Service Platform for Integration of Various M2M/IoT System

Kieu-Ha Phung, Hieu Tran, Vinh Tran-Quang*

School of Electronic Telecommunications, Hanoi University of Science and Technology

Received: June 22, 2019; Accepted: June 22, 2020

Abstract

The oneM2M is an international partnership project, established to aim at developing the global standard service platform for the Machine-to-Machine (M2M) as well as Internet of Thing (IoT). It promises to allow integration of various communication technology and application protocol in only consolidated open system while in the market, most of M2M/IoT solution are highly vertical and proprietary. oneM2M also can be developed in edge computing environment to process real-time numerous IoT devices' data. This work provides an overview of oneM2M and several actual implementation results that enable interoperating the disparate layer protocol and multiple technology in a simple scenario.

Keywords: oneM2M, M2M communication, IoT, testbed, edge computing

1. Introduction

The number of M2M devices is dramatically increasing and expected to reach to 20.0 billion in 2020. But at present, especially in industries, most M2M solutions provide customers with proprietary systems, which involve all layers, from application layer to physical layer, and specialized services. That results in limitations in the system extension supporting new services integrating different technologies and the interoperability of various M2M systems. It also makes difficulties in scalability, flexibility, and fault tolerance. Therefore, there is a strong demand to establish a common M2M service platform from various standard organizations. oneM2M is one of solution for such platform, which is expected to bridge the gap between individual technology and the platform.

In this article, we aim to provide an oneM2M structure implementation that demonstrates the interconnection of various IoT applications based on three protocols in the application layer (HTTP, CoAP, MQTT) and diversified wireless technologies (Wi-Fi, Bluetooth, Zigbee). Therefore, data can be transmitted without regarding to the physical layer or the difference in their upper transmission protocols. The composition of this paper is as follows. Section 2 introduces details of the oneM2M standard and these transmission protocols, technologies will be implemented. We describe an architecture to interpret the operation of the systems in section 3 and discuss about the capacity to expanding follow edge computing orientation in section 4. The conclusion is presented in section 5.

2. Background

2.1 oneM2M Standard

The oneM2M is designed for a horizontal M2M service platform with a group of common service functions to allow the interoperability of heterogeneous M2M system. Its aim is to provide both basic functionalities for itself and various advanced functionalities for interworking with other systems which are independent of underlying networks and topologies [1]. To do that, oneM2M has resource-based architecture. Every functional entity is represented by a node along with its resource, which is a tree structure with a set of attributes, compliant with oneM2M standard. Moreover, the oneM2M resource is supported Representational State Transfer (REST) architecture with five control methods: Create, Read, Update, Delete and Notify (CRUDN). These methods are accessible through Application Programming Interface (API) by a pair of messages called primitives. Request and response message can be represented by JavaScript Object Notation (JSON) or Extensible Markup Language (XML) format. The oneM2M defines the primitives as internal messages to communicate among inside modules of its core. Essential functional entities are represented in resources. Each entity is identified with a unique ID and a corresponding URI for its resource.

The architecture of oneM2M is present in Fig.1.

- Application Entity (AE) is an application layer entity implementing a M2M application service logic to provide an application, e.g. a power metering application. Common Services Entity

* Corresponding author: Tel.: (+84) 912636939
Email: vinhtq@hust.edu.vn

(CSE) comprises a group of "common service functions" (CFS) in the M2M environments. That involves data, security, device management, etc. The CFSs can be used by AEs or other CSEs.

- Network Services Entity (NSE) allows CSEs to make use of services in underlying network, such as device management, location services and device triggering. The oneM2M defines several kinds of node with different functions, composing of different entities.
- Infrastructure Node (IN) is the indispensable component in oneM2M systems. It must contain at least one CSE and has zero or more AEs. There is exactly one IN per service provider.
- Middle Node (MN) is considered as a gateway, which must comprise one CSE and can have zero or multiple AEs. It can connect to IN and/or other MNs.
- Application Dedicated Node (ADN) must have AE but no CSE. It represents a constrained M2M device and can connect to MNs or IN.
- Non-oneM2M Node (NoDN) is a Node that does not contain oneM2M Entities (neither AEs nor CSEs). Such Nodes represent devices attached to the oneM2M system for interworking purposes.

When communicating between modules inside one node, oneM2M using its protocol and primitive message. However, to connect a oneM2M node to other systems, it must define a method, standardized bindings, to mapping primitive messages and other application protocol. The oneM2M currently support three types of protocol binding: HTTP, CoAP and MQTT. While several protocol bindings are standardized and perform on application layer, Interworking Proxy Entities (IPE) allows to make connections through different technologies at a lower level as well as proprietary protocol.

2.2 MQTT and CoAP protocol

MQTT, a lightweight publish/subscribe messaging connectivity protocol, stands for MQ Telemetry Transport. It is designed for M2M telemetry in low bandwidth environment and constrained devices [2]. There are two types of MQTT devices: clients and brokers. A client can publish a message on a topic or subscribe to a certain topic to receive messages. The broker is primarily responsible for receiving all messages, filtering the messages, making decision of which is interested in them, and then publishing the selected messages to the subscribed clients.

The Constrained Application Protocol (CoAP) [3] is a specialized web transfer protocol compatible with constrained devices and constrained networks in the IoT environment. It is based on the REST architecture and supported by IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs). Although the interaction model of CoAP is similar with HTTP (using pair of messages (request/response) and response code), the CoAP has smaller message and more effective encoding method to save memory space and bandwidth.

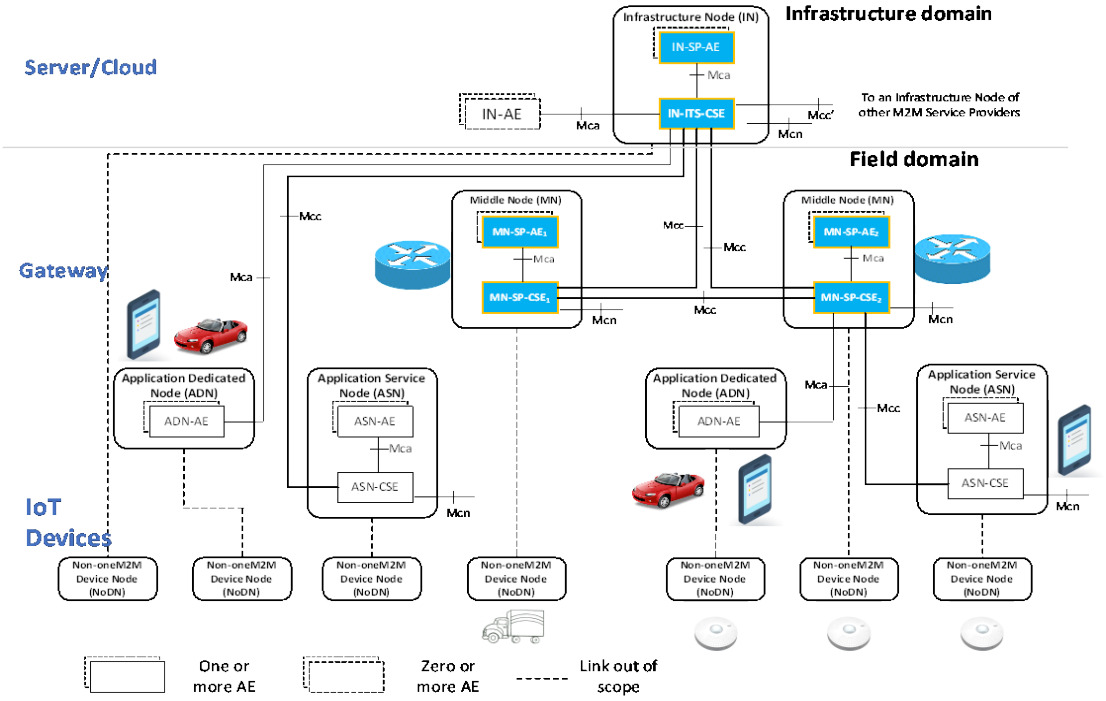

Fig. 1. The architecture of oneM2M for an application of smart parking in Intelligence Transportation System.

2.3 Bluetooth and Zigbee technology

ZigBee [4] is the commercial name of the IEEE standard 802.15.4 for low-rate wireless personal area network standard (WPANs). It operates in the ISM radio band, use the 868 MHz band in much of Europe, 915 MHz in the USA and 2.4 GHz in many other locations. The speed transmission depends on the used frequency band, but the maximum is 250 Kbps. Although, it is slower than other popular wireless technologies such as Wi-Fi to tradeoff for the lower power consumption and low-cost devices. ZigBee is commonly used for wireless control and monitoring applications in wireless sensor networks (WSNs).

Bluetooth [5] operates at 2.4GHz, the same unlicensed ISM frequency band where RF protocols like ZigBee and Wi-Fi also exist. Bluetooth networks have two types of functional devices: master and slave. A single master device can be connected up to seven different slave devices, while a slave device is connected to only one single master. Hence, the master can send data to any of its slave nodes and request data from them as well. Slave nodes are only allowed to transmit to and receive from their master. The two technologies are both using in various IoT/M2M system.

2.4 Related work

To interconnect among different frameworks and devices, oneM2M use two ways: *protocol binding* and *inter-proxy entity (IPEs)*. In the former, different application protocol data model is mapped to oneM2M primitives. In the latter, an additional module is designed and implemented to translate the communication with other frameworks using their own protocols. The two ways are both based on the core oneM2M primitives.

Protocol binding. If an existing IoT/M2M system runs on a certain application protocol, oneM2M MN must be installed a mediated module called protocol binding to help primitive message mapping to such application protocol message. oneM2M presently support up to three protocol binding: HTTP, CoAP and MQTT. While HTTP is used for stable connections such as a connection between MN and IN, CoAP and MQTT is suitable for connections to resource-constrained devices like sensors/actuators. The difficulty is the deployment of the mentioned IP-based protocol stacks on different network communication technologies, which leads to heavy load for resource-constrained devices. Besides, the message must follow a conversation structure of oneM2M standard to make changes in data messages of existing network structure. There are several studies that have been successful with this solution, LoRa based motes (as IoT devices) and gateway as MN, that enable LoRa-based devices to exchange data through MQTT and CoAP protocol [6]. However, it is costly to integrate IP-based application protocols in all technologies/ systems using Bluetooth, Z-wave. Hence, the following solution seems more appropriate.

Inter-Proxy Entity (IPEs). The other way is to develop a *plugin entity running in MN*. It communicates to other IoT/M2M system and converts its data structure into of conversation following oneM2M standard. IPE enables to preserve other proprietary system and not to change the content of existing proprietary messages. The drawback of IPE is that MN needs powerful hardware and plugged with a transceiver hardware module (e.g. Wi-Fi, Bluetooth, Zigbee radio).

A theoretical design and several hints of implementation of a system based on IPEs are shown in [7], but no detail testbed is described. To enhance the efficiency of IPE, [8] proposed the extension of protocol binding approach CFS included. Not only does IPE support connection to/from IoT devices or oneM2M networks, but it also interworks with others service platforms, e.g. building IPEs to bridge oneM2M-based system and IoTivity/AllJoyn-based system is presented in [9].

3. Hardware Architecture and Firmware Implementation for oneM2M-based interconnection

Our work resolves two main goals. The first is to combine various application layer protocols through standardized protocol binding. Secondly, we design and implement IPEs to integrate the Bluetooth devices into the system. To carry out oneM2M services, we use Eclipse OM2M project, which is an open source implementation of oneM2M standard, initiated by LAAS-CNRS [10].

3.1 Hardware architecture

In the infrastructure, IN node (server) is installed in a powerful computer. It enables Internet connectivity providing an available link with field domain and possibly end-users. In the field domain, we have several types of hardware:

i) MN/Gateway works as a multiple-tech gateway, which is currently based on a laptop. We make use of Network Interface Card (NIC) built-in to connect with IoT devices through Wi-Fi and Bluetooth. To offer connectivity with the Zigbee-based devices through CoAP on IPv6, the computer also assembles Z1 Zolertia node as border-router.

ii) IoT devices We deploy several types of IoT devices based on various technologies: Wi-Fi-based

device based on ESP8266 Node MCU acts as a sensor node; Zigbee-based device is Z1 Zolertia node considered as actuator node; Bluetooth-based devices can be hand-held devices like smartphones, tablets.

3.2 Firmware on Gateway

Firmware on multi-tech gateway (MN node) is a crucial component in the system. It has two main functions: to register with IN-CSE to manage devices and share CFSs (MN-CSE); all procedures are automatically configured in OM2M-IN and OM2M-MN and to create connections with IoT devices which belong to various networks and implemented different protocol. The gateway firmware needs to be customized and installed additional plugin. The detailed components we have implemented is described below.

3.2.1. Wi-Fi connection: Wi-Fi-based devices connect to gateway via MQTT protocol. Hence, a Mosquito broker and MQTT protocol binding plugin must be installed on the gateway. MQTT protocol binding is responsible for two-way message transportation using specific publish/subscribe topic defined in [11]. Mosquito broker ensures the operation of MQTT standard such as send, store and forward.

3.2.2. ZigBee connection: To communicate across ZigBee, we use a border-route for RPL-based network of ZigBee-based devices. Since this network is implemented with CoAP as application protocol, our gateway needs CoAP protocol binding plugin installation.

3.2.3. Bluetooth connection: We developed Bluetooth IPE using Bluecove library to connect the gateway to Bluetooth-based devices. The IPE includes two components, Bluetooth OBEX server and oneM2M AE. The former manages the pairing with Bluetooth-based devices and exchange of data through Bluetooth interface. The latter is responsible for mapping between data to/from Bluetooth-based devices and oneM2M primitives, creating representative data identification of the IoT devices in oneM2M MN/IN database and operational procedure interworking.

3.3 Use case description

The setup of the testbed is a case study for a typical monitoring and management IoT application, see Fig.2. The IoT devices consist of sensor/actuator nodes and MN-gateway to gather data in the field. The gateway also supports the direct access of system manager/admin for operation and maintenance. The center of data management locates in OM2M server/IN node and support the application access of users through Internet. The testbed deploys Wi-Fi, Zigbee, Bluetooth for access technology and MQTT, CoAP, HTTP at the application. The interconnection of heterogenous system is visualized.

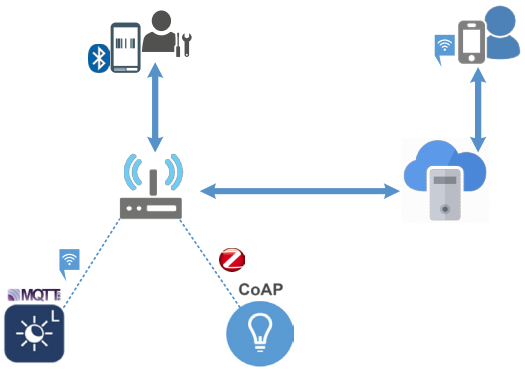

Fig. 2. The integration of IoT/M2M systems based on the common service platform oneM2M.

In the initial phase, MN-CSE automatically register with IN-CSE to make a basic OM2M system. After initiating/loading the CoAP protocol binding, MQTT protocol binding and Bluetooth IPE module, MN is ready to serve connections from IoT devices.

In the second phase, when IoT devices consisting of Wi-Fi-based device and Zigbee-based device are turned on, they will establish their resource trees (the information of their particular AEs) and their essential containers to store their data in MN. The establishment/resource registration with MN is processed through the primitives of OM2M. Afterward, the sensor node and actuator nodes start to send their data encapsulated in Content Instance (CINs) format to the gateway.

To process the collected data, we use Manager ADN loaded in MN gateway. It creates a *subscription* of certain resource to get notifications about the interested events. After analyzing, MN gateway can detect abnormal events and update its database or send control commands to the actuator ADN.

In our scenario, sensor node sends luminosity data using MQTT protocol on Wi-Fi to the MN-gateway every minute. The gateway receives data and sends the notification that contains the sensor values to a subscriber, the ADN named *manager,* which is a data processing module checking luminosity data to be over a specified threshold. If the value is lower, a control command *"turn LED ON"* will be sent to the actuator ADN and immediately forwarded to underlying actuator device using CoAP protocol on Zigbee. The actuator receives notification from MN-gateway and turn on LED. Hence, the data processing function can reside in the MN-gateway to reduce the data sending to IN node/OM2M server.

The Bluetooth IPE in MN-gateway is setup as a Bluetooth server. After the Bluetooth-based device paired with the gateway, all resources are automatically created, then temperature data and led status can be monitored on user's smartphone or system admin's smartphone, see Fig.2.

4. Discussion

Numerous IoT devices connected to oneM2M system generate large volume of data, which might cause server overloaded and increase network latency. To solve this problem, *edge computing* has been proposed to reduce the access bandwidth to core network/cloud and release workload of the cloud servers. Currently, oneM2M has already supported some simple functions to deploy edge computing environment: i) Database can be stored in MN; ii) Two MNs can directly exchange data with each other, without going through IN. In our proposed architecture, the processing component can reside in MN-gateway instead of locating in IN node. It is possible to deploy the data processing function for all IoT devices connected with up to three MN-gateways. The current limitation is the communication among MNs, which is presently point-to-point. The awareness of link existence is only made between two neighboring MNs. To resolve this problem, a routing protocol need to be added in oneM2M.

At the current stage of the work, we just use a laptop as a gateway platform for implementing the connect with IoT devices through Wi-Fi and Bluetooth. To offer connectivity with the Zigbee-based devices through CoAP on IPv6, the computer also assembles Z1 Zolertia node as border-router. In the future work, we focus on design a multi-platform IoT gateway embedding AI/Edge Computing and considering the problem of speed adaptation, devices' self-configuration, limited battery powered and secure problems. The proposal of multi-platform IoT gateway aims to be applied for a smart on-street parking management system.

5. Conclusion

We demonstrate an implementation of oneM2M system which operates on Wi-Fi, Zigbee and Bluetooth technology and uses three application protocol HTTP, CoAP, MQTT. By developing a plugin in the MN-gateway, we do not need to change the existing systems. We also deploy a simple data processing function at the MN-gateway to limit the amount of data sending to IN node. The interconnection of different systems allows data exchange efficiently and regular applications can be applied. In the future work, we plan to design stand-alone gateways to replace laptop and to design and deploy more function of edge computing one numerous gateways for performance evaluation.

Acknowledgments

This work is supported by the project T2018-PC-068 from Hanoi University of Science and Technology

References

[1] J. Swetina, G. Lu, P. Jacobs, F. Ennesser, and J. Song, Toward a standardized common M2M service layer platform: Introduction to oneM2M, *IEEE Wireless Communications,* vol. 21, no. 3, pp. 20-26, jun 2014.

[2] OASIS, MQTT Version 3.1.1, p. 81, 2014. [Online]. Available: http://docs.oasis open.org/mqtt.html

[3] Z. Shelby, K. Hartke, and C. Bormann, RFC 7252: The Constrained Application Protocol (CoAP), 2014. [Online]. Available: https://tools.ietf.org/html/rfc7252

[4] ZigBee Alliance, Inc., ZigBee Specification, 2012. [Online]. Available: http://www.zigbee.org/wp-content/uploads/2014/11/docs-05-3474-20-0csg-zigbee-specification.pdf

[5] BLUETOOTH SPECIFICATION Version 3.0, [Online]. Avaiable: https://www.bluetooth.com/specifications/bluetooth-core-specification/

[6] Thanh-Long Nguyen, Simone Patonico, Maite Bezunartea Steffen Thielemans, An Braeken and Kris Steenhaut, Horizontal Integration of CoAP and MQTT on Internet Protocol - based LoRaMotes, in 2018 IEEE 29th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Sept. 2018.

[7] Žitnik, S., Janković, M., Petrovčič, K., Bajec, M.: Architecture of standard-based, interoperable and extensible IoT platform, in Proceedings of the 24th Telecommunications Forum (TELFOR), Belgrade, pp. 1–4 (2016).

[8] J. H. Huh, D. H. Kim, J. Deokkim, oneM2M: Extension of protocol binding: Reuse of binding protocol's legacy services, Proceedings of International Conference on Information Networking (ICDIN), pp. 363-365, 13–15 Jan. 2016.

[9] C. W. Wu, F. J. Lin, C. H. Wang, N. Chang, OneM2M-based IoT protocol integration, 2017 IEEE Conference on Standards for Communications and Networking (CSCN), pp. 252-257, 2017.

[10] Eclipse OM2M-Open Source platform M2M communication. [Online].

Available: http://www.eclipse.org/om2m/

[11] oneM2M Standards for M2M and the Internet of Things - Published Specifications. [Online]. Availa-ble: http://www.onem2m.org/technical/published-drafts.