Enabling Over-The-Air Provisioning for Wearable Devices

Wei-Han Chen, Fuchun Joseph Lin(✉), and YaHua Lee

Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan
{alwayschoco.cs04g, fjlin, yahua.cs05g}@nctu.edu.tw

Abstract. The Internet of Things (IoT) is growing rapidly with more and more devices connected to the Internet. Among IoT devices, wearable devices have become an important category due to their wide applicability. However, most of the wearable products are designed to provide fixed services; once they are deployed, their functionality is difficult to change. In this research we develop new technologies that would enable the OTA (Over-The-Air) provisioning over BLE for the wearable devices. We use the method of Interworking Proxy Application Entity (IPE) defined in oneM2M to enable the OTA provisioning. Our experimental results show that the IPE method is an effective mechanism to support the OTA provisioning for wearable devices.

Keywords: OTA provisioning · Wearable devices · oneM2M
BLE · IPE

1 Introduction

In the recent years, the Internet of Things (IoT) has been growing rapidly with a large number of devices connected to the Internet. The Gartner forecasts that 20.4 billion connected devices will be in use by 2020 [1]. Among these devices, wearable devices which can be worn on the human body takes a great portion and brings great convenience to people's life. They are adopted in many IoT application areas such as healthcare and smart home [2]. However, most of the wearable products are not programmable. Consequently, it is very difficult to upgrade or change their service functions once they are deployed.

Over-The-Air (OTA) provisioning describes the ability to update firmware or install new applications for mobile devices via wireless networks. To enable the OTA provisioning for the wearable devices, we need to take their limited capabilities such as storage and battery life into consideration [3]. Bluetooth Low Energy (BLE) is an emerging wireless standard suitable for wearable devices due to its features of low power consumption and low storage requirement [4, 5].

In this research, we enable the OTA application provisioning for BLE devices based on oneM2M Interworking Proxy Application Entity (IPE) [6, 7]. To evaluate the effectiveness, we adopt four criteria including power consumption, system performance, transmission efficiency and software complexity and take the application size as the parameter to evaluate its impact on the performance.

Y.-B. Lin et al. (Eds.): IoTaaS 2017, LNICST 246, pp. 194–201, 2018.
https://doi.org/10.1007/978-3-030-00410-1_23

The rest of the paper is organized as follows: Sect. 2 gives some background knowledge of OTA provisioning, BLE protocols and oneM2M technologies. Section 3 introduces the related research works in OTA provisioning. Section 4 describes the tools, architecture design and implementation results. Section 5 explains the evaluation criteria and analysis results. Finally, in Sect. 6 we present our conclusion and future work.

2 Background

2.1 Over-The-Air Provisioning

OTA is a standard for transmitting application-related information to the devices wirelessly. As the smartphones provide more functionalities, OTA has been used to distribute the new applications to smartphones via cellular networks. Recently, with the growing of wireless sensor networks and IoT, OTA is being realized by using low power consumed protocols such as 802.15.4, Zigbee, and BLE.

2.2 Bluetooth Low Energy

BLE is part of the Bluetooth core specification 4.0 or later releases developed by the Bluetooth Special Interest Group (SIG). It is designed with low power consumption and low complexity suitable for constrained devices and IoT applications. Due to its low cost compared to other similar wireless technologies, it is rapidly and widely implemented in smartphones, wearable devices, sensors, and other devices.

2.3 oneM2M Technologies

The oneM2M is the global standard for M2M/IoT platforms. In addition to addressing the requirements for M2M Service Layer, it defines the architecture, protocols, APIs, interfaces and security for M2M/IoT services. In our research, we use an oneM2M-based platform, OpenMTC, developed by FOKUS [8] as our provisioning server that accepts users' requests and interacts with the BLE devices.

3 Related Work

Fjellheim [9] designed an adaptive platform to support customized application delivery via various protocols according to required metadata. They also implemented an adaptable Web server to support OTA over HTTP. Vo and Torabi [10] proposed a framework for OTA provider-initiated software update on mobile devices subscribed to the service. Ndie et al. [11] came up with a method of provisioning mobile applications via Bluetooth between two mobile devices. Though there are some interesting OTA provisioning ideas reported in these papers, few of them use BLE as the communication method. In addition, none of them are designed specifically for wearable devices or based on an IoT platform. Moreover, most of them require subscription to the providers. In our research, we propose OTA provisioning for the wearable devices over oneM2M and BLE that differs from existing works.

4 Architecture Design and Implementation

4.1 Tools for Architecture Design

- Nordic Device Firmware Update (DFU) Mechanism for nRF5 SDK v11.0.0

In order to support OTA provisioning, Nordic provides OTA DFU mechanisms in their Software Development Kit (SDK). The DFU Service exposes necessary information to perform provisioning for the devices. The high level architecture is depicted in Fig. 1 where two devices are required for its operations: (1) DFU Controller, which triggers the DFU procedure and transfers the application images, and (2) DFU Target on which a bootloader is needed to start the DFU mode, manage the DFU procedure and activate the new firmware.

- oneM2M Interworking Proxy Application Entity

The oneM2M defined IPE in TS-0001 [6] as the solution for interworking with non-oneM2M entities. Figure 2 shows that it is designed as an AE (Application Entity) that enables different level of interworking between non-oneM2M and oneM2M interfaces including protocol interworking, semantic information exchange and data sharing. For OTA provisioning, we use IPE to translate BLE protocol messages to oneM2M RESTful operations such as HTTP and to exchange the data models between BLE devices and oneM2M-based platforms.

4.2 Architecture Design of Over-The-Air Provisioning with IPE

In our research, we use nRF52 DK as our BLE device and a Raspberry Pi 3 model B running Linux-like Raspbian as our provisioning server. In addition, a CSR Bluetooth USB dongle adapter is installed in the Raspberry Pi 3 to enable BLE communications. Bluez [12] is installed to control the Bluetooth dongle adapter in the Raspberry Pi for supporting BLE layers and protocols on Raspbian. As OpenMTC Release 4 is implemented in Python while Bluez only provides APIs in C language, we have to adopt another open source, Bluepy [13], in order to use Python APIs for BLE communications on OpenMTC.

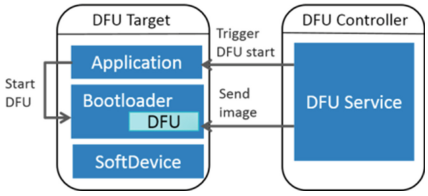

Fig. 1. Device Firmware Update architecture

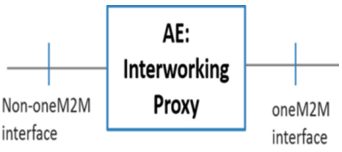

Fig. 2. Interworking Proxy Application Entity

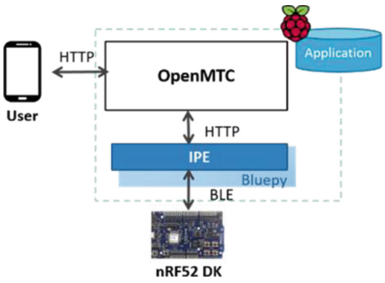

Fig. 3. System architecture

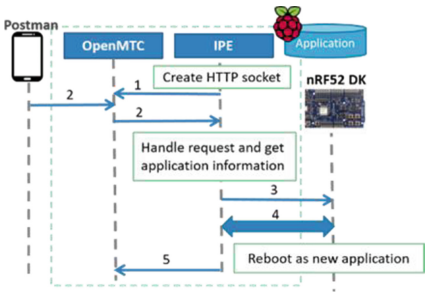

Fig. 4. Provisioning process

Our system architecture consists of four components as shown in Fig. 3.

1. OpenMTC: It is the provisioning server used to store the information of available applications and to forward the user's provisioning requests to the BLE device via IPE by subscription/notification mechanisms of oneM2M.
2. User: In our architecture, the user needs to send HTTP requests to retrieve the available application list and initiate application provisioning. We use Postman [14] to simulate the user's application for sending related HTTP requests.
3. BLE Device: This is nRF52 DK used as the provisioning target. The nRF52 DK is installed with s132 SoftDevice and the bootloader and is thus capable of supporting the DFU functionality.
4. IPE: Our IPE, which acts as the DFU controller and gateway application entity on the platform side, takes responsibility for interworking with the nRF52 DK and OpenMTC to achieve the provisioning functionality. It handles the HTTP requests from the platform, and manages the provisioning process with the BLE device.

4.3 Implementation of OTA Design

Figure 4 shows the step-by-step IPE provisioning process as follows.

1. First, we starts OpenMTC and IPE. The IPE will create a HTTP socket and subscribe to OpenMTC to receive the notification about the provisioning request.
2. When the user sends a provisioning request including the device address and the application to be provisioned, OpenMTC will notify IPE of the request. IPE then parses the request and gets the application path from the available application list.
3. Next, IPE uses APIs provided by Bluepy to connect to the nRF52 DK according to the device address in the request and triggers the DFU procedure.
4. IPE will carry out the DFU operations with nRF52 DK as defined in DFU Service by Nordic.
5. At the end of the process, nRF52 DK will reboot with the new application and IPE will create a resource in the platform to record the provisioning result.

Table 1. Sizes of applications.

Method/application	UART application	Proximity application	HRM application
IPE	26,566 bytes	35,040 bytes	23,180 bytes

5 Experiment Results and Analysis

5.1 Applications Used for Evaluation

To evaluate the effectiveness of our design of OTA provisioning, we create three different kinds of applications. Table 1 shows the sizes of these applications and each of them is explained below.

- Universal Asynchronous Receiver/Transmitter (UART) Application. The basic concept of UART is using one channel to transmit data and the other one to receive data between two devices. This application will receive data from the central device and print it on the monitor over serial connection on the nRF52 DK.
- Proximity Application. This application implements the Proximity Profile that alerts the user by the LEDs on the nRF52 DK when the connected device are too far apart. The blinking frequency will increase with the distance between the nRF52 DK and the central device connected.
- Heart Rate Measurement (HRM) Application. HRM is an implementation of the Heart Rate Service profile. When the notification is enabled, the simulation of heart rate value will be sent to the central device.

We need to integrate DFU Service into these applications, and propagate the BLE stack events to DFU Service. Besides, the applications have to support sharing bonding information which enables the device to advertise directly after entering the DFU mode.

5.2 Evaluation Criteria

Below we describe our evaluation criteria.

- Power Consumption. It is one of the critical issues in the design of wearable devices and their applications due to their inherent limit of battery-supplied power. As the nRF52 DK provides power measurement functionality, we can use the ampere-meter to directly monitor the current and derive power consumption during the DFU process.
- Transmission Efficiency. This is an important metrics to measure the efficiency of the connection method design for data transmission. We use Wireshark to capture the relevant packets and investigate the transmission time and average rate of application image transmission.
- System Performance. As the number of connected devices keeps growing, the system performance of the M2M platform becomes more and more important. We investigate the CPU utilization of the oneM2M platform while executing the application provisioning with the device. To profile our implementations more precisely, we adopt psutil, a Python program profiling tool, into our implementations instead of using the traditional tools in Linux environment (e.g. ps, top).
- Design Complexity. Finally, we analyze the design and implementation complexity of our architecture based on the system requirements including protocol support, prerequisite knowledge, device and firmware design etc.

5.3 Experiment Results and Analysis

The experiment environment is set up as follows. First, we use OpenMTC as our oneM2M platform and install it on a Raspberry Pi 3 Model B. For wearable devices, we adopt the nRF52 DK with s132 SDK and bootloader as the target device. Since the connection intervals of BLE connections will affect the performances of power consumption and transmission efficiency, we set the minimum as 7.5 ms and the maximum as 30 ms to reduce this influence. Below we report the results for each evaluation criteria.

- Power Consumption. The current of the nRF52 DK changes according to different operations in the DFU procedure. Consequently, we record all the values for analysis. Take the provisioning UART application for example (See Fig. 5), the average current is between 1 and 1.5 mA during the file transmission and the curve is stable. The analyses of provisioning three applications indicate that the power consumption will be affected by the application size (See Tables 2 and 3).
- Transmission Efficiency. For the transmission efficiency, we use Wireshark to catch the packet information during the provisioning process. The analysis results are shown in Table 3. Due to the limited payload of BLE, each packet can contain only 20 bytes of data at most. However, we can modify the connection intervals to achieve high throughput. In our experiment, the Proximity application has the largest packet which is about 35000 bytes but it only takes 20 s to transmit, which is acceptable for the users.
- System Performance. The platform CPU utilization of provisioning three applications respectively are shown in Fig. 6. We have marked the important timestamps for each, such as managing HTTP request, establishing connection with the device and transmitting the application images. We found that if the application size is bigger, then it will consume more CPU resource during the transmission procedure.

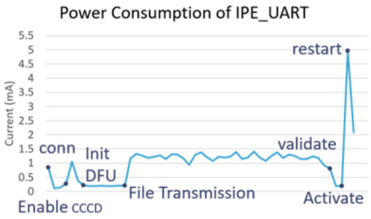

Fig. 5. Power consumption results of UART application

Table 2. Analysis results of power consumption.

	UART	Proximity	HRM
Image transmission (mA)	18.25	21.55	13.92
Whole procedure (mA)	23.71	26.07	18.32

Table 3. Analysis results of transmission efficiency.

	UART	Proximity	HRM
Process time (s)	25.89	34.10	20.72
Image transmission (s)	14.82	20.19	13.17
Average rate (bytes/s)	1792.67	1735.57	1760.24

- Design Complexity. The analysis results of the proposed method are shown in Table 4. The results show that our method is easy for development because we don't need to implement the whole system by ourselves like other OTA systems. In addition, it can be adopted for any BLE devices equipped with Nordic nRF51 SoC and later models.

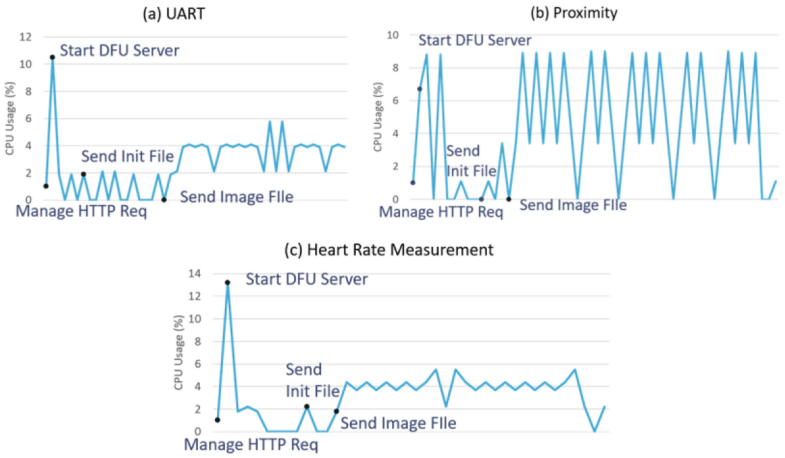

Fig. 6. Analysis results of system performance (CPU utilization)

Table 4. Design complexity analysis results

Items	IPE method
Support protocol stack	BLE 4.0 and above
Prerequisite knowledge	BLE, Nordic DFU Mechanism, oneM2M
Device requirement	Nordic nRF51 DK and above (nRF52 here)
Firmware requirement	nRF51/nRF5 SDK, SoftDevice + Bootloader
Enable OTA in application	Include DFU Module and Device Manager Library
Programing in oneM2M platform	IPE
Programming language	Python + ARM C
Third-party package/library	Bluez, Bluepy, OpenMTC

6 Conclusion and Future Work

This research proposed the architecture of the OTA application provisioning over BLE for the wearable devices based on oneM2M IPE. We have shown that our architecture is a flexible and effective method for OTA provisioning. With our architecture, the users can upgrade their wearable devices wirelessly by sending HTTP requests via the smartphones or the computers.

As the number of devices grows rapidly, IPv6 is an ideal protocol with more available addresses and stateless address auto-configuration tools for IoT network applications. 6LoWPAN over BLE recently defined by IETF describes the details of IPv6 over BLE links. In our future work, we plan to enable OTA provisioning via 6LoWPAN over BLE and compare it with the IPE method.

Acknowledgment. The research in this paper is funded by Ministry of Science and Technology (MOST) of Taiwan Government under Project Number MOST 105-2218-E-009-004.

References

1. Gartner: 8.4 billion Connected "Things" Will be Use in 2017, Up 31 Percent from 2016, 7 February 2017. http://www.gartner.com/newsroom/id/3598917
2. Lee, S.Y.: Situation Awareness in a smart home environment. In: IEEE 3rd World Forum on Internet of Things, pp. 678–683 (2016)
3. Patel, M., Wang, J.: Applications, challenges, and prospective in emerging body area networking technologies. IEEE Wirel. Commun. **2010**(17), 80–88 (2010)
4. Dementyev, A., Hodges, S., Taylor, S., Smith, J.: Power consumption analysis of Bluetooth Low Energy, Zigbee and ANT sensor nodes in a cyclic sleep scenario. IEEE Int. Wirel. Symp. **2013**, 1–4 (2013)
5. Gomez, C., Oller, J., Paradells, J.: Overview and evaluation of bluetooth low energy: an emerging low-power wireless technology. Sensors **12**(9), 11734 (2012)
6. oneM2M: TS 0001 v2.10.0, Functional Architecture
7. Ting, Y.Y.: A Comparison and Evaluation of Different BLE Connection Methods for Wearable Devices. National Chiao Tung University, Hsinchu (2016)
8. Fraunhofer Fokus: OpenMTC. http://www.open-mtc.org
9. Fhellheim, T.: Over-the-air deployment of application in multi-platform environments. In: Australian Software Engineering Conference (2006)
10. Vo, C.C., Torabi, T.: A framework for over the air provider-initiated software deployment on mobile devices. In: 19th Australian Software Engineering Conference, pp. 633–638 (2008)
11. Ndie, T.D., Tangha, C., Sangbong, T., Kufor, A.F.: Mobile application provisioning using Bluetooth wireless technology. J. Softw. Eng. Appl. **4**, 95–105 (2011)
12. Bluez. http://www.bluez.org
13. Bluepy. https://github.com/IanHarvey/bluepy
14. Postman. https://www.getpostman.com