developing and delivering wide area network services. The goal of PlanetLab is to grow up to 1000 geographically distributed sensor nodes, which are connected by diverse links. PlanetLab allows multiple services to run simultaneously and continuously, each service in a separate sensor node. Worldwide network services have appeared such as network-embedded storage [7], peer-to-peer file sharing [8], content distribution networks [9], robust routing overlays [10], scalable object location [11], scalable event propagation [11]. All of these applications have one common: take advantage of the wide connectivity with network protocols. To support the design and evaluation of the applications, PlanetLab has been developed to form a global network of networks which is initially deployed over 100 nodes distributed on 42 websites. In the near future, PlanetLab will be a microcosm for the next generation Internet.

MoteLab [12] includes a lot of sensor nodes connected to a central server that handles reprogramming and logs generated data by experiments by a continuous database. The user can access data via a web interface or directly from the database. MoteLab also allows users to interact directly with individual sensor node during testing using the Web interface. MoteLab helps everyone can deploy quickly by providing connections into real-sensor network device network. Additionally, MoteLab speeds up error correction and application development by automatically data logging, allowing the software to evaluate the performance of sensor networks operate offline. In addition, by providing a web interface, MoteLab allows both internal and external users to access the testbed, the system limits access time and limits bandwidth to ensure uniform distribution. The project has great significance for research as well as teaching. MoteLab's source is shared free, easy to install and has been used in some research institutes. The widespread use of MoteLab will accelerate and improve the research of wireless sensor networks.

FIT/IoT-LAB [13] is part of the FIT experimental platform, providing a method for testing IoT with mobile wireless communication devices in both network and application layer, thereby speeding up design and research advanced network technologies for the Internet. There are more than 1500 wireless sensors nodes spread over six different sites in Grenoble, Lille, Saclay, Strasbourg, Paris, Lyon (France) to create a heterogeneous platform. The project provides fixed nodes or mobile nodes moving on the ground to serve mobility in testing. The remote users access the Web interface to register necessary entities, with direct command line connection to the platform.

EU IoT Lab [14] is a large-scale research project in Europe, aiming to study the potential of providing IoTs equipment in society, creating a multidisciplinary research environment, experiment with interaction from multiple users. EU IoT Lab also provides Testbed service, a platform to create a combination of participant groups (workers, end-users, and researchers) to solve practical and collaborative challenges at work.

Indiana IoT Lab [15] provides pioneers in this IoTs industry with resources in a collaborative environment. Indiana IoT Lab builds a connection between industries established in India with technology companies specializing in IoT. The area is 24,000 feet wide.

Almost systems in the world use the expensive boards with the ability to operate in the large project; however, these types of board are not popular with students, who start to study and approach to IoTs technologies. The re-programming research uses Arduino ESP8266 platform, which is low cost and easy to program, is suited to the current research needs. To sum up, the proposed solution solves problems in remote (re)programming low-cost Arduino-based devices with many utilities such as managing IoTs equipment remotely, providing an environment for users to test their code, connecting with each other to support research and work. The equipment in the project can be provided for a large number of users to conduct their experiments simultaneously with various scenarios.

3 Implementation

The IoTiOt-VLab is a system which comprises a set of software tools for managing an IoTs testbed of connected sensor network nodes. A central server handles the authorized connection between user and device, reprogramming nodes, logging data, and providing a web interface for users. The UiTiOt-VLab consists of three main software components include:

- Mongo Database Backend: stores data collected during experiments, information used to generate web content, and state driving testbed operation.
- Web Interface: ReactJS generate pages which present a user interface for deployment, granting, and data collection as well as an administrative interface to testbed control functionality.
- Job Daemon: Python script run as a cron job to set up and tear down jobs.

Throughout this section, we refer to a "job" running on UiTiOt-VLab. A UiTiOt-VLab job includes some number of executables nodes, a description mapping each pair of nodes used to executable, and several Javascript files used for data logging. To create a job, the user uploads the sketch onto board via a web interface. Once the job is created, UiTiOt-VLab executes the script on the real device and send data to the user. The UiTiOt-VLab limits usage time of each device so that other users can use the testbed to run their own code.

The system architecture is described in detail by listing all components of the model. The Arduino device block includes a master and slave board. The master is preconfigured to receive and transmit code to Slave. The sensor/actuator devices directly connect to the Slave board to run the author's programs. The receiver data, parameters are distributed to Master, then forwarded to the server.

In Fig. 1, describes the overview architecture of the system, in which:

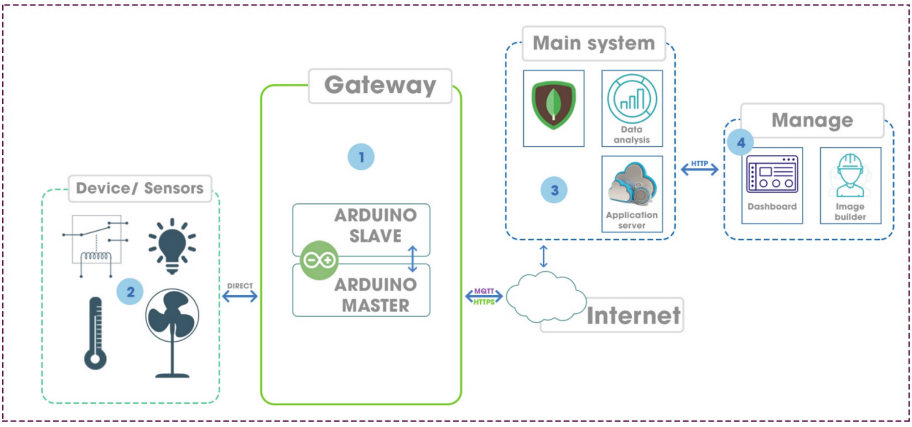

Fig. 1. Architecture of testbed system

- IoTs gateways are Arduino-based devices which are remotely (re) programmable.
- Sensors connected directly to IoTs gateway via the Slave board.
- Application system, including rental service, management, and access devices.
- The manage block is a web-based user interface which is used to upload code into the board.

According to the above architecture, our IoTs system consists of three main components: *(i)* sensors, actuators control block, *(ii)* firmware update mechanism for the master and slave Arduino boards, and *(iii)* user interface for remote deployment.

3.1 Sensors and Actuator Control Block

Depend on the using purposes, there are many types of sensor nodes for monitoring environmental parameters. For example, the sensors array can include pH sensor, temperature sensor, humidity sensor, module relay to control the on-off of the light or the machine, which is connected to the Arduino circuit directly. In this proposed system, all sensors, actuators have to be connected in advanced to provide the sensor lists and connection ports for users on the website.

3.2 OTA Update Firmware Mechanism for Arduino Boards

Arduino [16] is a microprocessor circuit board used to build applications that interact with each other or with a favorable environment. The Arduino hardware includes a power circuit board designed on the Atmel 8bit AVR processor platform or 32-bit ARM Atmel. The current models are equipped with 1 USB interface port, 6 analog input pins, 14 digital I/O ports compatible with many different expansion boards.

It is reported that Arduino-based programming is easy, inexpensive for scientists, students, and professionals. It is easy to design a simple IoT-based application that can interacts with the environment through sensors and actuators such as simple robots, environmental parameters collection or motion detection. In addition, the Arduino community has also provided an integrated development environment (IDE) that runs on regular personal computers and allows users to write Arduino programs in code C/C++.

The Arduino supports both digital and analog input. In terms of output, we can use PWM to simulate an analog output. Digital I/O pins on Arduino can be configured for input or output. Only analog input pins are used for Input.

3.2.1 The Limitation of the Existing OTA Update on ESP8266

To conduct the basic OTA update firmware process on ESP8266, it is efficient to use an embedded computer such as Raspberry Pi works as a local server. Raspberry Pi (RPi) is a small-sized computer with powerful hardware built-in capable of running the operating system and installing many applications. RPi is suitable for applications that need powerful processing, multitasking, entertainment, and especially low cost. RPi features built around the Broadcom BCM2835 SoC processor (is a powerful mobile processor with small size, or used in mobile phones) including CPU, GPU, audio/video processor, and other features… integrated inside this low-power chip (Fig. 2).

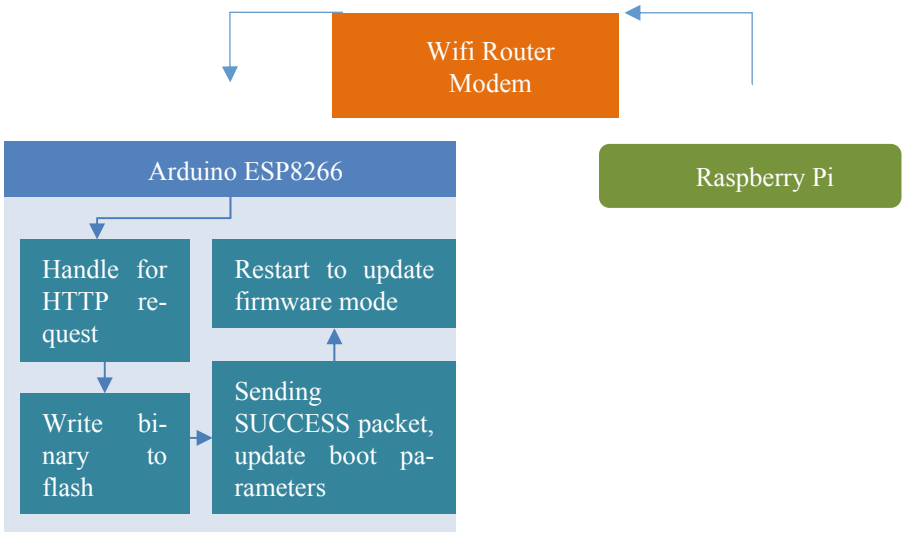

Fig. 2. Diagram of loading an Arduino ESP8266 with basic OTA library

In this library, RPi works as a local server, to control and deploy program onto Arduino ESP8266. The detailed operation of the reprogramming via the RPi follows steps below:

1. RPi checks and compiles sketch into a .bin file, send a signal via Wifi router to Arduino start the process of reconfiguring.
2. The packet is received by the Arduino, processed and written to the last section of the flash memory.
3. After receiving the last packet of the binary file, Arduino returns the SUCCESS packet to RPi and installs boot params to enable the built-in OTA mode in the bootloader.
4. The OTA mode available after ESP restart, copy the entire binary to 0x000000 in flash and restart after the copy has been completed.

However, this approach has some disadvantages which are inappropriate to be used in a reusable IoT testbed. These drawbacks include: (1) the user's code must be reprocessed, insert the code to call the OTA library to make sure OTA is active after being uploaded; (2) the OTA library won't work properly if there is an infinite loop or delay too long in the user code; (3) all Serial.print() functions can't run.

3.2.2 The Proposed Method for OTA Programming the Running Board

Due to the physical design limitation, Arduino only handles single threads per task, so we cannot run the received signal to begin the reconfiguration process and user code execution tasks. In this case, we suggest using an ESP8266 board to be able to handle the two tasks simultaneously (Fig. 3).

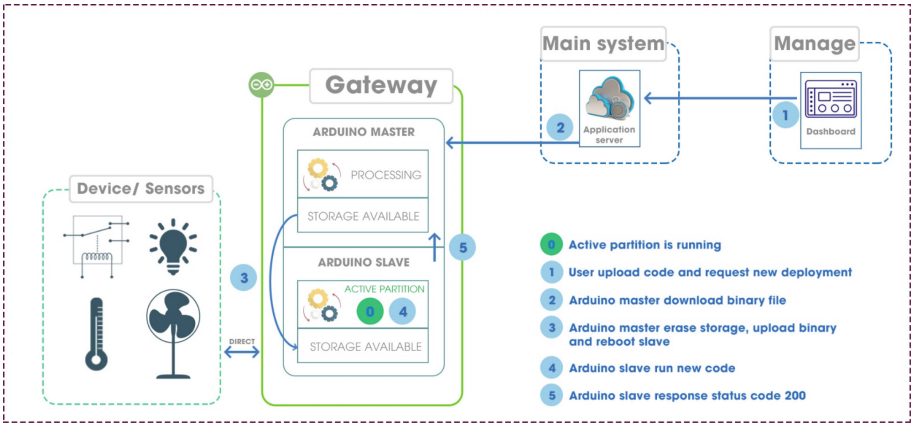

Fig. 3. The diagram of uploading firmware uses an additional ESP as a virtual COM port

The code on the ESP8266 master simulates the operation of the upload code with the virtual COM port set up by sending and receiving HTTP request. After receiving the binary from the Server via Ethernet port and writing to flash, the ESP8266 Master sends restart signal to Slave and then switches to the upload mode and sends it to SYNC to establish the connection. The flash section used to store the binary file is blanked by writing 0xff. The Master board continues to transmit binary packets to write to flash starting at 0x000000 address. Finally, the Slave board is restarted into the executive mode.

The detailed operation of the Master board sequentially follows steps below:

1. Every 10 s, the Master board sends a ping signal to the server with a device name and password to inform the status of the operation.
2. Every 10 ms, the Master board reads packets from Slave's serial port; The Master sends information from the serial port to the MQTT server with topic ARD_NAME/com every 100 ms, so that this information is available to the user promptly.
3. If there is an MQTT packet to topic ARD_NAME/binary, then receive the file, write it to memory and stop reading the data from the Serial, stop sending the Ping packet to set device status to offline.
4. After receiving the binary file completely, the master begins waiting for the server sends the signal to start the process of uploading code on the device. From the time of the last packet of the binary file is completed, the master waits for a 10 s timeout period, if there is no request, responding to the server status with the topic ARD_NAME/status.

5. After the process of uploading the code to Slave, the Master evaluate whether the loading process has an error, whether the Slave startup with an error and record the result into EEPROM. Then, send the Flash process feedback to the server with the topic ARD_NAME/status.
6. Finally, the Master performs a soft reset to return the original state.

The code uploading process from user to Slave board consists of 3 steps below:

1. The server sends a published message MQTT to the topic ARD_NAME/binary with the compiled binary file and requests that the device proceeds the reconfiguration. The Master board subscribes to a topic with the device identifier to receive signals from the server. After completing the file transfer process, the Master board automatically upload binary to Slave board.
2. After uploading the binary file to flash memory of the Slave, the Master sends an MQTT request to the topic ARD_NAME/status to notify the status of the uploading process to the server (Failed, successfully uploaded or no file). If the timeout expires, report the error reported from the server to the user.
3. The server continues to send requests include the data received from Slave, which are processed by the server and forward to users.

3.3 The Web-Based Application for Remote Deployment

The main function of the web-app is the user interface to manage, deploy the reconfiguration, and activate the reprogramming process. The main components of the web-app include: (i) Displaying ready-to-use devices, separating each platform, (ii) Managing lists, modifying information of all devices in the system, (iii) Managing user list, edit profile, system management permissions for each object, (iv) managing content of practice tutorials, (v) Interacting with Arduino devices (Fig. 4).

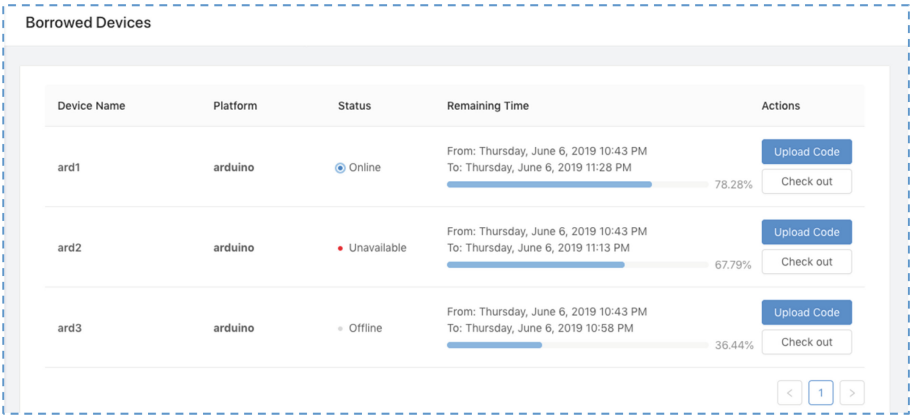

Fig. 4. Manage devices in use interface

Device management system with functions for management level (admin, technical staff, lecturers): add, delete, get device description, information access equipment, edit the description, show device's status; functions for all role, including show device lists, borrow and return devices, load code onto Arduino board (Fig. 5).

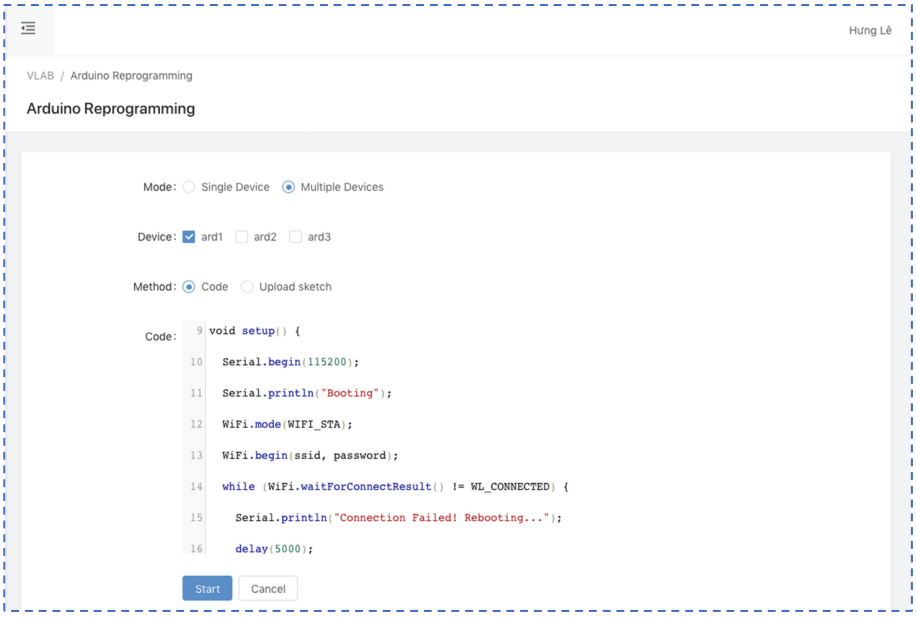

Fig. 5. Upload code onto the Arduino board interface

The editing source code page provides 2 modes: compile and upload code onto one device or multiple devices. The upload to single device mode is to upload and deploy on the device that the user presses the "Load" button on the management site. The upload to multi-device mode is to upload and deploy one array of user-selected devices below. In addition, we provide two more methods: type the code directly on the website or upload the sketch file to the system. However, the method of processing code and feedback for users is similar.

4 Experiment Results

In this section, the experimental results of the proposed system are illustrated. The results provide information on various metrics related to the use of algorithms, analysis of packet transmission performance of wireless sensor nodes.

Compared to the three OTA libraries provided by Arduino Dev, our solution has some remarkable betterment, which is suitable to be used in practical implementation. The Table 1 below indicates the advantages and disadvantages of our proposed solution with the existing OTA approaches in ESP8266.

Table 1. The comparison of three OTA libraries on ESP8266 with our proposed solution

Feature	**With Arduino IDE**	**With Web Browser**	**With HTTP Server**	**The proposed method**
User code reprogramming	✓	✓	✓	×
Using only one board	✓	✓	✓	×
Affected by errors from user code	✓	✓	✓	✓
Sending the data from the serial port	×	×	×	✓
Deploying on the circuit without wireless protocol	×	×	×	✓
Multi-upload onto the board at the same time	×	×	✓	✓
Easy to manage device	×	×	×	✓

By embedding a timer counter on the device, the time of distribution code from the server system to the device clearly, accurately. We have collected some useful information such as the code size before compilation, the code size after compiling (before loading onto the device), disseminating time, number of packets to be transmitted as well as the average size of the package. The diagram below shows the results in the distribution process of source code with 05 different applications, each application tested 05 times (Table 2).

Table 2. The experimental results of the distribution process with 5 applications

- Application name - File .ino length (kilobytes) - File .bin length (kilobytes)	No.	Download time (s)	Time of deployment from master to slave (s)	Time of deployment with the system (s)	Time of deployment with IDE (s)
Blink 0,627 261.824	1	5,500	31,604	37,104	32,80
	2	5,394	31,605	36,999	33,13
	3	5,273	31,575	36,848	32,71
	4	5,505	31,581	37,086	32,41
	5	5,369	31,575	36,944	32,10
Basic HTTP Client 1,712 280.752	1	5,685	33,341	39,026	35,52
	2	5,603	33,345	38,948	36,34
	3	5,475	33,338	38,813	36,54
	4	5,550	33,339	38,889	36,13
	5	5,583	33,340	38,923	36,05
Web updater 0,989 306.352	1	6,025	35,635	41,660	39,13
	2	6,000	35,634	41,634	39,31
	3	6,061	35,642	41,703	39,80
	4	5,990	35,639	41,629	39,33
	5	6,014	35,633	41,647	39,04
Chat server 2,026 294.768	1	5,938	34,535	44,473	43,06
	2	5,676	34,542	44,218	42,92
	3	5,785	34,538	44,323	42,95
	4	5,674	34,542	44,216	42,88
	5	5,784	34,534	44,318	43,10
Basic OTA 1,540 287.792	1	5,845	33,986	39,831	37,29
	2	5,719	33,987	39,706	37,51
	3	5,840	33,985	39,825	37,54
	4	5,684	33,959	39,643	38,34
	5	5,739	33,985	39,724	37,99

5 Conclusion and Future Work

In summary, this paper introduces the development of UiTiOt-VLab, which is low cost, physical IoTs testbed, working as an IoTs virtual laboratory for testing and learning IoTs programming remotely via the wireless protocols. Our enhancement on OTA upgrade firmware mechanism in ESP8266 board allows the running devices can receive new firmware version simultaneously. The proposed solution is not only suitable to build a physical IoTs testbed but also helpful for real world IoT-based deployments.

In the future, we will do more research in the development of reconfiguration and reprogramming solutions on other microchip platforms and integrate such developed solutions in the actual IoTs-based projects.

Acknowledgement. This research is funded by University of Information Technology-Vietnam National University HoChiMinh City under grant number D1-2019-12.

References

1. Building A Digital World Consumers Can Trust: Proposed recommendations from the consumer movement to the G20 member states. Consumers International and The Federation of German Consumer Organizations (2017)
2. TongKe, F.: Smart agriculture based on cloud computing and IOT. J. Convergence Inf. Technol. (JCIT) **8**(2) (2013)
3. Li, R., Song, T., Capurso, N., Yu, J., Couture, J., Cheng, X.: IoT applications on secure smart shopping system. IEEE Internet Things J. **4**(6), 1945–1954 (2017)
4. Song, T., Capurso, N., Cheng, X., Yu, J., Chen, B., Zhao, W.: Enhancing GPS with lane-level navigation to facilitate highway driving. IEEE Trans. Veh. Technol. **66**(6), 4579–4591 (2017)
5. Hattachi, R.E., Erfanian, J.: 5G White Paper. NGMN Board (2015)
6. Chun, B., et al.: PlanetLab: an overlay testbed for broad-coverage services. ACM SIGCOMM Comput. Commun. Rev. **33**(3), 3–12 (2003)
7. Kubiatowicz, J., et al.: OceanStore: an architecture for global-scale persistent storage. In: ACM SIGARCH Computer Architecture News - Special Issue: Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000), vol. 28, no. 5, pp. 190–201 (2000)
8. Rowstron, A., Druschel, P.: Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. In: SOSP 2001 Proceedings of the Eighteenth ACM Symposium on Operating Systems Principles, pp. 188–201 (2001)
9. Wang, L., Pai, V., Peterson, L.: The effectiveness of request redirection on CDN robustness. In: ACM SIGOPS Operating Systems Review - OSDI 2002: Proceedings of the 5th Symposium on Operating Systems Design and Implementation, vol. 36, pp. 345–360 (2002)
10. Andersen, D., Balakrishnan, H., Kaashoek, F., Morris, R.: Resilient overlay networks. In: SOSP 2001 Proceedings of the Eighteenth ACM Symposium on Operating Systems Principles, vol. 35, no. 5, pp. 131–145 (2001)
11. Ratnasamy, S., Handly, M., Karp, R., Shenker, S.: Topologically-aware overlay construction and server selection. In: Proceedings of the IEEE INFOCOM Conference, pp. 1190–1199, 2002
12. Werner-Allen, G., Swieskowski, P., Welsh, M.: MoteLab: a wireless sensor network testbed. In: IPSN 2005 Proceedings of the 4th International Symposium on Information Processing in Sensor Networks, vol. 68 (2005)
13. Adjih, C., et al.: FIT IoT-LAB: a large scale open experimental IoT testbed. In: 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), Milan, Italy (2016)
14. Ziegler, S., Rolim, J., Nikoletsea, S.: Internet of Things, crowdsourcing and systemic risk management for smart cities and nations: initial insight from IoT Lab European research project. In: 2016 30th International Conference on Advanced Information Networking and Applications Workshops (WAINA), Crans-Montana, Switzerland (2016)
15. Velis, M.: 'Smart City' trends and opportunities for collaboration. In: Consulate General of the Kingdom of the Netherlands, Chicago (2017)
16. Sarik, J., Kymissis, I.: Lab kits using the Arduino prototyping platform. In: 2010 IEEE Frontiers in Education Conference (FIE), Washington, DC, USA (2010)