A Feasibility Study on Developing IoT/M2M Applications over ETSI M2M Architecture

Fuchun Joseph Lin, Yi Ren and Eduardo Cerritos
College of Computer Science
National Chiao Tung University
1001 Ta Hsueh Rd., Hsinchu, Taiwan 300, R.O.C.
fjlin@nctu.edu.tw, renyi@cs.nctu.edu.tw, eduardo.cerritos@gmail.com

***Abstract*— A common service platform is considered as the key enabler to catalyze the development of IoT/M2M applications for a large variety of vertical markets such as smart energy, smart transportation, home and industry automation, eHealth and connected vehicles. This paper reports an initial effort at National Chiao Tung University in Taiwan with such an experimental approach. We first train our graduate students with an ETSI M2M architecture-compliant service platform OpenMTC from FOKUS, then charter them with the tasks of developing diverse M2M applications. The effort is used as a feasibility study to investigate how useful the notion of a common service platform for IoT/M2M is and how urgent an international standard is required in defining such a platform. We intend to use the result of the study to create a suitable IoT/M2M curriculum for our students.**

Keywords—IoT/M2M Applications; Horizontal Service Platforms; ETSI Standards; OpenMTC

I. INTRODUCTION

IoT/M2M is a reality. Industry research estimates by 2020 the number of connected devices could reach more than 50 billion. This will undoubtedly impact every aspect of the society [1]. However, if we take a look at the current landscape of IoT/M2M, available applications belong only to particular vertical markets.

IoT/M2M, in its current stage, is not flexible enough to accommodate 50 billion of devices, along with new and diverse services and applications. Therefore IoT/M2M should get into a transition phase towards maturity with the development of a common service platform [2]. Standard developing organizations have realized this and commenced efforts to produce a definition for such a platform. Nevertheless, as any other disruptive technology, its effectiveness will depend on the adoption from industry players, service providers, developers and finally the end-users.

In this study we train a group of NCTU students, as potential researchers or developers of creative applications, with an implementation of a European Telecommunication Standard Institute's (ETSI) M2M standard compliant common service platform. As a result they developed four IoT/M2M applications using such a platform. It is worthy to mention that students are from computer science and related backgrounds with an intermediate knowledge of programming.

In the following sections we briefly introduce the ETSI M2M standards. Then in Section 3 we present our architecture. Section 4 explains the applications produced on this study: Energy Saving, Bus Tracking, Smart Lighting and Facility View. Section 5 provides our evaluation and finally, in Section 6 we present our concluding remarks and future work.

II. BACKGROUND

International standard developing organizations, such as the European Telecommunication Standard Institute (ETSI) have set their effort in the definition of a standard for a high level service layer platform for M2M. For the elaboration of such a standard, ETSI collected requirements based on use cases from several important vertical markets of IoT/M2M, such as Connected Vehicle, Smart Metering and e-Health.

ETSI on its high level architecture (Fig. 1), identifies three domains: Application, Network, and Device. The Network Domain comprises a collection of service capabilities in addition to core and access networks. Service Capabilities expose M2M functionalities through a set of open interfaces [3]. The Device Domain contains sensors, actuators and other devices typically found in a wireless sensor network. ETSI also defines an M2M Gateway which also holds service capabilities and works as a proxy to the Network Domain for capability-limited devices that are not smart enough to directly connect to the M2M Core. The Application Domain includes M2M Applications and Client Applications.

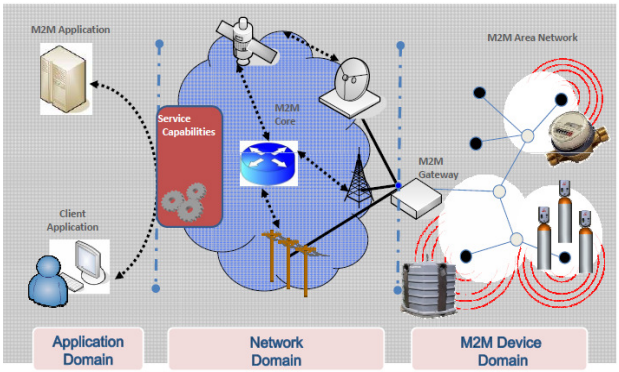


Fig. 1. ETSI High Level Architecture (Source: ETSI)

ETSI defines a resource tree as their M2M platform data model. The hierarchical structure reflects every element present in the platform as a resource. Applications, sensor data,

subscriptions to such data, etc. have a representation as resources. The resource tree resides, along with the service capabilities, within the Device Domain inside M2M Gateway and also in the Network Domain inside M2M servers.

ETSI's services capabilities communicate with each other and other elements inside the three domains of the M2M network through resources. Therefore ETSI specifies a RESTful resource oriented approach as their communication mechanism, where operations are mapped to a protocol specific verb e.g. create is mapped to HTTP Post. However, HTTP is not the only transport protocol. For capability-limited devices ETSI also defines mappings to CoAP.

III. ARCHITECTURE

In our architecture we group components into three layers: core, interoperability layer, and application layer. This approach is designed in such a way it considers students' diverse set of programing skills, level of exposition to ETSI's standard. For example, students with more knowledge about the standard are assigned to work closer to the platform. Fig. 2 illustrates our architecture.

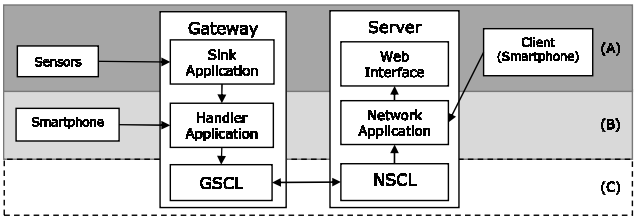


Fig. 2. Architecture, (A) Application layer, (B) Interoperability layer, and (C) Core (OpenMTC)

A. Core Layer

Our architecture core component is the OpenMTC platform, a "standard compliant middleware platform for M2M" [5]. OpenMTC, at its current release, implements the interfaces required for the communication among ETSI's architecture domains. It also includes a basic subset of ETSI defined service capabilities for the correct functioning of an M2M system. These services capabilities are present in the gateway and in the network server through Gateway Service Capability Layer (GSCL) and Network Service Capability Layer (NSCL), respectively. Both NSCL and GSCL contain a Resource Tree.

B. Interoperability Layer

Interactions with OpenMTC are based on a well-defined set of RESTful operations. However, the developer needs a deep knowledge of ETSI standards to interact with the platform. Therefore we define a second layer of abstraction, Interoperability Layer that comprises mechanisms of interoperability for each of our applications, to further simplify application development.

1) Handler Application

In order to communicate with the GSCL, application developers should register a Gateway Application (GA) into its resource tree. Once a GA is registered, it needs to declare some container, a customizable element in ETSI M2M Resource Tree. Each GA has its own container collection and the sensor-collected data is stored in the content instances of those containers. Different from the traditional data modeling using entity-relationship approach, the M2M data modeling is done with the containers in the resource tree. When a GA declares a container, it also specifies its internal structure (similar to a JavaScript Object) and attributes such as the maximum number of content instances the container can hold. For example, Fig. 3 shows a smart home's GA for thermometers that defines two containers: one with only one content instance to store the latest reading (its content was overwritten on every data pushed from the device), and a second one without maximum number of content instances to store historical data. After containers are created, GA then can send data to GSCL.

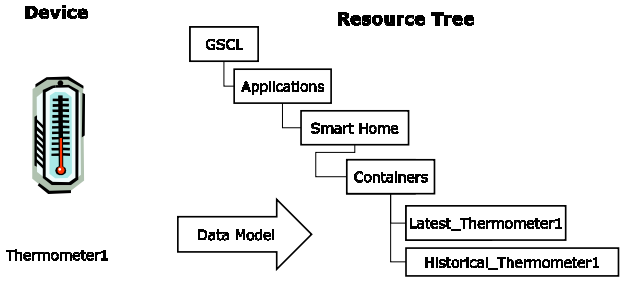


Fig. 3. Data model: Devices Mapped to One or Multiple Containers in the Resource Tree.

To relieve application developers from the details of ETSI M2M standards regarding data insertion operations in the resource tree, the GA functions are separated into a handler application and a sink application. The former translates incoming sensor data into ETSI compliant format while the latter interacts with sensors to collect their data. A simple HTTP post can be used to pass the data from the sink application to the handler application. In this way developers can focus on their respective scope to reduce the development effort.

2) Network Application

Unpredictability is the characteristic of M2M traffic, and more specifically of wireless sensor networks within M2M Device Domain. This implies that Network Application (NA), where the application logic in an ETSI M2M system typically resides, may not know when to retrieve sensor data. To take care of such a situation ETSI defines a subscription/notification mechanism on which NAs can subscribe to the changes on relevant resources of its NSCL. NSCL will then notify the corresponding NA whenever those changes happen.

Such a subscription/notification paradigm also affects how we should design Client Applications. In our solution, whenever an NA is notified with data, it delivers a customized data resource in the form of a temporary file. Client Applications can consume their` customized data resources whenever they need disregard of the mechanisms employed by NA to acquire the data. Fig.4 illustrates this data retrieval mechanism from NSCL to Client Application.

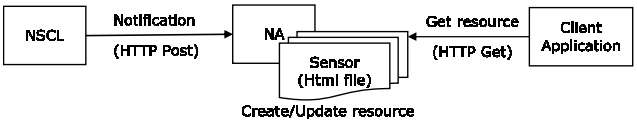


Fig. 4. Data Retrieval Mechanism

C. Application Layer

On this layer, we use sensors on Jennic boards and Android smartphones to capture data. Jennic boards communicate using ZigBee protocol with a Sink Application. Then the Sink Application transmits sensor data via HTTP post to the Handler Application. On the client side, applications have web interface and/or Android application. More details about applications are discussed in the next section.

IV. APPLICATIONS

In this section, we present four M2M applications developed on top of the aforementioned ETSI M2M architecture (as implemented in OpenMTC) by NCTU students including Bus Tracking System, Building Energy Saving System, Smart Lighting System, and Facility View System, respectively.

Fig. 5 shows a simplified view of how these four M2M applications are over a common OpenMTC platform. Each M2M application has different sensor setting in its area networks. Via an M2M wireless base station (what wireless protocol employed depends on the nature of the application), the application-dependent sensors connect to the M2M gateway equipped with GSCL provided by OpenMTC. The GSCL then communicates with the NSCL on M2M Application Servers via Internet. The users via their Android handsets can access these M2M applications. A quick overview of each M2M application is presented below.

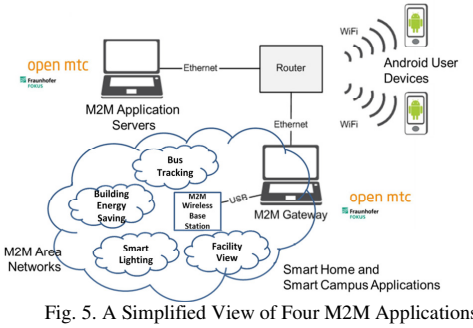


Fig. 5. A Simplified View of Four M2M Applications over a Common Platform

A. Bus Tracking

The Bus Tracking System allows students to track the location of a bus on Google maps in real time. This helps students to catch a bus on time for their classes.

As shown in Fig. 6, the buses under tracking are equipped with Android phones with GPS. The Android phones (GPS sensors) obtain the current locations of the buses and send them to GSCL via 3G mobile networks in (or near) real time.

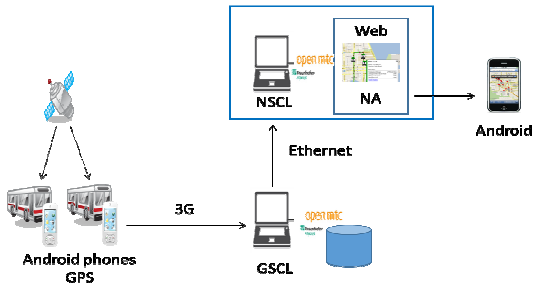
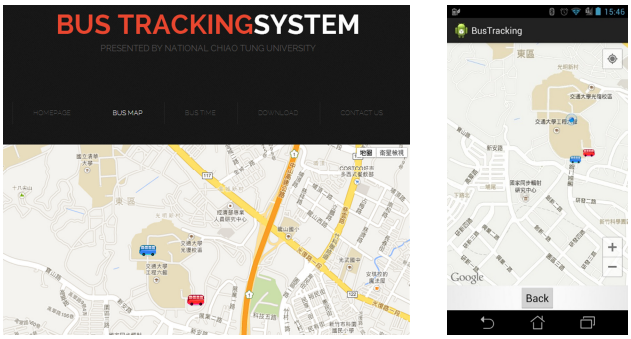


Fig 6. A Simplified Bus Tracking System Architecture

In the client application, the users can see the bus location and the arrival time of the next bus by using either a web browser or an Android App. More specifically, Fig. 7(a) shows a snapshot of our bus tracking web GUI. One can see that two bus icons (one red and one blue) are shown in the Google maps, and their locations are updated every second. In addition, Fig. 7(b) shows a snapshot of the bus tracking Android App GUI, which has the same function as the web-based GUI. This application will enable NCTU students rely no longer on fixed bus timetables but a more accurate tracking system based on M2M technologies.



(a) (b)

Fig. 7(a) Snapshot of Bus tracking Web GUI and (b) Snapshot of Bus Tracking App GUI

B. Building Energy Saving

The Building Energy Saving System is designed to reduce energy consumption in our campus buildings. Currently, one of the primary ways to save energy consumption in buildings is to monitor and analyze energy usage in buildings via various sensors. Based on the results of analysis, the system is thus able to perform predefined actions such as to turn off or turn low light, monitor, air conditioning, etc. to reduce energy consumption.

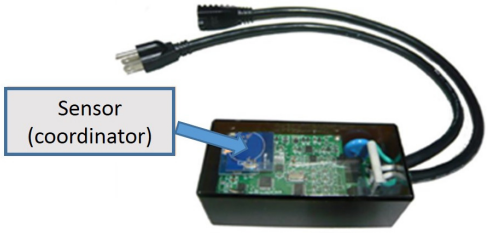


Fig. 8. An Example of Smart Meter

Each electronic appliance in this application is equipped with a smart meter [6], as shown in Fig. 8. This smart meter has a Jennic-based sensor platform, which can communicate with an M2M gateway (see Fig. 5) and send data to M2M application servers. Via the OpenMTC platform, mobile users can easily check and control the status (on/off) of an appliance as shown in the leftmost and the middle pictures of Fig. 9, respectively. In addition, the appliances can be programmed to automatically change their status by a predefined schedule set by the users, as shown in the rightmost picture of Fig. 9.

C. *Smart Lighting*

Besides the Building Energy Saving System, NCTU students also developed a Smart Lighting System. There are two basic functions in this demo system: i) Android based mobile phone App to switch the LED lights on/off and to control the LED light intensity; and ii) Automatic motion detection to adjust LED light intensity. For the latter, motion sensors [7] are deployed in our system to detect if there are people nearby or moving. As illustrated in Fig. 10, when a user is near a LED light, the sensor will increase the LED light intensity. Otherwise it keeps brightness low for saving energy. If the user continues stay under the LED light, the sensor will adjust the LED to the brightest.

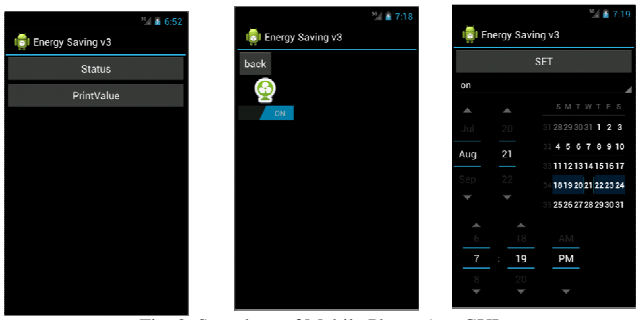


Fig. 9. Snapshots of Mobile Phone App GUI

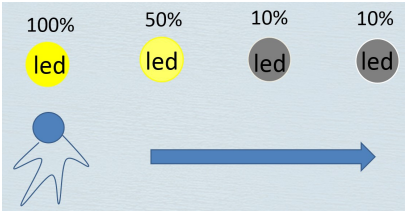


Fig. 10. Smart Lighting – LED's Light Intensity adjusted according to the User's Location and Movement

D. *Facility View*

The basic idea of Facility View System is to monitor the status of facility in real time. The application is also built on top of OpenMTC and its first application is for real-time monitoring of the washers and dryers in the laundry rooms of NCTU campus. The application allows students to know the status of each washer and dryer in the basement of their dorms. The students can thus better plan their time to do this errand.

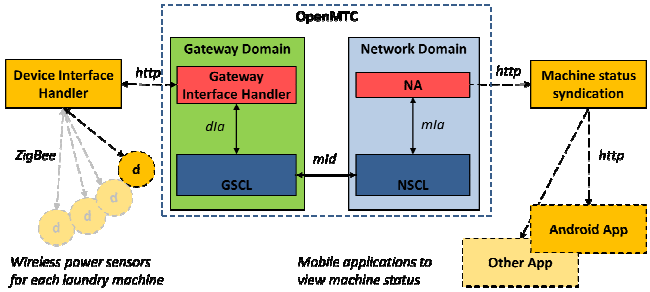


Fig. 11. A Simplified Architecture of the Facility View System.

As shown in Fig. 11, the remote sensing of machine status is accomplished through a Zigbee-based Jennic sensor device (indicted by "d"). A Device Interface Handler converts Zigbee messages into HTML messages to communicate with the Gateway Interface Handler on top of the GSCL. Via the GSCL and the NSCL, the NAs can easily collect data from the GAs and are able to discover new devices as they come online. This allows for easy system growth. Through the NA, the system is able to centralize information and present it to client applications. An Android App can thus connect to the NA-created information and obtain immediate machine status for all connected washers and dryers.

V. EVALUATIION

To develop the above four smart campus/smart home applications, we have recruited 14 students to participate in the project. We divide the students to 4 different teams with each team charted with the development of one particular application. As a result, each group consists of 3-4 students. Majority of our students are graduate students with an intermediate level of programming skill and major in either CS or EE. Nevertheless, they have not been exposed to much technical background in sensor or IoT/M2M networks.

In our study, we intend to verify whether a common service platform really helps with rapid development of IoT/M2M applications. Thus such a choice of students without strong background in sensor/IoT/M2M seems ideal as the effectiveness of a common service platform will be more obvious. In the beginning of the project, all participating students are divided into two groups and trained differently. One group of students is trained with deeper understanding of the common platform while the other group of students is trained with more sensor network skillset. In each application development team, however, we ensure that the team members consist of both types of students.

With a common service platform, the development effort for the IoT/M2M application shifts to four major software modules as illustrated in Fig. 11:

1. Device Interface Handler
2. Gateway Interface Handler
3. Network Application (NA)
4. Machine Status Syndication

It turns out that the students trained with platform expertise are best fit for the tasks of developing Gateway Interface Handler and NA while those trained with senor network

expertise are best suited for the tasks of Device Interface Handler and Machine Status Syndication. Of course, students with these two different types of expertise are both required to make the application work.

By relying on a common service platform such as OpenMTC, student developers are relieved from the tedious task of developing complex underlying network infrastructure such as GSCL and NSCL. They can focus on the application level effort involving mostly about the service features of the application itself.

Through this experimental exercise, it becomes obvious that different IoT/M2M applications will have the following unique components:

1. Different ways of collecting data from M2M device /sensor network as reflected in the Device Interface Handler module.
2. Different ways of presenting data to the client application as reflected in the Machine Status Syndication module.
3. Different Gateway Interface Handlers in order to insert specific data collected into the GSCL resource tree.
4. Different Network Applications (NAs) customized according to how the data are going to be used by client applications.

Moreover, data models used for different applications will also be different. As a result, an M2M common service platform needs to have the capacity to accommodate a large variety of applications with different data requirements. As the resource tree is commonly used as the data models for IoT/M2M applications. It is important to verify that the resource tree representation is universally applicable to any IoT/M2M applications. For the four smart campus applications NCTU students have developed, this is shown to be true.

For our student developers, data models seem to be the most difficult concept for them to learn and apply to develop applications on top of a common service platform such as OpenMTC. At the programming level, how to store and retrieve data to and from a resource tree requires a steeper learning curve. However, as our experiment shows, if we have 1 or 2 developers familiar with this technique in a development team, an IoT/M2M application can be developed more rapidly with a common service platform than without such a platform.

VI. CONCLUSION & FUTURE WORK

Our experiment runs over a period of 4 months and within 4 months four preliminary prototypes of IoT/M2M applications across four different areas are finished by our students who initially do not have much background knowledge. This demonstrates the advantage of a common service platform for developing IoT/M2M applications. Nevertheless, it is also discovered that there is a steep learning curve for developers to acquire the required platform expertise to build applications on top of it. It works the best if software development can be divided into two parts: M2M area network and M2M service platform.

In the future, we plan to continue expand on these IoT/M2M applications to either add more difficult features or to include other new applications. The goal is to investigate how useful the notion of a common service platform for IoT/M2M is and how urgent an international standard is required in defining such a platform. We believe with the emergence of IoT/M2M, it is time for the university to closely watch the development of this area and create a suitable IoT/M2M curriculum for our students.

ACKNOWLEDGMENT

We would like to acknowledge all students who participated in this experiment of developing four smart campus applications on top of a common service platform OpenMTC. They are Haoru Chen, Hsiang Wen Chen, Kenny Kuo, Juan Jose Celada, Yi-Ta Chuang, Chang-Li Liao, Siau Hong Ng, Chih-Wei Tung, Chen-Yu Wu, Ren-Jie Wu, Tsung Hsiang Wu, and Chun-Yu Yeh. Other faculty members in the College of Computer Science that also contributed to the effort include Prof. Yu-Chee Tseng, Prof. Jyh-Cheng Chen and Prof. Chih-Wei Yi. In addition, thanks to FOKUS in Germany in providing OpenMTC under a licensing agreement with NCTU that allows NCTU students to experiment with various IoT/M2M service ideas over a common platform. The project reported in this paper is sponsored by National Science Council of Taiwan Government under NSC Project Number NSC102-2218-E-009-002.

REFERENCES

Ericsson, "More than 50 Billion Connected Devices," 2011.

D. Boswarsthick, O. Elloum and O. Hersent, M2M Communications a System Approach, 1st ed., West Sussex: John Wiley & Sons Ltd., 2012, pp. 26-32.

European Telecommunication Standard Institute, "Machine-to-Machine communications (M2M); Definitions," 2013.

European Telecommunication Standard Institute, "Machine-to-Machine commincation; mIa, dIa and mId interfaces," 2012.

Fraunhofer FOKUS, "The OpenMTC Vision," [Online]. Available: http://www.open-mtc.org/vision/openmtc_vision/index.html.

L.-W. Yeh, Y.-C. Wang, and Y.-C. Tseng, “iPower: An Energy Conservation System for Intelligent Buildings by Wireless Sensor Networks”, Int'l J. of Sensor Networks, Vol. 5, No. 1, 2009, pp. 1-10.

L.-W. Yeh, C.-Y. Lu, C.-W. Kou, Y.-C. Tseng, and C.-W. Yi, "Autonomous Light Control by Wireless Sensor and Actuator Networks", IEEE Sensors Journal, Vol. 10, No. 6, June 2010, pp. 1029-1041.