Peer-to-Peer Netw. Appl. (2018) 11:139–151
DOI 10.1007/s12083-016-0505-9

Towards the oneM2M standards for building IoT ecosystem: Analysis, implementation and lessons

Jaeho Kim[1,2] · Sung-Chan Choi[2] · Jaeseok Yun[2] · Jang-Won Lee[1]

Received: 14 March 2016 / Accepted: 8 September 2016 / Published online: 30 September 2016

Abstract As the Internet of Things (IoT) revolution presents an enormous opportunity for all industry verticals ranging from startups to large enterprises to create new types of services, standard bodies and global alliances have been working on establishing common standards for IoT systems. The oneM2M is the global partnership developing standards for Machine-to-Machine (M2M) communications and the Internet of Things. It develops technical specifications for the globally-applicable, interoperable common M2M/IoT service layer platforms, which play a pivotal role in building the ecosystem driven by key players, including developers and consumers. In this paper, we analyze the oneM2M standards, and introduce Mobius and &Cube, which are oneM2M-compliant M2M/IoT software platforms for servers and devices, respectively. We also present four pilot services using the platforms and several prototype IoT devices. Finally, we discuss three aspects, advanced discovery, open API, and peer-to-peer that are required for the oneM2M to build IoT ecosystem by attracting developers and consumers into the emerging IoT ecosystem.

Keywords Internet of Things · oneM2M standards · IoT platforms · Mobius · &Cube · Discovery · Open API · Peer-to-Peer · IoT ecosystem

✉ Jaeho Kim
jaehokim@yonsei.ac.kr; jhkim@keti.re.kr

Sung-Chan Choi
csc@keti.re.kr

Jaeseok Yun
jaeseok@keti.re.kr

Jang-Won Lee
jangwon@yonsei.ac.kr

[1] School of Electrical and Electronic Engineering, College of Engineering, Yonsei University, 50 Yonsei-ro, Seodaemun-gu, Seoul 120-749, South Korea

[2] IoT Platform Research Center, Korea Electronics Technology Institute, 25 Saenari-ro, Bundang-gu, Seongnam 463-816, South Korea

1 Introduction

Over the past years, the Internet of Things (IoT) has been an obvious trend and changed our everyday lifestyle with new types of smart devices and applications. Due to advances in the IoT enabling technologies including low-power sensors, wearables, embedded systems, smartphones, and wireless connectivity, things in the world can communicate each other and thus share information about the changes in their status and surroundings. Such Internet connectivity for 'things' allows developers and manufacturers to build creative products and entirely new services, which can benefit people in an *intelligent* and *proactive* manner. In other words, IoT gives us a chance to extend long-established Internet-based services beyond existing business models.

The main obstacle for building the emerging IoT ecosystem will be certainly the lack of standardized infrastructure that enables IoT systems to be interoperated across different industry verticals. Several studies have been conducted to build an integrated and interoperable architecture for smart grid systems [1, 2] and peer-to-peer (P2P) network and communication [3]. Similarly, anticipating the new opportunities and challenges by the emerging IoT industries, global tech companies and standards development organizations

(SDOs) have established an international partnership project for IoT standards, the oneM2M Global Initiative, in 2012 and announced the first release of oneM2M specifications in 2015 [4].

Since the initial release of oneM2M specifications, we have developed oneM2M-compliant IoT software platforms, Mobius and &Cube, which serve as IoT service server platform and device software platform, respectively. The Mobius offers common service functions required by oneM2M specifications and protocol bindings for hypertext transfer protocol (HTTP), message queuing telemetry transport (MQTT), constrained application protocol (CoAP). We also develop additional parts (i.e., non-oneM2M features) designed for establishing a global IoT ecosystem, including global discovery, IoT application store, mashup API organization, and open API scheme. The &Cube offers common service functions that enable IoT devices to communicate with the Mobius. Also, additional features are developed for managing things connected (i.e., sensors and actuators) and applications running on the & Cube.

In this paper, we introduce the oneM2M standards and the full details on our oneM2M platforms, Mobius and &Cube, focusing on how we implement their common service functions and what functions we have additionally developed. We also present several prototype devices and pilot services to show the practical usability of the platforms. From the lessons of the pilot services and experience on collaborating with developers and manufacturers, we also discuss the future direction for the oneM2M standards to build IoT ecosystem.

The Mobius and &Cube have been under test in our institute over the past years, and SK Telecom Co., a telecommunication operator in S. Korea, has launched a commercial platform based on the Mobius, called ThingPlug in 2015. In addition, we have established an open source-based partnership, OCEAN (Open allianCE for iot stANdards), in order to provide alliance members with the open source of the Mobius and &Cube [5]. We believe that such open source platforms based on IoT standards will help startups and small and medium-sized enterprises (SMEs) collaborate across different industry verticals but also create new IoT products and services.

The remaining organization of the paper is as follows. Section 2 presents various IoT enabling technologies and standards. Section 3 introduces oneM2M standards and important features. In Section 4, we demonstrate our efforts to build IoT ecosystem, including oneM2M IoT platforms (Mobius and &Cube), prototype devices and services, and experiences for IoT ecosystem. In Section 5, we summarize the lessons from our experience, and Section 6 offers concluding remarks.

2 IoT enabling technologies and standards

In IoT environment, we need to have standardized technologies in order to prevent missing interoperability between IoT devices and services. Thus, the IoT standards development has begun and those works mainly came from standard development organizations (SDOs) as well as industry alliances. Besides, companies with big market power like Apple and Google have developed and promoted their products to attract and keep customers in the emerging IoT ecosystem.

The oneM2M started as a global partnership project established by 7 SDOs and 5 industrial consortia in 2012. The oneM2M develops technical specifications for globally-applicable and access-independent M2M/IoT solutions, and it has standardized Release 1 specifications in January 2015 [4]. Likewise, the 3GPP (3rd Generation Partnership Project) was organized from 7 SDOs as a global partnership whose area is mainly mobile communication technology. The 3GPP defines machine-type communication (MTC) specifications enabling machines to communicate each other on cellular networks. The IETF (Internet Engineering Task Force)'s objective is to develop Internet protocols and IoT-related technologies. It has been developing various Internet protocols like 6LoWPAN [6] and CoAP [7] to use in constrained environments. Major device management technologies such as the open mobile appliance device management (OMA-DM) and broadband forum (BBF) TR-069 have been developed.

Industry alliances have been working on IoT standards as well. The Thread Group started by Google in 2014 focuses on the networking protocol with secure and low-power features using IPv6 technology with 6LoWPAN. The AllJoyn is a software framework announced by AllSeen Alliance, which provides the functions of discovery, connectivity, and interaction between devices on a same Wi-Fi network. A rival consortium, Open Interconnect Consortium (OIC) was launched in 2014 by Intel and Samsung Electronics. The target is to develop standards and open source softwares for IoT connectivity and resource framework supporting mainly P2P communication in local area network like a home environment.

Without pursuing a unified standard, there is another activity led by tech companies with popular commercialized products and big market power. For example, in 2014, Apple released HomeKit framework, a smart home platform for iOS through which every home appliance can be connected and controlled. Google also started the IoT business by acquiring Dropcam and Nest Labs.

To all appearances, it currently looks like all organizations and alliances are contending each other. However, as the IoT ecosystem grows, each IoT standard technology and industry market will find a better way to collaborate one another. As such an effort, the oneM2M provides common service platforms which cover various vertical domains, and it also makes progress on interworking with other standards e.g., AllJoyn and OIC. In this view, the oneM2M already shows a global standard collaboration model, and also echoes our motivation that we have developed oneM2M-based platforms to build a global IoT ecosystem.

3 Overview of oneM2M standards

There exist many vertical business service domains including, energy, healthcare, transportation, residence, etc. Thus, vertical M2M/IoT solutions have been designed independently for different applications, which inevitably cause the fragmentation of M2M service solutions and hinder a large-scale M2M deployment. To combat with fragmentation and make a healthy ecosystem with economies of scale, the oneM2M has made efforts to standardize a common service layer platform for globally applicable and access-independent M2M/IoT services. Specifically, the oneM2M has six working groups (WGs): WG1-REQ for use cases and requirements; WG2-ARC for architecture; WG3-PRO for protocols; WG4-SEC for security; WG5-MAS for management, abstraction and semantics; WG6-TST for testing. Swetina et al. summarize well the organization and standardization activities for the oneM2M WGs [8].

In this section, we give an overview of the oneM2M standards released in January 2015 (Release 1) in several aspects: functional architecture, common services functions, addressing and discovery, communication protocols, and security. Also, we describe several ongoing work items (WIs) for the next version of technical specifications (Release 2).

3.1 Functional architecture

The oneM2M initiative aims to develop common technical specifications for IoT platforms applicable across different industry verticals. To this end, oneM2M collected use cases from a wide range of vertical business service domains. Using the collected use cases, oneM2M formulated requirements for the oneM2M service layer [9], and then designed the system architecture [10].

As shown in Fig. 1, the oneM2M architecture divides M2M/IoT environments into two domains (infrastructure domain and field domain), and defines four types of nodes: infrastructure node (IN), middle node (MN), application service node (ASN), and application dedicated node (ADN), where nodes are logical entities identifiable in the M2M systems [10, 11]. Exactly an IN can be located in the infrastructure domain of any given M2M service provider whereas its field domain can contain any group of oneM2M nodes including MN, ASN, and ADN, or even non-oneM2M nodes.

For the node architecture, oneM2M defines a layered model for supporting M2M/IoT services, composed of application layer, common services layer, and underlying network services layer, as shown in Fig. 2. Each layer is represented as an entity model using application entity (AE), common services entity (CSE), and network service entity (NSE), respectively. An AE represents an entity in the application layer residing in a number of nodes and providing various service logics. A CSE stands for an instantiation of a set of common service functions (CSFs) of the M2M environments, which will be explained in more detail in the next section. A NSE provides network services from the underlying network to the CSEs.

For interconnecting various entities in the oneM2M system, the oneM2M defines four reference points for communication flows between entities as shown in Fig. 1: Mca (M2M Communication with AE); Mcc (M2M Communication with CSE); Mcn (M2M Communication with NSE); Mcc' (M2M Communication with CSE of different M2M Service Provider).

3.2 Common services functions (CSFs)

CSFs are groups of common service capabilities residing in a CSE, which can be shared by different applications. As shown in Fig. 2, CSFs include registration, data management and repository, device management, security, communication management and delivery handling, discovery, subscription and notification, service charging, and location, etc. We introduce the formal definition and responsibilities of the CSFs (summarized from TS-0001 [10]):

- Application and Service Layer Management: provides management of the AEs and CSEs on the oneM2M nodes such as IN, MN, ASN, and ADN.
- Communication Management and Delivery Handling: provides communications with other entities including AEs, CSEs, and NSEs.
- Data Management and Repository: provides data storage and mediation function.
- Device Management: provides management capabilities for devices on the oneM2M nodes in the field domain such as MN, ASN, and ADN.
- Discovery: provides capabilities for searching specific resources or attributes containing information about oneM2M services and applications.

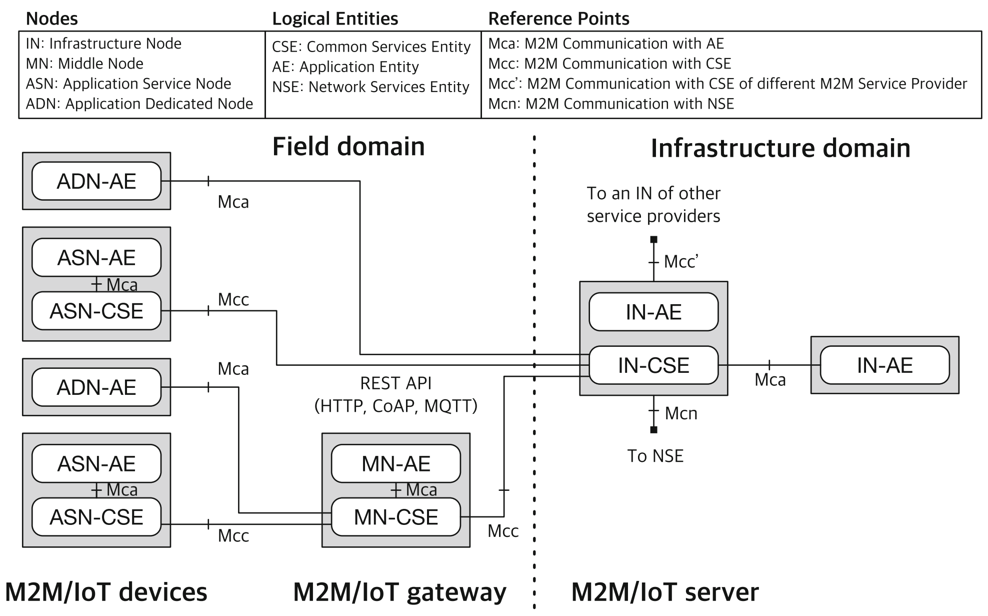
Nodes
IN: Infrastructure Node
MN: Middle Node
ASN: Application Service Node
ADN: Application Dedicated Node
Logical Entities
CSE: Common Services Entity
AE: Application Entity
NSE: Network Services Entity
Reference Points
Mca: M2M Communication with AE
Mcc: M2M Communication with CSE
Mcc': M2M Communication with CSE of different M2M Service Provider
Mcn: M2M Communication with NSE
Field domain
Infrastructure domain
ADN-AE
Mca
ASN-AE
Mca
ASN-CSE
Mcc
To an IN of other service providers
Mcc'
IN-AE
IN-CSE
Mca
IN-AE
Mcn
To NSE
ADN-AE
Mca
REST API (HTTP, CoAP, MQTT)
ASN-AE
Mca
ASN-CSE
Mcc
MN-AE
Mca
MN-CSE
Mcc
M2M/IoT devices
M2M/IoT gateway
M2M/IoT server

Fig. 1 The oneM2M reference architecture and interconnection between entities

- Group Management: provides capabilities for handling group related requests.
- Location: allows AEs to obtain geographical location information of oneM2M nodes (e.g., ASN and MN) for location-based services.
- Network Service Exposure, Service Execution and Triggering: manages communication with underlying networks for access network services via Mcn reference points.
- Registration: processes registration requests from AEs or other CSEs.
- Security: handle sensitive data handling, security administration, provisioning and administration of subscriptions, and security association establishment.
- Service Charging and Accounting: provides charging functions for the service layer, e.g., online real-time credit control.
- Subscription and Notification: provides notifications pertaining to a subscription keeping track of changes on an oneM2M resource (e.g., resource creation and deletion).

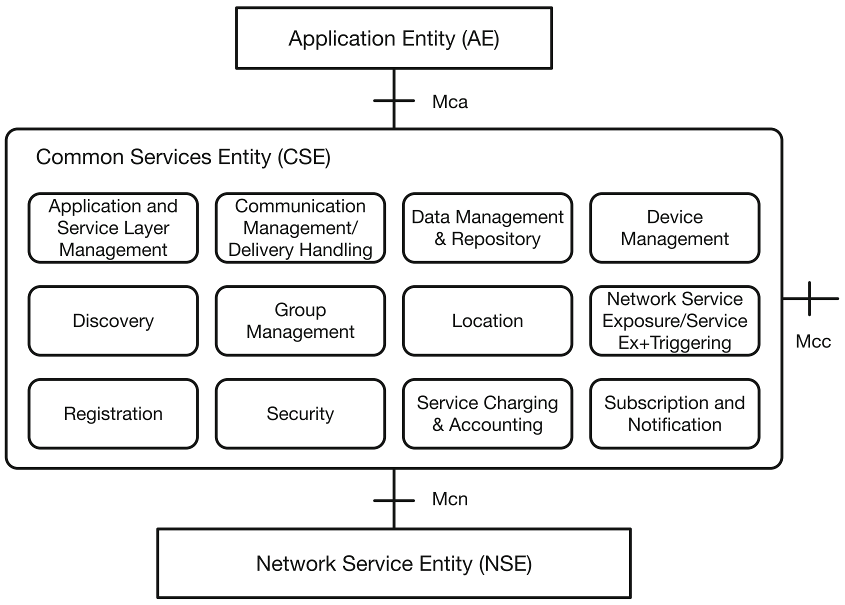
Application Entity (AE)
Mca
Common Services Entity (CSE)
Application and Service Layer Management
Communication Management/ Delivery Handling
Data Management & Repository
Device Management
Discovery
Group Management
Location
Network Service Exposure/Service Ex+Triggering
Registration
Security
Service Charging & Accounting
Subscription and Notification
Mcc
Mcn
Network Service Entity (NSE)

Fig. 2 The oneM2M layered model composed of AE, CSE, and NSE, and common services functions residing in a CSE

Such services are provided to AEs via Mca as well as to CSEs via Mcc reference points. An instantiation of a CSE can have a subset of CSFs described above.

3.3 Naming and addressing

oneM2M adopts a resource-oriented architecture (RoA) as a naming structure [12]. Thus, IoT devices, data, and services are represented by resource data model. In addition, each resource can be uniquely identified by its resource address where it supports two different methods for addressing: non-hierarchical and hierarchical way. Every resource in CSE has resource name as well as resource ID. For non-hierarchical addressing, each resource is addressable via its resource ID. In contrast, for hierarchical addressing, resource name is used for addressing its resource. Fig. 3 shows an example of oneM2M resource structures illustrated by a tree style. Therefore, when it comes to addressing a resource through hierarchical address, resource names are concatenated from parent to child of the target resource.

For enabling an actual connection establishment and delivering data between nodes in IoT systems, each node needs to know the peer's network address to access it. To this point, oneM2M introduces Point of Contact (PoC) mechanism for providing peer's network address. In registration process, oneM2M node provides its network address as a PoC attribute of its resource, and then the PoC can be used for finding actual network address as follows: when a node receives a message including a target address whose format is non-hierarchical or hierarchical name as above mentioned, it discovers the target peer and finds its PoC attribute for accessing it. Regarding that, oneM2M currently does not support a fully distributed name to address translation mechanism, and thus there needs more considerations regarding an optimization point such as how to handle name to network address translation efficiently as in [13].

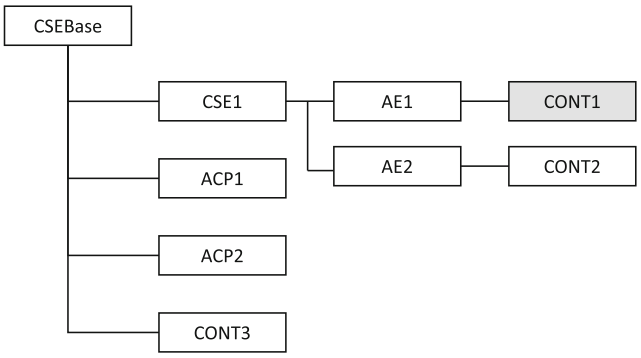

Fig. 3 Resource-oriented architecture (RoA) based oneM2M resource structure

3.4 Communication protocols

Based on the architecture-level design, the oneM2M defines a service layer protocol, including primitives (common service layer message formats), APIs, and message procedures exchanged over the Mca, Mcc, and Mcc' reference points [14]. After specifying the service layer protocol, the oneM2M also developed standardizations of protocol binding between the service layer protocol and underlying protocols. The primitives are mapped to application layer communication protocols like CoAP [15], HTTP [16], and MQTT [17], which use TCP or UDP as the transport layer. In addition, WebSocket binding has been developed and will be included in the oneM2M Release 2 standards. Comparing with other IoT platforms such as AllJoyn, OIC, and OMA lightweight M2M, oneM2M considers various protocols for enabling flexible operations according to the communication environment of IoT services. Furthermore, oneM2M currently started a study work item regarding data distribution service (DDS) and open platform communications (OPC) unified architecture (UA) protocol in order to consider real-time and reliable communication environment required in industrial domain.

3.5 Security and device management

The oneM2M standards considers security aspects as one of basic functions of a common service layer platform. To this end, oneM2M reflects security capabilities as follows: credential deployment and management, secure connection establishment and management, authorization, and access control [18]. In addition, for device management, the oneM2M specifies device abstraction resources to adapt the specific device management technologies like OMA DM [19] and BBF TR-069 [20].

3.6 Technical specifications release

The oneM2M initiative has issued its Release 1 technical specifications in January 2015. Table 1 summarizes published technical specifications (Release 1) categorized by WGs. WG6-TST was not organized for Release 1, so no TS was publically issued in 2015. All the TS documents are available in the oneM2M website [4].

Currently, several new work items for Release 2 are in progress. The new work items include interworking between oneM2M and other M2M/IoT standards like AllJoyn, OIC, and 3GPP, and enhanced security solutions covering end-to-end security and group authentication, and consideration for the wide-scale deployment like home/industrial domain enhancement. In addition, testing and certification related activity has started and the advanced features (e.g., big data analysis and semantic technology) are being discussed. All these considerations would help the oneM2M standards deliver a solution that builds a standard-based IoT ecosystem.

4 Our efforts to build IoT ecosystem based on oneM2M standards

We look into the IoT ecosystem from two sides: developers (e.g., hardware manufacturers, application developers) and end-users (i.e., consumers). For developers, it would be important to support developer enablement components (e.g., platforms, prototype devices, and APIs) that help them create new innovative products quickly and easily. For end-users, we first need to demonstrate appealing, easy-to-find/use IoT services to bring about their attraction that will eventually turn into sales. Fig. 4 shows our efforts to build an IoT ecosystem with regard to developers and end-users.

4.1 oneM2M-based IoT platforms: Mobius and &cube

Without standards, IoT solutions would be developed independently for different vertical domains, causing high fragmentation problems and increasing the overall cost for development. Thus, the oneM2M standards and its compliant platforms enable device manufacturers and software developers to develop IoT solutions using open APIs regardless of underlying networks, eventually facilitating interoperability of devices and softwares between vertical domains and boosting up the IoT ecosystem.

We have implemented three types of oneM2M-defined CSEs: IN (as a part of the Mobius), MN-CSE (&Cube:Rosemary), ASN-CSE (&Cube:Lavender), as shown in Fig. 5. The Mobius is a server platform located in the infrastructure domain. Mobius consists of an IN of oneM2M, GDP (global discovery platform), and ASP (IoT application store platform). The IN-CSE provides CSFs defined in the oneM2M. In addition, to interwork with exterior points, the Mobius provides bindings for HTTP, MQTT, and CoAP protocols. We now support certificate-based security solutions working with an additional authentication server. For managing data from hundreds of thousands of devices, the Mobius utilizes in-memory database Redis and NoSQL database MongoDB in a hybrid way. With this mechanism, the Mobius provides a very efficient method through which a large amount of data can be handled in a short time. Additionally, the IN of the Mobius provides the functionality of creating mashup API with which users can efficiently access to resources in the Mobius. The way of forming a new mashup API composed of random number and text will enhance security by not exposing the resource topology in the Mobius and it can also provide APIs with shortened URL.

The GDP supports interworking with other INs in different service providers by collecting their metadata with open APIs. Those metadata information can be used for discovery mechanism afterwards. The GDP also supports real-time indexing regarding metadata saved in the database, and provides various discovery criteria like keywords, location, access rights, and ID. The GDP manages metadata for devices as well as for users and services, e.g., registration, profile, access authority of users and services. Additionally, it provides a function for managing a 'Topic' under which a community of users and devices can be managed as a group.

The ASP supports the marketplace in IoT environment, which can enlarge the IoT application ecosystem from the existing smartphone application ecosystem. Developers can publish their application, which can monitor and control an IoT device, to the ASP via the App Management component in Fig. 5. Once published, the application will be managed in the App Repository component. Users can discover and download the apps using the App Discovery component and the Download Management component.

For the field domain, we have developed two types of IoT device software platforms (i.e., &Cube) for providing a base for IoT gateways and devices: &Cube:Rosemary and &Cube:Lavender. The &Cube:Rosemary is designed to be a MN-CSE in oneM2M standards, acting as a gateway for IoT devices. In contrast, the &Cube:Lavender is developed to be an ASN-CSE in oneM2M standards, and be able to directly interwork with Mobius as well as through the &Cube:Rosemary.

4.2 Prototype devices and pilot services

For developers to get started with our oneM2M platforms, we have been running the Mobius with a public domain

Table 1 oneM2M technical specifications (Release 1) published in January 2015

WGs	Reference	Title
WG1	TS 0002	Use cases and requirements
	TS 0011	Common Terminology
WG2	TS 0001	Functional architecture
WG3	TS 0004	Service layer core protocol specification
	TS 0008	CoAP protocol binding
	TS 0009	HTTP protocol binding
	TS 0010	MQTT protocol binding
WG4	TS 0003	Security solutions
WG5	TS 0005	Management enablement (OMA)
	TS 0006	Management enablement (BBF)

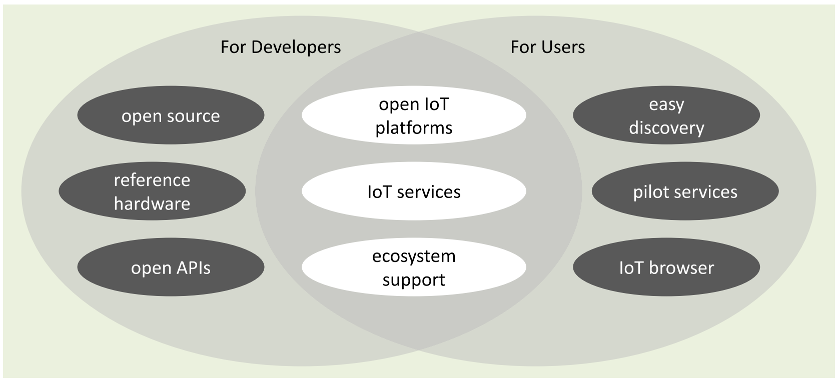

Fig. 4 Our efforts to build an IoT ecosystem from developer and end-user perspectives

name (http://iotmobius.com), and tested the &Cube on various Linux-based embedded devices including Raspberry-Pi. We have also developed several prototype devices that can be used for developing IoT services. Finally, to show the vision of the oneM2M standards and the practical usability of our platforms, we have implemented four pilot services.

4.2.1 Prototype devices

Fig. 6 shows five embedded hardware devices with which we have tested &Cube and five prototype devices.

– IoT gateway devices
  Because we developed the &Cube as a middleware running on the Java Virtual Machine (JVM), it would be easy for developers to install it to any embedded system with a JVM. We have performed testing with three open hardware kits including Raspberry-Pi (Fig. 6a), BeagleBone Black (Fig. 6b), and IoTG100 from CRZ tech (Fig. 6c) as well as 3G (Fig. 6d) and LTE (Fig. 6e) modules.
– IoT devices
  We have developed five types of IoT devices, including multi-functional sensors, smart plugs, wireless

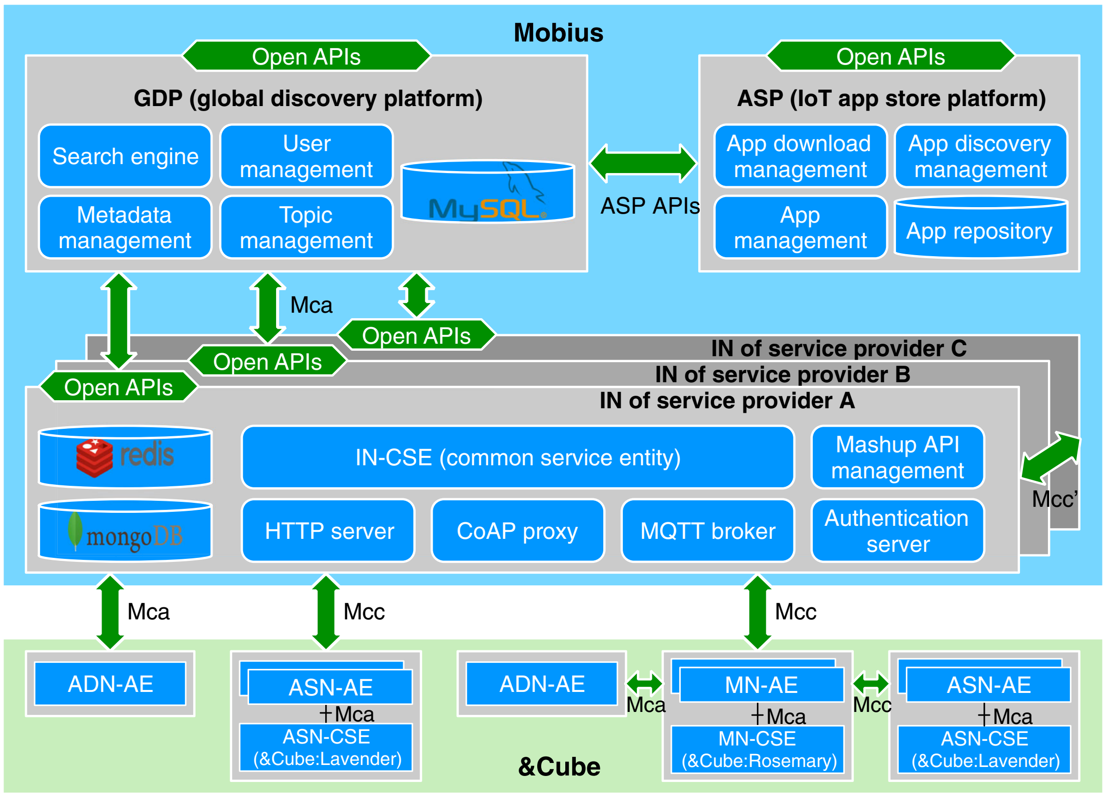

Fig. 5 Overall configuration of oneM2M-compliant platforms, Mobius and &Cube

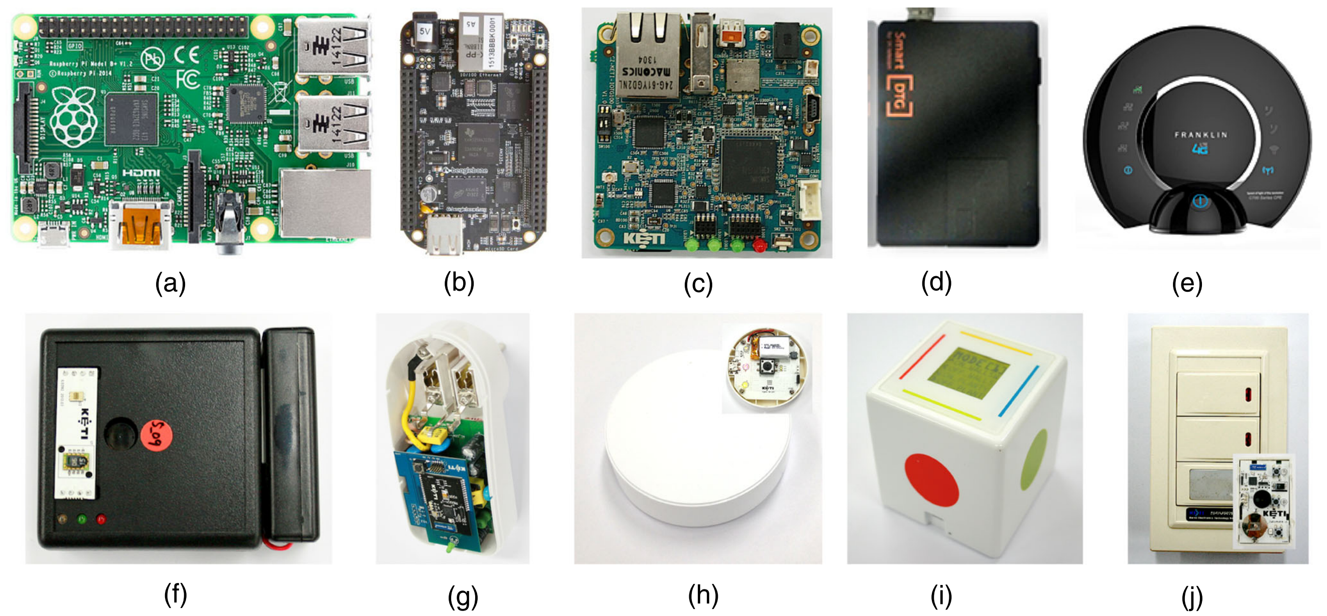

Fig. 6 IoT gateway devices running &Cube (a)-(e) and prototype IoT devices (f)-(j), (a) Raspberry-Pi, (b) BeagleBone Black, (c) IoTG100, CRZ tech, (d) NEO-W100 3G router, MDS tech, (e) BPL-R300 LTE router, B&P, (f) multi-functional sensor for measuring temperature, humidity, illumination, and occupancy, (g) smart plug, (h) wireless switch, (i) smart dice, (j) wall light switch

switches, smart dices, and wall light switches. Multi-functional sensors can measure temperature, humidity, illumination, and occupancy (Fig. 6f). Smart plugs can be used as both sensors and actuators (Fig. 6g). A smart plug can measure electrical power consumed in an electrical device that it was plugged in, and can instantly turn on/off the device. For the easy-to-use input interface, we have developed wireless switches (Fig. 6h) and smart dice (Fig. 6i). Wireless switches command smart plugs to turn electrical devices on/off just like a remote controller. The smart dice embedded with a 3-axis accelerometer provides six different outputs by positioning each sensitive axis (X, Y, Z) of the accelerometer at +g and g, each of which can command smart plugs or another devices to perform a task. For wireless wall light switches (Fig. 6g), we have modified a commercially-available IR remote control wall light. All devices use a Texas Instruments SoC solution for IEEE 802.15.4 applications, CC2530, to provide wireless connectivity to IoT gateway devices.

4.2.2 Pilot Services

We have developed four pilot services: iThing, TTEO, Planty, and iDrone. We have uploaded each demonstration video on YouTube, and Fig. 7 illustrates the still images captured from the videos.

– iThing
  The iThing provides a voice-based home control service using Android's SpeechRecognizer API converting a user's verbal command to text via Android phones. According to the interpretation of the user's command, the Mobius will send appropriate control signals (i.e., on/off) to smart plugs (i.e., home appliances). The iThing can also be used to trigger user-defined actions for the connected home, for example, "sleep mode!" (see Fig. 7a and the video, http://youtu.be/6pe1HdpUOnA).
– TTEO
  The TTEO is a rule-based home automation service that allows users to program the connected home by setting rules for IoT devices to automate daily tasks without human intervention. The TTEO app provides a GUI with which users create simple but powerful if-then rules between home appliances working with our platforms. This easy-to-use GUI-based interaction method will clearly help people deploy and configure their connected homes without any technical or programming knowledge (see Fig. 7b and the video, http://youtu.be/9Veka6C2FrE).
– Planty
  The Planty is a personal connected flowerpot that allows users to remotely monitor and grow their flower. It includes sensors that measure temperature, light, and

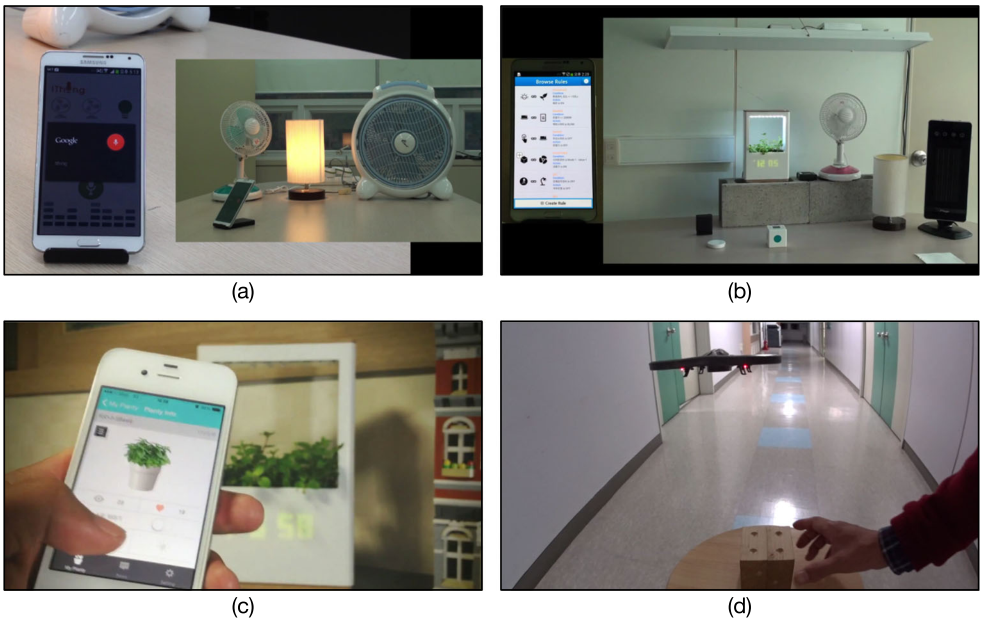
Fig. 7 Captured images of the developed pilot services: (a) iThing, (b) TTEO, (c) Planty, (d) iDrone

soil moisture as well as actuators that control light and water for plants. All functions can be automated via its smartphone app (see Fig. 7c and the video, http://youtu.be/xdMzjYU1xyM).

– iDrone

The iDrone is a connected drone (i.e., unmanned aerial vehicle) that we can control in (almost) real-time. Considering the hot trend of integrating IoT with drones, we have developed a smartphone app that sends control commands to a Parrot AR.Drone using its open APIs and our platforms. We have also created an application to control the drone by flipping over the smart dice (see Fig. 7d and the video, http://youtu.be/azMP0_XY3-M).

4.3 Performance analysis

We conduct simulation tests to show stable operation of the developed IoT platforms in terms of three performance indicators: the number of devices that can be registered, average response time and transactions per second (TPS) for requests of resource creation and retrieval. We use Apache JMeter, a performance measuring tool for Web applications, to test functional behaviors and measure the performance indicators we have chosen.

First, the device registration test shows that we can successfully register over 300,000 of IoT devices with the Mobius. Next, we perform two tests for measuring average response time and TPS when 300,000 of registered IoT devices simultaneously request resource creation or resource retrieval. Table 2 shows the result of simulation tests. Both tests show around 50 msec of average response time and over 300 TPS transaction processing capability.

4.4 Experiences for the IoT ecosystem

Besides developing IoT platforms and pilot services, we have also made some efforts for establishing the IoT ecosystem. For developers, we opened the core source codes of the Mobius and &Cube to disseminate those oneM2M platforms and to lower barriers for developers to create oneM2M-compliant IoT products. To support these efforts, we established an open-source based partnership, OCEAN [5], to leverage collaboration among different verticals by disseminating the open source codes and guidelines for developing IoT services. Another keystone for developers in the IoT ecosystem will be an open API-based development framework because it will help developers separately develop their products and applications just like a smartphone ecosystem. Thus, we have built a web portal, called the Open API Site, for sharing the information about open APIs for IoT devices [21].

From the perspective of consumers, in the emerging IoT era, we will be surrounded by countless IoT devices, and probably need a way of browsing them near the place where we are or we want to explore. To this end, we have developed the IoT Browser, a map-based smartphone application for searching IoT devices worldwide. It can display the IoT device's location on the map (currently available for Google, Daum, Naver Map), and provide capabilities for

Table 2 Result of simulation tests for resource creation and resource retrieval

	# of requests	Average response time (msec)	TPS
Resource creation	190,869	42	317.4
Resource retrieval	190,887	53	317.3

discovering IoT devices in terms of ID, topic, keyword, address, and location by working together with the Mobius's open APIs. The IoT Browser is also designed to work as an app launcher, which allows us to select the IoT device we want to connect, and then run (or download if not installed) its compatible smartphone app. The IoT Browser can be found at the iOS and Android application stores [22, 23].

5 Lessons from experience to support the IoT ecosystem

In this chapter, we discuss three key features which will be required for the oneM2M to build a global IoT ecosystem: advanced discovery, open API and peer-to-peer support.

5.1 Advanced discovery

In IoT environments, there are a myriad of devices and huge data resources as well, so providing an efficient discovery mechanism is considered as one of key technologies. Edwards compared existing discovery systems and suggested new challenging directions in ubiquitous computing, including scalability, seamless interconnectivity covering different protocols, and enhanced searching mechanism e.g., context-awareness [24]. Meshkova et al. [25] presented an extensive survey on service discovery frameworks in various network scales from local-area to internet-scale. Zhu et al. [26] analyzed the design of service discovery protocols and compared them according to several criteria like discovery scope (network topology and administrative domain), the way of discovery and registration (query- and announcement-based), communication method (unicast, multicast, and broadcast). They also pointed out that high-level context information like temporal, spatial, and user activity information can save users' time and effort in discovering services. Those literatures stress scalability and context-awareness of discovery systems as the key features for enabling seamless IoT services.

In oneM2M standards, to support discovery, there exists the discovery CSF which performs search mechanism regarding information which represents applications and services using resources in the directory administrated by its service provider by matching the filter criteria like a combination of time, keywords, type and size of contents. However, oneM2M systems only support directory-based discovery, but not peer-to-peer based discovery and context-aware discovery. In the Mobius, the directory can expand its scope by interlinking directories. As explained in Section 4.1, the GDP can collect discovery information regarding the resources located in distributed oneM2M platforms of other service providers. The Mobius also supports a way to specify location information representing various physical locations on the map, working with street view and augmented reality.

Cirani et al. [27] implemented a scalable and self-configuring, peer-to-peer based architecture for large scale IoT networks, aiming at providing an automated service and resource discovery mechanism. They highlighted the importance of scalability regarding service discovery in the IoT system, and suggested a combining mechanism considering both Zeroconf-based local scope discovery and P2P-based large-scale service discovery. As the Cirani's example explains, and learned from our work, we believe that a well-established global discovery method will be pivotal in improving accessibility to IoT services. The aspect for better accessibility is to provide users with an easy-to-find way of discovering the IoT services they want. Thus, we need to bring peer-to-peer, scalability and context-awareness of discovery for IoT devices within the scope of the next oneM2M standards.

5.2 Open API

Several literatures presented the necessity of providing standardized open APIs in a network platform. Mulligan [28] summarized the activities of Google's Open APIs and platforms (i.e., OpenSocial and Android) for the developer community, and highlighted the direction of open APIs for the next generation networks (NGNs) and their standardization. Sneps-Sneppe and Namiot [29] introduced the progress of open APIs for M2M applications, in particular presented as ETSI standards, and proposed a framework for client-side service discovery and inter-application communication for M2M systems to be able to find appropriate services based on the user's preference. Fraunhofer FOKUS's team [30, 31] developed its own M2M platform (called OpenMTC) together with a set of RESTful APIs that enable 3rd party applications to access the platform. More recently, they extended the OpenMTC architecture to support a software development kit (SDK) that enables 3rd party developers to create their own M2M applications.

As in the new OpenMTC architecture, the Mobius is designed to support RESTful APIs available for developers to use oneM2M CSFs. In addition, as explained in Section 4.4, Mobius allows device makers to create an IoT device together with its open APIs. The device and API information can be open via our web portal, Open API Site [21], so 3rd party application developers will be able to develop new applications for the device using the APIs' name and reference to the associated resources in the Mobius platform. Accordingly, such developer enablement resources will potentially help the developer community create new IoT products in an efficient way.

Likewise, the latest trend in the IoT market shows the role of open APIs to accelerate the involvement of 3rd party software developers. One example is Philips Hue [32], the Internet-connected LED lightbulb controllable via smartphone applications. Surprisingly, in 2013, Philips released its open APIs and SDK for iOS developers (now also available for Android), and guides for both hardware and software makers [33]. Now, it is possible to see more than 90 iOS apps (even more Android apps) working with Hue created by all different developers.

As the Hue example explains, and learned from the Mobius implementation work, open APIs enable the rapid creation of new IoT services by bringing device makers and 3rd party application developers together via common service platforms. Thus, this open innovation strategy combined with open APIs ensures that we need to bring open APIs supporting for IoT devices (just like Hue) within the scope of the oneM2M standardization activities.

5.3 Peer-to-Peer support

Wu et al. presented a valuable survey on M2M/IoT systems, and introduced a high-level IoT system architecture [34]. According to the literature, M2M/IoT systems are divided into two different ones: hierarchical architecture and peer-to-peer system architecture. A hierarchical system based on a client-server model (e.g., oneM2M system) offers an effective solution in IoT service environments. However, there are several issues to be addressed.

First, a huge number of things joined to IoT services lead to the explosion in the volumes of data collected and exchanged. The explosion of data traffic causes network and processing bottlenecks on the server side in the systems based on a client-server model. Second, the client-server model is not robust to large scale IoT services. If the server providing IoT services fails, the whole IoT service system goes down. Lastly, systems based on the client-server model manage the personal data (e.g., user profile and health information) on storages of their server. It could cause a serious privacy issue–for example, due to cyber attacks, and thus users do not prefer or agree that their personal data will be filed into the central server.

To alleviate those problems, we need to take advantage of peer-to-peer systems. Markatos in his research showed that peer-to-peer computing and networking not only enables clients to take a more active role in the information dissemination process, but also may significantly increase the performance and reliability of the overall system, by eliminating the traditional notion of the 'server' which could be a single point of failure, and a potential bottleneck [35]. In addition Chae et al. proposes a privacy data leakage prevention method in P2P networks [36]. The proposed method can prevent a privacy data from being leaked by releasing a P2P sharing information without privacy data using a privacy data removing technology.

Consequently, it will be carefully considered to take peer-to-peer features into the next release of oneM2M standards, and the following issues need to be tackled: how to support the zero configuration; how to add P2P discovery into the previous discovery mechanism; how to support P2P communication between ASN-CSEs using Mcc interface.

6 Conclusion

Although standard bodies and big tech companies now seem to eagerly compete with each other to dominate the emerging IoT market, they will probably and eventually agree to make their standards talk with each other to broaden their IoT market share. We believe the oneM2M is at the core of the overall vision as it will lead the collaboration together with industry alliances like AllSeen Alliance and OIC, and tech giants like Google and Apple. We have demonstrated our works based on oneM2M standards including oneM2M-compliant platforms, pilot services, and additional efforts for an IoT ecosystem. We have also presented three key aspects for the oneM2M to support the IoT ecosystem including advanced discovery, open API and P2P system support. Through our efforts, we hope to help accelerate the diffusion of oneM2M standards, and thereby build an unfragmented global IoT ecosystem by successfully attracting both developers and consumers.

Acknowledgments Kim, Choi, and Yun were supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No.B0184-15-1003, Development of oneM2M Conformance Testing Tool and QoS Technology) Lee was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIP) (No.B0717-16-0024, Development on the core technologies of transmission, modulation and coding with low-power and low-complexity for massive connectivity in the IoT environment).

References

1. Mulla A, Baviskar J, Yerunkar A, Sarwadnya R (2015) Convergence of Wireless Sensor Network with Smart Grid Environment Based on IPv6 Protocol. In: Proceedings of the Fifth International Conference on Communication Systems and Network Technologies (CSNT '15), pp 153–158
2. Albano M, Ferreira LL, Pinho LM (2015) Convergence of Smart Grid ICT Architectures for the Last Mile. IEEE Transactions on Industrial Informatics 11:187–197
3. Jo S-M, Kim G, Han J (2016) Convergence P2P context awareness. Peer-to-Peer Netw Appl 9:461–464
4. oneM2M. http://www.onem2m.org/technical/published-documents (2016). Accessed 9 March 2016
5. OCEAN (Open allianCE for iot stANdard) http://iotocean.org (2016). Accessed 9 March 2016
6. Shelby Z, Bormann C (2009) 6LoWPAN: The wireless embedded Internet. Wiley
7. Bormann C, Castellani AP, Shelby Z (2012) CoAP: An Application Protocol for Billions of Tiny Internet Nodes. IEEE Internet Comput 16:62–67
8. Swetina J, Lu G, Jacobs P, Ennesser F, Song J (2014) Toward a standardized common M2M service layer platform: Introduction to oneM2M. IEEE Wirel Commun 21:20–26
9. oneM2M. TS-0002-Requirements-V-1.0.1. Technical Specification (2015)
10. oneM2M. TS-0001-Functional-Architecture-V-1.6.1. Technical Specification (2015)
11. oneM2M. TS-0011-Common-Terminology-V-1.2.1. Technical Specification (2015)
12. Guinard D, Trifa V, Wilde E (2010) A resource oriented architecture for the web of things. Internet of Things (IOT), pp 1–8
13. Amoretti M, Alphand O, Ferrari G, Rousseau F, Duda A (2014) DINAS: a DIstributed NAming Service for All-IP Wireless Sensor Networks. In: Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '14), pp 2781–2786
14. oneM2M. TS-0004-Service-Layer-Core-Protocol-Specification-V-1.0.1. Technical Specification (2015)
15. oneM2M. TS-0008-CoAP-Protocol-Binding-V-1.0.1. Technical Specification (2015)
16. oneM2M. TS-0009-HTTP-Protocol-Binding-V-1.0.1. Technical Specification (2015)
17. oneM2M. TS-0010-MQTT-Protocol-Binding-V-1.0.1. Technical Specification (2015)
18. oneM2M. TS-0003-Security-Solutions-V-1.0.1. Technical Specification (2015)
19. oneM2M. TS-0005-Management-Enablement(OMA)-V-1.0.1. Technical Specification (2015)
20. oneM2M. TS-0006-Management-Enablement(BBF)-V-1.0.1. Technical Specification (2015)
21. Open API Site. http://programmable-things.net (2016). Accessed 9 March 2016
22. IoT Browser for Android, https://play.google.com/store/apps/details?id=com.einsware.iot_browser (2016). Accessed 9 March 2016
23. IoT Browser for iOS, https://itunes.apple.com/us/app/iot-browser-search-explore/id727763099 (2016). Accessed 9 March 2016
24. Edwards WK (2006) Discovery Systems in Ubiquitous Computing. IEEE Pervasive Comput 5:70–77
25. Meshkova E, Riihijrvi J, Petrova M, Mhnen P (2008) A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. Comput Netw 52:2097–2128
26. Zhu F, Matt WM, Lionel MN (2005) Service discovery in pervasive computing environments. IEEE Pervasive Computing 4:81–90
27. Cirani S, Davoli L, Ferrari G, Leone R, Medagliani P, Picone M, Veltri L (2014) A Scalable and Self-Configuring Architecture for Service Discovery in the Internet of Things. IEEE IoT J 1:508–521
28. Mulligan CEA (2009) Open API Standardization for the NGN Platform. IEEE Commun Mag 47:108–113
29. Sneps-Sneppe M., Namiot D. (2012) M2M Applications and Open API: What Could Be Next? Internet of Things, Smart Spaces, and Next Generation Networking:429–439
30. Elmangoush A, Magedanz T, Blotny A, Blum N (2012) Design of RESTful APIs for M2M services. In: Proceedings of the 16th International Conference on Intelligence in Next Generation Networks (ICIN '12), pp 50–56
31. Elmangoush A, Steinke R, Al-Hezmi A, Magedanz T (2014) On The Usage of Standardised M2M Platforms for Smart Energy Management. In: Proceedings of the International Conference on Information Networking (ICOIN '14), pp 79–84
32. Philips Hue LED lightbulb. http://www.meethue.com (2016). Accessed 9 March 2016
33. Philips Hue Developer Program. http://www.developers.meethue.com (2016). Accessed 9 March 2016
34. Wu G, Talwar S, Johnsson K, Himayat N, Johnson KD (2011) M2M: From Mobile to Embedded Internet. IEEE Commun Mag 49:36–43
35. Markatos EP (2002) Tracing a Large-scale Peer to Peer System: an Hour in the Life of Gnutella. In: Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID '02)
36. Chae C-J, Shin Y, Choi K, Kim K-B, Choi K-N (2016) A Privacy Data Leakage Prevention Method in P2P Networks. Peer-to-Peer Netw Appl 9:508–519

Jaeho Kim is a managerial researcher and a team leader in IoT Platform Research Center at the Korea Electronics Technology Institute. He is now serving as IoT Convergence Service Project Group chair of TTA and Device Working Group chair of Korea IoT Association. He received the BS and MS degrees from the Hankuk University of Foreign Studies. Currently, he is a Ph.D. candidate in the electrical and electronic engineering from the Yonsei University. His research interests are in the areas of wireless sensor networks, medium access protocols, and Internet of Things.

Sung-Chan Choi is a senior researcher in the IoT Platform Research Center at the Korea Electronics of Technology Institute (KETI). He received the BS and MS degrees in Electrical and Electronic Engineering from the Yonsei University, South Korea, in 2006 and 2008, respectively. Before he joined the KETI, he worked as a Research Staff at the Samsung Advanced Institute of Technology in Giheung, South Korea for 5 years. His research interests are in the area of QoS for Internet of Things, Network Optimization, and Flying Adhoc Networks.

Jaeseok Yun is a senior researcher in the IoT Platform Research Center at Korea Electronics Technology Institute (KETI). Prior to his current position, he worked as a postdoctoral research scientist in the Ubiquitous Computing Research Group in the School of Interactive Computing at Georgia Institute of Technology, GA, USA. He earned his M.S. and Ph.D. in Mechatronics from Gwangju Institute of Science and Technology (GIST). His research interests include ubiquitous computing, Internet of Things (IoT), embedded systems, and human-computer interaction.

Jang-Won Lee is a Professor of the School of Electrical and Electronic Engineering, Yonsei University, Seoul, Korea. He received his B.S. degree in Electronic Engineering from Yonsei University, Seoul, Korea in 1994, M.S. degree in Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea in 1996, and Ph.D. degree in Electrical and Computer Engineering from Purdue University, West Lafayette, IN, USA in 2004. In 1997-1998, he was employed with Dacom R&D Center, Daejeon, Korea. In 2004-2005, he was a Postdoctoral Research Associate in the Department of Electrical Engineering at Princeton University, Princeton, NJ, USA. Since September 2005, he has been with the School of Electrical and Electronic Engineering at Yonsei University, Seoul, Korea. He is a senior member of IEEE. His research interests include resource allocation, QoS and pricing issues, optimization, and performance analysis in communication networks, and smart grid.