IET Networks

Special Issue: Edge Network Computing for Big Data Systems over Internet of Things

IET Journals
The Institution of Engineering and Technology

DormTalk: edge computing for the dormitory applications on campus

ISSN 2047-4954
Received on 6th October 2018
Revised 22nd October 2018
Accepted on 5th November 2018
E-First on 7th March 2019
doi: 10.1049/iet-net.2018.5178
www.ietdl.org

Yi-Bing Lin[1], Min-Zheng Shieh[2] ✉, Yun-Wei Lin[1]

[1]*Department of Computer Science, National Chiao Tung University, 1001 University Road, Hsinchu City, Taiwan*
[2]*Information Technology and Service Center, National Chiao Tung University, 1001 University Road, Hsinchu City, Taiwan*
✉ *E-mail: mzshieh@nctu.edu.tw*

Abstract: Internet of things (IoT) connects a large number of edge IoT devices to produce huge volumes of data. To effectively manipulate the real-time IoT data, it is essential to integrate edge computing with the cloud computing. This study investigates how real-time IoT applications benefit from edge computing. Specifically, the authors show how edge computing and cloud computing can be nicely integrated to build a smart campus environment in National Chiao Tung University (NCTU). Based on an IoT device management platform called IoTtalk, the authors develop an edge IoT computing system called DromTalk for dormitory applications in NCTU. Conventional approaches are difficult to interact with IoT servers of different platforms. The authors' approach proposes a novel method that uses a cyber IoT device to nicely bridge the IoTtalk server in the cloud and the DormTalk servers at various dormitory buildings (the edges) without modifying these servers. The authors investigate the response time performance of the edge DormTalk server, which shows that smartphones can effectively control the dormitory appliances with good user experience.

1 Introduction

Internet of things (IoT) is one of the mainstreams in information and communications technology which connects a large number of edge IoT devices to produce huge volumes of data. Manipulation of the real-time IoT data consumes significant amount of energy in a centralised IoT platform. To resolve this issue, shifting the centralised approach to a distributed structure is essential. Recently, research on edge computing systems has become popular, which offers a paradigm of local computation and data processing of IoT devices to meet the latency/delay requirements, increases the scalability and energy efficiency of IoT applications. This paper investigates how real-time IoT applications benefit from edge computing. Specifically, we show how edge computing and cloud computing can be nicely integrated to build a smart campus environment in National Chiao Tung University (NCTU). Since 2016, NCTU has been deploying IoT-based smart campus applications in several categories [1]:

- Environment sensing includes detection of temperature, CO_2, humidity, PM2.5, wind, rain, ultraviolet, and so on. Fig. 1*a* illustrates a micro weather station with environment sensors deployed in a university farm. The lower part of Fig. 1*b* shows the temperature, the pH value, and the conductivity of an indoor aquarium. Fig. 1*c* is an indoor plant box with the temperature, the humidity, the CO_2, and other sensors.
- Camera monitoring includes indoor monitoring (the upper part in Fig. 1*b*), outdoor monitoring (the left-hand side in Fig. 2*a* for NCTU farm), and security monitoring (Fig. 2*b*) that recognises the objects for various security purposes.
- Appliance sensing and control includes status detection of parking space (Fig. 3*a*), washing machines (Fig. 3*b*), and dryers (Fig. 3*c*). Fig. 2*a* shows the insect light control where the control switches in the right-hand side control the insect light and other farm appliances in the monitored screen at the left-hand side.

The above applications are created based on IoTtalk [2–4], an IoT application management platform that can be installed on top of other IoT protocols such as NB-IoT [5], Arduino [6], and so on. In [1], we tailored IoTtalk for creation of campus applications. In our design, the IoT devices and network applications are modularised and can be conveniently reused through a graphical user interface (GUI). Therefore, the students with or without programming ability can easily create IoT innovation with new applications. We have developed smart applications in various NCTU fields such as the university library, the student dormitory buildings, the office buildings, the research buildings, and so on.

This paper uses the smart student dormitory to describe how we utilise cloud and edge computing for NCTU smart campus. The paper is organised as follows. Section 2 introduces IoTtalk and shows how we can use multiple IoTtalk platforms to fit edge and cloud computing. Section 3 elaborates on DormTalk, an edge computing environment for student dormitory in NCTU. Section 4 investigates the response time performance of DormTalk.

2 Cloud and edge computing with IoTtalk

This section introduces IoTtalk [2–6]. We first show how IoTtalk works with machine learning capability. Then we elaborate on how a smart application can be developed in IoTtalk.

2.1 IoTtalk with machine learning

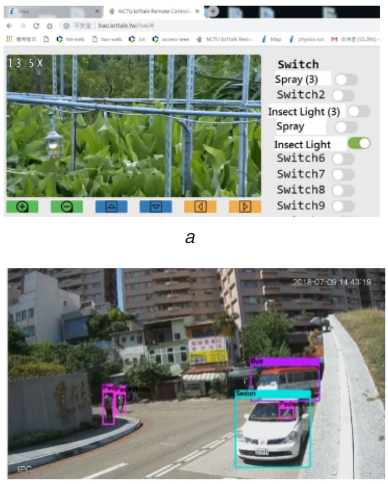
The IoTtalk platform consists of the device and the network domains. In the device domain, every IoT device is installed a software unit called device application [DA; see Fig. 4, (c)–(e)]. The DA is used to communicate with the IoTtalk engine [Fig. 4, (b)] in the network domain. Besides DA, the IoT device needs to implement the IoT device application (IDA) unit. The IDA is a software driver for the IoT device hardware (i.e. sensors, controller, or actuators). For example, the IDA of a parking device is responsible for extracting the signal obtained from the magnetometer, translating it into the binary value (0 for empty and 1 for occupied), and sending the binary value to the DA.

The network domain of IoTtalk consists of two components: the GUI for configuring the IoT applications [GUI; Fig. 4, (a)] and the IoTtalk engine [Fig. 4, (b)]. The IoTtalk engine provides either MQTT or HTTP-based RESTful application programming interfaces (APIs). Through these APIs, a DA sends the data to be manipulated by the IoTtalk engine and/or receives the data from the IoTtalk engine. The GUI is a web-based user interface for one

a b

c

Fig. 1 *Environment sensing examples*
(a) Micro weather station,
(b) Indoor aquarium,
(c) Indoor plant box



a

b

Fig. 2 *Camera monitoring examples*
(a) Outdoor monitoring (NCTU farm),
(b) Front gate security monitoring

to establish the connections and meaningful interactions among the sensors and the actuators through machine learning.

In IoTtalk, a device is defined as a set of input and/or output 'device features' (DFs). An input DF (IDF) is either a sensor (such as a magnetometer) or a controller (such as a button). Therefore, in NCTU's smart campus applications, 'environment sensing', 'appliance sensing', or 'appliance control' are represented by IDFs. An output DF (ODF) is an actuator [such as an air conditioner (AC) or the insect light in Fig. 2*a*] to be controlled by the IoTtalk engine. In our approach, there is a special cyber IoT device called 'ML_device' that serves as the machine learning mechanism for IoTtalk [7]. We use ML_device [Fig. 4, (f)] to describe the device feature philosophy as follows. In ML_device, the DA is responsible for manipulating the IDFs [Fig. 4, (1) and (2)] and ODFs [Fig. 4, (3) and (4)] to communicate with the IoTtalk engine. The IDA of ML_device is an AI unit [Fig. 4, (5)–(8)] implemented with the Python open source tool called scikit-learn [8]. The DA interacts with the AI unit to perform the machine learning functions as follows. Through the IoTtalk engine, the Label-O ODF [Fig. 4, (3)] receives the label from the remote control [Fig. 4, (e)], and the Feature *i* ODFs [Fig. 4, (4)] receive the data from the sensors [Fig. 4, (c)]. The DA sends the sensor values of the features to the AI unit. The feature extraction module [Fig. 4, (5)] extracts the desired characteristics of the sensor data to form a feature vector. To train a machine learning model, both the feature vectors and the user's control signals [i.e. the labels; see Fig. 4, (3)] are sent to the machine learning algorithm module [Fig. 4, (6)] to conduct the model training process, which will generate prediction results by executing the algorithms. The results together with some statistics are sent out to IDFs Result-I [Fig. 4, (1)] and the Statis-I [Fig. 4, (2)] through the DA. Result-I is used to control the actuators [Fig. 4, (d)]. To enhance the performance of the AI unit, the ODF values are also used to conduct validation [Fig. 4, (7)] to tune the hyper-parameters [Fig. 4, (8)] and then fed back to the machine learning algorithm. The values produced by Stats-I are typically illustrated in a display actuator, which can also be used to manually tune the hyper-parameters of the machine learning model. The detailed operation of the AI unit is out of the scope of this paper and is treated in a separate paper [7].

2.2 Developing dormitory applications through the GUI

In NCTU, IoTtalk is tailored to build various smart applications in the dormitory buildings. The platform is called DormTalk. This subsection shows how the configuration in Fig. 4 can be realised for the dormitory garden application through DormTalk.

In IoTtalk, the GUI [Fig. 4, (a)] illustrates the IoT devices and their connections represented by icons and line segments. The GUI allows one to configure the device features, the connections, and the functions corresponding to the IoT devices. For example, the garden application of DormTalk is configured in the GUI as illustrated in Fig. 5. The project name is called 'DormGarden' [Fig. 5, (a)]. The IoT devices used in this project are pre-built as 'device models' and can be accessed from the 'Model' drop-down list [Fig. 5, (b)]. When a device model is selected from the list, the device will be shown in the GUI window graphically. In the GUI, every device is represented by two icons. The input device icon is placed at the left-hand side of the window [e.g. Fig. 5, (c–f)]. Inside the device icon, there are several icons that represent the IDFs of the device. For example, the input device 'Sensors' [Fig. 5, (c)] has IDFs including 'Soil Humidity', 'ATM Pressure' (atmosphere pressure), 'Temperature', 'UV Strength' (ultraviolet strength), and 'Bug Number'. Similarly, the output device icons are placed at the right-hand side of the window [Fig. 5, (d and g)]. In the DormGarden application, Fig. 5 (c and e) implements Fig. 4 (c and e), respectively. Fig. 4 (d) is implemented by Fig. 5 (d). ML_device [Fig. 4, (f)] is implemented by ML_garden [Fig. 5, (f and g)]. To connect an IDF to an ODF, we only need to drag a line between the corresponding IDF icon and the ODF icon (joins 1–10 in Fig. 5), and DormTalk automatically creates the software to handle the interaction between them. In the dormitory garden, the sprinkle, the liquid fertiliser, and the liquid pesticide [Fig. 5, (d)] are controlled by the throttle valves 1, 2, and 3 [Fig. 5, (e)] through joins 8–10, respectively. Originally, these values are manually controlled by the gardener. In DormTalk, we develop the machine learning algorithm ML_garden to automatically control the valves. The automatic process is actually built when the devices are linked together through joins 6 and 7. That is, the garden actuators are automatically controlled through joins 6 and 7 even if the gardener does not take any action on the valves.

3 DormTalk as an edge computing IoT platform

Besides the garden application described in Section 2.2, DormTalk also implements several applications for a dormitory, including smart socket [9] and those for washing machines (Fig. 3*b*), dryers (Fig. 3*c*), ACs, indoor aquarium (Fig. 1*b*), and indoor plant box (Fig. 1*c*). Each of them is considered as a project in DormTalk. This section elaborates on issues for implementing these applications through cloud and edge computing.

3.1 Cloud computing for smart campus

In NCTU, the smart applications for school buses, parking lots, dormitory, and others are managed by an Integrated Operation

a

b

c

Fig. 3 *Appliance sensing examples*
(a) Parking sensor,
(b) Washing machines detector,
(c) Dryer detector

Center (IOC) that allows the university technical staff to conveniently administrate all smart campus applications under one web-based GUI. The NCTU IOC is implemented by a specific IoTtalk server called MapTalk (a web page illustrated in Fig. 6) [10]. MapTalk effectively mosaics various IoT applications in a two-dimensional (2D) map, which can automatically accommodate IoT applications with few simple steps. In MapTalk, an IoT application is represented as a coloured button for a drop-down list [Fig. 6, (a)]. When the button is clicked, the names of all objects (e.g. buses, dormitory buildings, and so on) of that application are shown in the list. We can select one or all objects. Then the objects are shown in the map at their locations. For example, the icon in Fig. 6 (b) represents a camera, and the icons in Fig. 6 (f and g) represent school buses with movement history [Fig. 6, (c)]. Fig. 6 (d) illustrates the numbers of available washing machines and dryers in a dormitory. Fig. 6 (e) gives the PM2.5 level in colour, where the PM2.5 value for a location is represented by a rounded square icon with a label 'P' and its value is represented in colour. All applications can be found in the 'App' list [Fig. 6, (i)]. The routing button [Fig. 6, (j)] provides travel route planning by allowing the student to specify the start and destination locations. This application is the same as that offered in Google Maps.

In the early deployment of NCTU smart campus, all applications were developed together with MapTalk, a centralised

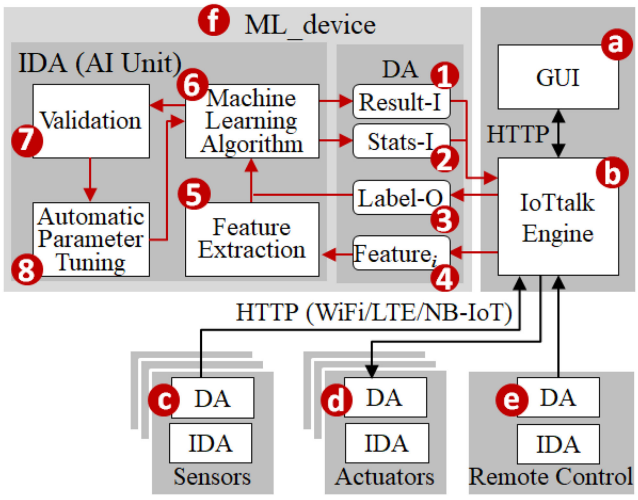


Fig. 4 *IoTtalk platform with machine learning*

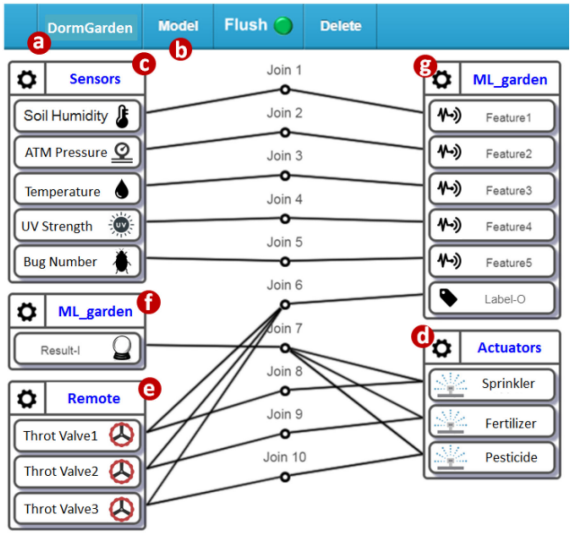


Fig. 5 *Configuring a dormitory garden application in DormTalk*

IoTtalk platform installed in a cloud. Based on MapTalk, we have developed PM2.5 detection (through LoRA and Wi-Fi), parking (NB-IoT), emergency button, dog tracking (through LoRA, LTE, Bluetooth, and Wi-Fi), and so on. We have also created IoT applications at the campus fields including the university library, student dormitory buildings, research buildings, and so on. New applications are being developed in these fields. As the number of applications grows, several issues have appeared in the centralised approach. First, a huge number of small IoT packets are delivered between the IoT devices and the IoTtalk server. This type of traffic is often considered as virus attacks and may be blocked by the firewall. We have developed quick packet aggregation and disaggregation for P4 switches to reduce the number of packets delivered in the network [11]. However, the total amount of information is still the same. Second, it is more difficult to scale with the centralised server, and when it fails, all applications are shut down. Furthermore, when new applications are developed in a centralised IoT platform, they may accidentally interact with the old applications. This phenomenon is called feature interaction that typically occurs in large-scale telecommunications services. To resolve these issues, we introduce edge computing to NCTU smart campus. We use the dormitory as a service field example, where a DormTalk server is installed locally at the dormitory building to perform edge computing. All applications in the dormitory (which are implemented as individual projects in DormTalk) are maintained locally, and the tedious details should not be revealed at the university IOC level. Therefore, it makes sense to partition the campus into several service fields (a dormitory, an office building, university library, and so on), and the IoT applications for each

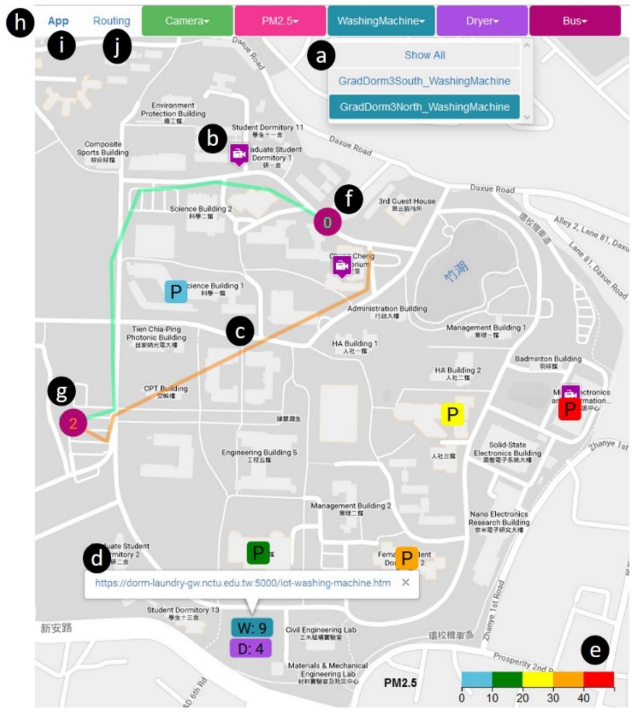


Fig. 6 *NCTU IOC: 2D visual map based on MapTalk*

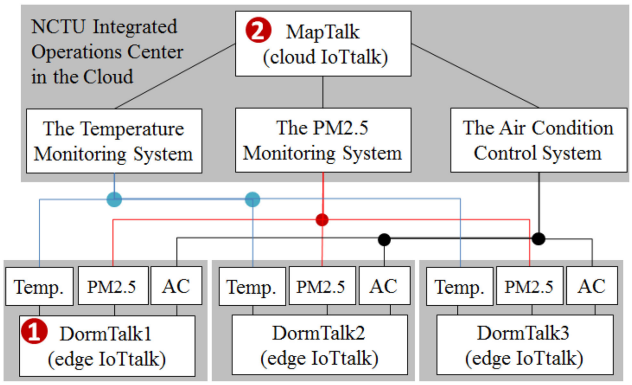


Fig. 7 *Cloud and edge computing in NCTU smart campus*

field are operated by an individual local IoTtalk server tailored for that service field.

Although the servers handle the applications independently at their service fields, certain information should be passed to the IOC for global management and computation. For example, the PM2.5 sensors installed in different service fields are managed in the corresponding local IoTtalk servers. At the same time, all PM2.5 values from different service fields should also be sent to the IOC to be shown in the map (see Fig. 6). To do so, we exercise global and edge computing at the university level and the service field level, respectively. Consider the example illustrated in Fig. 7.

In Fig. 7, the AC at each of the three dormitory buildings is smartly controlled by the local temperature, PM2.5, and humidity sensors through its DormTalk server [Fig. 7, (1)]. Although the control is maintained by edge DormTalks, the sensor values are sent to the monitoring systems at MapTalk [Fig. 7, (2)] for global computation in the cloud.

In NCTU, we developed several independent edge IoT servers that need to communicate with MapTalk. In most IoT approaches, the servers/gateways do not talk to each other transparently. For example, an oneM2M server [12] typically does not interact with an AllJoyn server [13]. Instead, tedious work is required to port AllJoyn devices to connect to the oneM2M server. Such interaction can be easily achieved in IoTtalk as illustrated in Fig. 8. In this example, the edge DormTalk1 installed at Dormitory Building 1 is connected to MapTalk in the cloud through a 'cyber' IoT device called Dorm1. From the viewpoint of DormTalk1, Dorm1 is an

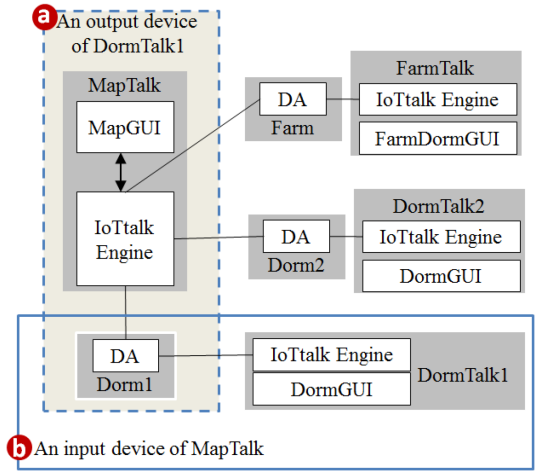


Fig. 8 *Connecting MapTalk with the edge Dormtalk1 through the Dorm1 device*

output device [Fig. 8, (a)], and the IDA of Dorm1 is the whole MapTalk system that will receive the temperature and the PM2.5 values of the building.

On the other hand, since DormTalk1 sends the sensor values to Dorm1's temperature and PM2.5 ODFs, from the viewpoint of MapTalk, Dorm1 is an input device [Fig. 8, (b)], and the IDA of this device is the whole DormTalk1 system that will send the sensor values to the temperature and PM2.5 IDFs. For other edge IoTtalk servers (e.g. FarmTalk and DormTalk2 in Fig. 8), they can interact with MapTalk through the same way as DormTalk1 does.

3.2 Implementation of the cyber device that bridges edge and cloud computing

Now we use the dormitory laundry as an example to show how the structure in Fig. 8 is implemented in the MapTalk and the DormTalk1 GUIs. In this example, the laundry application tells the students whether a washing machine or a dryer is available. Fig. 9 shows the LdryRm project implemented in DormTalk1 (i.e. the edge DormTalk of Dormitory Building 1). In this project, the IoT devices are selected from the 'Model' drop-down list [Fig. 9, (1)] for configuration. LdryRmWM [Fig. 9, (2)] selected form the 'Model' list represents the washing machines in the laundry room, where the WashMach-I IDF gives the status of multiple washing machines (e.g. machine 1 has been used for 30 min) and the LocWM-I IDF gives the location of the washing machines. Similarly, LdryRmD [Fig. 9, (4)] represents the dryers in the laundry room with the dryer status (Dryer-I) and the location (LocDryer-I). Note that the WashMach-I IDF is a sensor that detects the statuses of multiple washing machines. The technique for accommodating multiple physical IoT devices in one IDF was described in [5]. The values of the WashMach-I and the LocWM-I IDFs are sent to the output device WMStatus [Fig. 9, (6)] for execution to provide local intelligence such as sending an alert to inform a student that the washing machine he/she used is stopped. Similarly, DryerStatus [Fig. 9, (9)] processes the values sent from LdryRmD [Fig. 9, (4)]. From the viewpoint of DormTalk1, WMStatus and DryerStatus are the 'output Dorm1' devices [Fig. 8, (a)] where their IDAs are the whole MapTalk server.

We deployed the Map-Project of MapTalk in the cloud (Fig. 10) to collect the laundry information from all laundry rooms on campus. From MapTalk's viewpoint, WMStatus and DryerStatus are the 'output Dorm1' devices whose IDAs include the whole DormTalk1 system [Fig. 8, (b)]. In Fig. 10, WMStatus is connected to the Map device through join 5 [Fig. 10 (10)], which results in the icon shown in Fig. 6 (d). The icon is labelled with 'W:9' to represent that there are nine empty washing machines. Similarly, DryerStatus is connected to the Map device through join 6 [Fig. 10 (11)], which results in an icon with the 'D:4' label as shown in Fig. 6 (d). This icon represents that there are four empty dryers. Another application of school bus, i.e. BusTracking is connected to

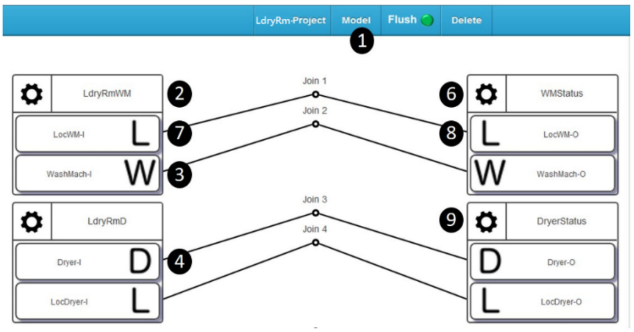


Fig. 9 *LdryRm (Laundry Room) Project in DormTalk1*

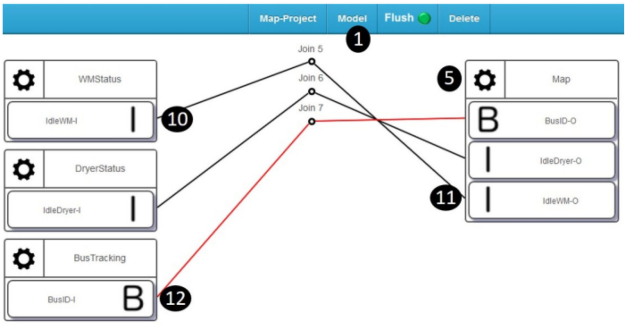


Fig. 10 *Map project for the laundry rooms in Dormitory 1 (at MapTalk)*

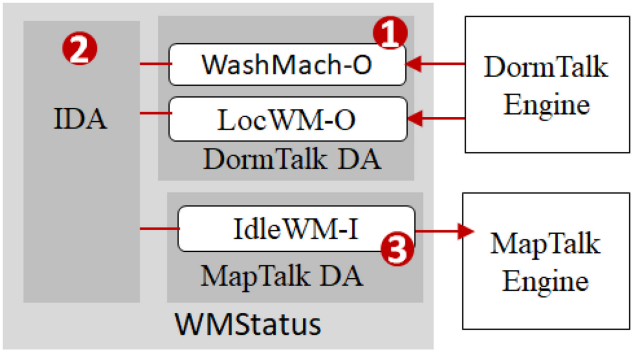


Fig. 11 *WMStatus bridges edge and cloud computing*

Map through join 7 [Fig. 10 (12)] just like WMStatus and DryerStatus, and the bus statuses are shown in Fig. 6 (f and g).

Cyber devices WMStatus and DryerStatus also show that our techniques can bridge edge and cloud computing well. DormTalk engine is installed in a dormitory, and it collects and computes data right there. Obviously, it is an edge computing service. On the other hand, MapTalk engine is a cloud computing service deployed in the data centre. Fig. 11 illustrates how the cyber device WMStatus bridges DormTalk and MapTalk. WMStatus has two DA interfaces connecting to DormTalk and MapTalk, respectively. First, it reads the information of the washing machines from DormTalk [Fig. 11, (1)]. Then, its IDA compute the number of available washing machines [Fig. 11, (2)]. At last, it provides the result to MapTalk [Fig. 11, (3)] via IdleWM-I IDF. Therefore, the computation can be done on the edge, and the communication overheads can be reduced greatly. Similar ideas can be applied to other applications to optimise the efficiency of communication and computation.

4 Performance of DormTalk

In DormTalk, the actuators (ACs, lights, smart sockets etc.) are controlled by the sensors or the students (controllers). We have accommodated smartphones as IoT devices in IoTtalk through a web-based technique called SmpartphoneTalk [1], and the students can use their smartphones to access IoT-based smart campus services without installing any mobile apps. Fig. 12 shows a smartphone for air condition control. When the smartphone is used

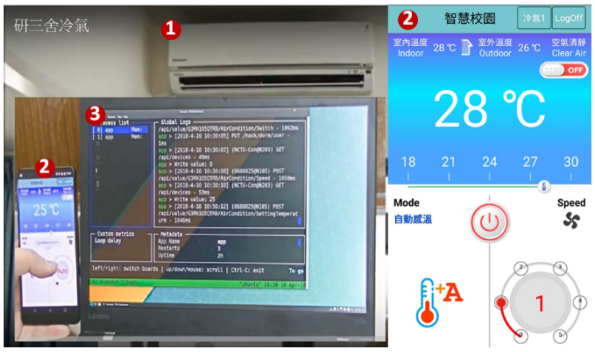


Fig. 12 *Smartphone for air condition control: (1) the AC; (2) the smartphone; (3) the DormTalk engine*

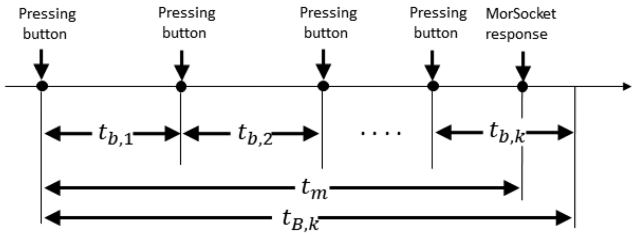


Fig. 13 *Timing diagram for re-pressing the button*

to wirelessly control a dormitory appliance, it is important that the appliance quickly responds to the student's instruction. Let t_m be the delay between when the student presses the button of the smartphone and when the appliance responds. In this section, we investigate two issues about t_m.

First of all, it is important that t_m is short enough so that the student will not continue to press the button. The user experience is poor if the student feels that the dormitory appliance does not respond and therefore has to keep pressing the button. Section 4.1 models the probability that if the user experience is good. Also, when two or more students use the smartphones to compete for a dormitory resource (e.g. the reservation of a study room), the delays of the smartphones may cause unfair results. Section 4.2 proposes an analytic model to investigate if the student competition is fair.

4.1 Performance of DormTalk

Let the *user tolerance time* $t_{b,i}$ be the time interval between when the user presses the bottom for the ith time and the $i+1$st time. Then $t_{b,i}$ is the 'delay limit' that the student can tolerate before he/she attempts to press the button again. Beyond this time limit, the student will consider that the control message sent to the dormitory appliance is lost and unhappily press the button again. This issue was discussed in [9]. We summarise the results here for the reader's benefit. Then we extend the model to investigate the fair reservation issue in Section 4.2.

In Fig. 13, assume that t_m has the Erlang distribution with the shape parameter n and the rate parameter λ, that is, the density function $f_m(t_m)$ can be expressed as

$$f_m(t_m) = \frac{\lambda^n t_m^{n-1} e^{-\lambda t_m}}{(n-1)!} \tag{1}$$

From (1), the Laplace transform $f_m^*(s)$ of $f_m(t_m)$ is expressed as

$$f_m^*(s) = \int_{s=0}^{\infty} f_m(t_m)\, e^{-st_m} dt_m = \frac{\lambda^n}{(s+\lambda)^n} \tag{2}$$

Let $f_{b,i}(t_{b,i})$ be the density function of $t_{b,i}$ with the Laplace transform $f_{b,i}^*(s)$. Let $t_{B,k} = \sum_{i=1}^{k} t_{b,i}$, which has the density function $f_{B,k}(t_{B,k})$. Then the Laplace transform $f_{B,k}^*(s)$ of $t_{B,k}$ is the convolution of that for $t_{b,i}$ $(1 \le i \le k)$. If $t_{b,i}$ are i.i.d. random

variables, then notation $t_{b,i}$ can be simplified as t_b. The density function $f_{b,i}(t_{b,i})$ and the Laplace transform $f^*_{b,i}(s)$ for all i are the same and are denoted as $f_b(t_b)$ and $f^*_{b,i}(s)$, respectively. Therefore, if $t_{b,i}$ is a gamma random variable with the shape parameter α and the rate parameter β, the Laplace transform $f^*_{B,k}(s)$ of $f_{B,k}(t_{B,k})$ is expressed as

$$f^*_{B,k}(s) = \prod_{i=1}^{k} f^*_{b,i}(s) = \frac{\beta^{\alpha k}}{(s+\beta)^{\alpha k}} \tag{3}$$

Let random variable K represent the number of times that the student presses before the appliance responds. Then from (3) and the derivation in [9], we have $\Pr[K > k] = \Pr[t_{B,k} < t_m]$

$$= \sum_{j=0}^{n-1} \binom{\alpha k + j}{j} \left[\frac{\lambda^j \beta^{\alpha k}}{(\lambda+\beta)^{\alpha k + j}} \right] \tag{4}$$

Since $\Pr[K = 1] = 1 - \Pr[K > 1] = 1 - \Pr[t_{B,1} < t_m]$, from (4) we have

$$\Pr[K = 1] = 1 - \sum_{j=0}^{n-1} \binom{\alpha + j}{j} \left[\frac{\lambda^j \beta^{\alpha}}{(\lambda+\beta)^{\alpha + j}} \right] \tag{5}$$

For $n = 1$, (5) is rewritten as

$$\Pr[K = 1 \,|\, n = 1] = 1 - \left(\frac{\beta}{\lambda + \beta} \right)^{\alpha} \tag{6}$$

Similarly, substitute $n = 2$ into (5) to yield

$$\begin{aligned} \Pr[K = 1 \,|\, n = 2] &= 1 - \sum_{j=0}^{1} \binom{\alpha + j}{j} \left[\frac{\lambda^j \beta^{\alpha}}{(\lambda+\beta)^{\alpha + j}} \right] \\ &= \Pr[K = 1 \,|\, n = 1] - \frac{(\alpha+1)\lambda\beta^{\alpha}}{(\lambda+\beta)^{\alpha+1}} \end{aligned} \tag{7}$$

Note that the above results were derived and validated by simulation in [9].

4.2 Modelling fairness

We have developed a dormitory voting system as a cyber IoT device that allows the students to vote with their smartphones through DormTalk. In some dormitory applications, the students need to compete for the resources through this voting mechanism. Therefore, the delay t_m from when a student presses the smartphone and when the voting IoT device receives the message must be short enough so that the 'first-press–first-serve' rule for fairness can be enforced when multiple smartphones are competing the dormitory resources simultaneously. For example, the students may use the study room reservation system to reserve a specific study room. Consider the timing diagram in Fig. 14, where student 1 presses the button of the smartphone at time τ_0, and the message arrives at the reservation server at time τ_3. Student 2 presses the button at time $\tau_1 > \tau_0$, and the message arrives at the reservation server at time τ_2.

If $\tau_2 < \tau_3$, then the result is not fair to student 1. Let $t_{m,1} = \tau_3 - \tau_0$, $t_{m,2} = \tau_2 - \tau_1$, and $t_3 = \tau_1 - \tau_0 > 0$. Then the reservation process is not fair if $t_{m,2} + t_3 < t_{m,1}$. We derive the unfair probability $\Pr[t_{m,2} + t_3 < t_{m,1}]$ as follows. Let t_3 be a random variable with the density function $f_G(t_3)$, then

$$\begin{aligned} &\Pr[t_{m,2} + t_3 < t_{m,1}] \\ &= \int_0^{\infty} \int_0^{\infty} \int_{t_3 + t_{m,2}}^{\infty} f_m(t_{m,1}) f_m(t_{m,2}) f_G(t_3) \mathrm{d}t_{m,1} \mathrm{d}t_{m,2} \mathrm{d}t_3 \end{aligned} \tag{8}$$

Substitute (1) into $f_m(t_{m,1})$ in (8) to yield

$$\begin{aligned} \Pr[t_{m,2} + t_3 < t_{m,1}] &= \int_0^{\infty} f_G(t_3) \int_0^{\infty} f_m(t_{m,2}) \\ &\quad \times \left\{ \int_{t_3 + t_{m,2}}^{\infty} \left[\frac{\lambda^n t_{m,1}^{n-1} \mathrm{e}^{-\lambda t_{m,1}}}{(n-1)!} \right] \mathrm{d}t_{m,1} \right\} \mathrm{d}t_{m,2} \mathrm{d}t_3 \\ &= \int_0^{\infty} f_G(t_3) \int_0^{\infty} f_m(t_{m,2}) \\ &\quad \times \left[\sum_{j=0}^{n-1} \frac{\lambda^j (t_3 + t_{m,2})^j \mathrm{e}^{-\lambda(t_3 + t_{m,2})}}{j!} \right] \mathrm{d}t_{m,2} \mathrm{d}t_3 \\ &= \sum_{j=0}^{n-1} \left(\frac{\lambda^j}{j!} \right) \int_{t_3 = 0}^{\infty} f_G(t_3) \mathrm{e}^{-\lambda t_3} \\ &\quad \times \int_{t_{m,2} = 0}^{\infty} f_m(t_{m,2}) \mathrm{e}^{-\lambda t_{m,2}} \left[\sum_{l=0}^{j} \binom{j}{l} t_3^l t_{m,2}^{j-l} \right] \mathrm{d}t_{m,2} \mathrm{d}t_3 \\ &= \sum_{j=0}^{n-1} \left(\frac{\lambda^j}{j!} \right) \sum_{l=0}^{j} \binom{j}{l} \left[\int_{t_3 = 0}^{\infty} f_G(t_3) t_3^l \mathrm{e}^{-\lambda t_3} \mathrm{d}t_3 \right] \\ &\quad \times \left[\int_{t_{m,2} = 0}^{\infty} f_m(t_{m,2}) t_{m,2}^{j-l} \mathrm{e}^{-\lambda t_{m,2}} \mathrm{d}t_{m,2} \right] \end{aligned} \tag{9}$$

From the frequency-domain general derivative of the Laplace transform, for a function $f(t)$ we have

$$\int_{t=0}^{\infty} t^j f(t) \mathrm{e}^{-st} \mathrm{d}t = (-1)^j \left[\frac{f^{*(j)}(s)}{\mathrm{d}s^j} \right]$$

Therefore, (9) is rewritten as

$$\begin{aligned} &\Pr[t_{m,2} + t_3 < t_{m,1}] \\ &= \sum_{j=0}^{n-1} \left[\frac{(-\lambda)^j}{j!} \right] \sum_{l=0}^{j} \binom{j}{l} \left\{ \left[\frac{\mathrm{d}^{(l)} f^*_G(s)}{\mathrm{d}s^l} \right]_{s=\lambda} \right\} \\ &\quad \times \left\{ \left[\frac{\mathrm{d}^{(j-l)} f^*_m(s)}{\mathrm{d}s^{j-l}} \right]_{s=\lambda} \right\} \end{aligned} \tag{10}$$

$$\begin{aligned} &= \sum_{j=0}^{n-1} \left[\frac{(-\lambda)^j}{j!} \right] \sum_{l=0}^{j} \binom{j}{l} \left\{ \left[\frac{\mathrm{d}^{(l)} f^*_G(s)}{\mathrm{d}s^l} \right]_{s=\lambda} \right\} \\ &\quad \times \left\{ \left[\frac{(-1)^{j-l}(n+j-l-1)!}{(n-1)!} \right] \left[\frac{\lambda^n}{(s+\lambda)^{n+j-l}} \right]_{s=\lambda} \right\} \\ &= \sum_{j=0}^{n-1} \left(\frac{1}{j!} \right) \sum_{l=0}^{j} \binom{j}{l} \left\{ \left[\frac{\mathrm{d}^{(l)} f^*_G(s)}{\mathrm{d}s^l} \right]_{s=\lambda} \right\} \\ &\quad \times \left[\frac{(-1)^{j-l}(-1)^j (n+j-l-1)!}{(n-1)!} \right] \left[\frac{\lambda^{n+j}}{(2\lambda)^{n+j-l}} \right] \\ &= \sum_{j=0}^{n-1} \left(\frac{1}{j!} \right) \sum_{l=0}^{j} \binom{j}{l} \left\{ \left[\frac{\mathrm{d}^{(l)} f^*_G(s)}{\mathrm{d}s^l} \right]_{s=\lambda} \right\} \\ &\quad \times \left[\frac{(-\lambda)^l (n+j-l-1)!}{2^{n+j-l}(n-1)!} \right] \end{aligned} \tag{11}$$

If t_3 is a gamma random variable with the shape parameter δ and the rate parameter η, then the Laplace transform of the density function $f_G(t_3)$ is expressed as

$$f^*_G(s) = \frac{\eta^{\delta}}{(s+\eta)^{\delta}} \tag{12}$$

and (11) is rewritten as

$$\Pr[t_{m,2}+t_3<t_{m,1}]=\sum_{j=1}^{n-1}\left(\frac{1}{j!}\right)\sum_{l=0}^{j}\binom{j}{l}\times\left\{\left[\frac{(-1)^l\Gamma(\delta+l)}{\Gamma(\delta)}\right]\left[\frac{\eta^\delta}{(s+\eta)^{\delta+l}}\right]\Big|_{s=\lambda}\right\}\times\left[\frac{(-\lambda)^l(n+j-l-1)!}{2^{n+j-l}(n-1)!}\right]$$
$$=\sum_{j=1}^{n-1}\left(\frac{1}{j!}\right)\sum_{l=0}^{j}\binom{j}{l}\binom{\delta+l}{l}\left[\frac{\eta^\delta}{(\lambda+\eta)^{\delta+l}l!}\right]\times\left[\frac{\lambda^l(n+j-l-1)!}{2^{n+j-l}(n-1)!}\right] \quad (13)$$
$$=\sum_{j=1}^{n-1}\sum_{l=0}^{j}\binom{j}{l}\binom{\delta+l}{l}\left[\frac{(n+j-l-1)!}{j!l!(n-1)!}\right]\times\left[\frac{\lambda^l\eta^\delta}{2^{n+j-l}(\lambda+\eta)^{\delta+l}}\right]$$

The above derivations have been validated by the simulation follows the same approach in [9, 14–16], and the details are omitted. For the dormitory applications considered in this paper (i.e. the delay measurements in Section 4.3), the discrepancies between the analytic model and the simulation are small as illustrated in Fig. 15 (to be elaborated in the next subsection).

4.3 Numerical examples

We have measured the t_m for DormTalk. Fig. 16 plots the histogram of t_m, which can be approximated as a mix of two Erlang distributions, where $E[t_m]=1027.21$ ms. With probability 0.646, the first t_m distribution has the shape parameter $n=2$ and the rate parameter $\lambda=0.344$. With probability 0.354, the second t_m distribution has the shape parameter $n=2$ and the rate parameter $\lambda=0.361$. The patient time t_b was measured in [9], which can be approximated by the gamma distribution with the expected delay $E[t_b]=2047.06$ ms and the variance $\mathrm{V}[t_b]=0.096E[t_b]^2$.

Both the Erlang and the Gamma distributions are often used for time complexity analysis in telecommunications [14–17] because these distributions (or mixtures of the distributions) can be shaped to represent many distributions as well as the measured data. The t_m and t_b distributions obtained from the measurements are used in simulation to compute $\Pr[K=1]=0.9703$ which indicates that the user experience is good.

On the fairness problem, we investigate the probability $\Pr[t_{m,2}+t_3\geq t_{m,1}]=1-\Pr[t_{m,2}+t_3<t_{m,1}]$ which is the complement of the unfair probability mentioned in Section 4.2. We conduct three experiments to simulate the reservation competition process with t_3 approximated by the gamma distribution with $E[t_3]=2047.06$ ms and different variances. Fig. 16 plots the $\Pr[t_{m,2}+t_3\geq t_{m,1}]$ curves against t_3 variance ranging from $10^{-3}E[t_3]^2$ to $10^4E[t_3]^2$. The traced-driven simulation uses the measured $t_{m,1}$ and $t_{m,2}$ to obtain $\Pr[t_{m,2}+t_3\geq t_{m,1}]$. The distribution-driven simulation uses the mix of two Erlang distributions to approximately $t_{m,1}$ and $t_{m,2}$. The analytic analysis uses (13) to compute the probability.

The discrepancies among the three experiments are at most 0.0258. Hence, the analytic model captures the reality quite well. In our measurements, $V[t_3]=0.096E[t_3]^2$, and $\Pr[t_{m,2}+t_3\geq t_{m,1}]=1$. In practice, it is rare to have $V[t_3]\geq E[t_3]^2$ and the fair probability $\Pr[t_{m,2}+t_3\geq t_{m,1}]\geq 0.995$ when $V[t_3]\leq E[t_3]^2$ in all experiments. Therefore, the 'first-press–first-serve' property is preserved.

5 Conclusion

This paper showed how edge computing and cloud computing can be nicely integrated to build a smart campus environment in NCTU. Based on an IoT device management platform called IoTtalk, we developed edge IoT computing platform called DormTalk for dormitory applications in NCTU. Conventional approaches are difficult to support interaction of IoT servers from different platforms. Our approach proposed a novel solution that uses a cyber IoT device to nicely bridge the IoTtalk server in the cloud and the DormTalk servers at the edges without modifying the servers in different IoT platforms. With the edge DormTalks, the IoT packet traffic between MapTalk in the cloud and the dormitory appliances/sensors at the edges can be reduced. Also, our edge and cloud computing structure allows edge DormTalks to operate independently from MapTalk, which enhances availability, scalability, and security. We investigate the response time performance of the edge DormTalk server, which showed that smartphones can effectively control the dormitory appliances with good user experience. We have also investigated the fairness issue when multiple smartphones compete to control a dormitory appliance or resources. Our study indicated that with the edge DormTalk server, the fairness rule for 'first-press–first-serve' is satisfactorily followed.

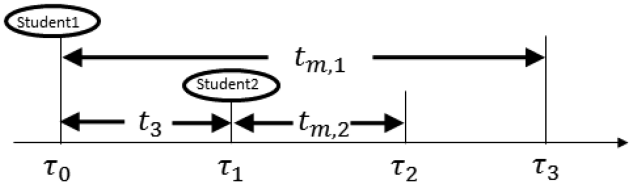


Fig. 14 *Timing diagram for fairness issue*

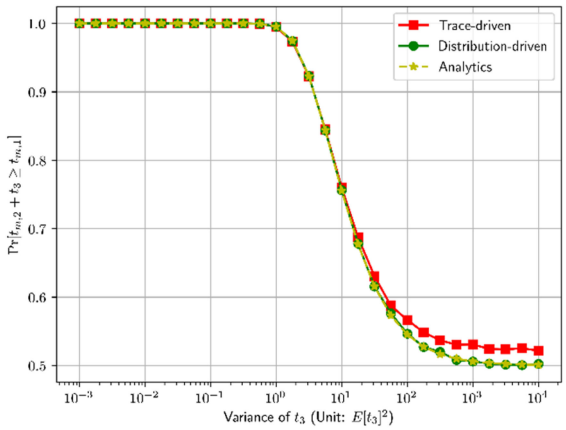


Fig. 15 *T_m histograms*

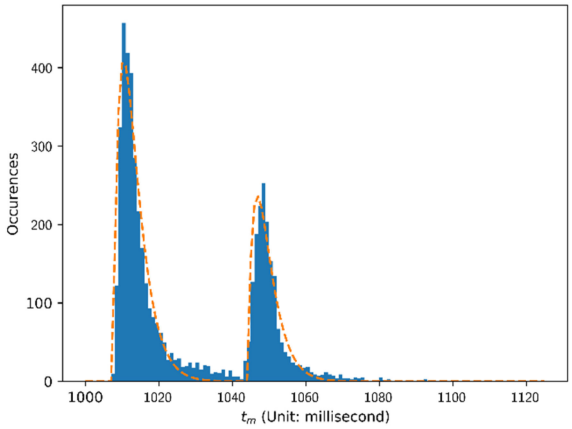


Fig. 16 *Fair probability*

6 References

[1] Lin, Y.-B., Chen, L.-K., Shieh, M.-Z., *et al.*: 'Campustalk: IoT devices and their interesting features on campus applications', *IEEE Access*, 2018, **6**, (1), pp. 26036–26046

[2] Lin, Y.-B., Lin, Y.-W., Huang, C.-M., *et al.*: 'Iottalk: a management platform for reconfigurable sensor devices', *IEEE Internet Things J.*, 2017, **4**, (5), pp. 1152–1562

[3] Lin, Y.-B., Lin, Y.-W., Chih, C.-Y., *et al.*: 'Easyconnect: a management system for iot devices and its applications for interactive design and art', *IEEE Internet of Things J.*, 2015, **2**, (6), pp. 551–561
[4] Lin, Y.-W., Lin, Y-B., Hsiao, C.-Y., *et al.*: 'IoTtalk-RC: sensors as universal remote control for aftermarket home appliances', *IEEE Internet Things J.*, 2017, **4**, (4), pp. 1104–1112
[5] Lin, Y.-B., Tseng, H.-C., Lin, Y.-W., *et al.*: 'NB-IoTtalk: a service platform for fast development of NB-IoT applications'. *IEEE Internet Things J.*, 2018, doi:10.1109/JIOT.2018.2865583
[6] Lin, Y.-W., Lin, Y.-B., Yang, M.-T., *et al.*: 'Ardutalk: an arduino network application development platform based on IoTtalk', *IEEE Syst. J.*, 2017, pp. 1–9, doi:10.1109/JSYST.2017.2773077
[7] Lin, Y.-W., Lin, Y.-B., Liu, C.-Y.: 'AItalk: a tutorial to implement AI as IoT devices', *IET Netw.*, 2018, doi:10.1049/iet-net.2018.5182
[8] 'scikit-learn, machine learning in python'. Available at http://scikit-learn.org, accessed 30 September 2018
[9] Lin, Y.-B., Huang, C.-M., Chen, L.-K., *et al.*: 'Morsocket: an expandable iot-based smart socket system', *IEEE Access*, 2018, **6**, pp. 53123–53132
[10] Lin, Y.-B., Lin, Y.-W., Shieh, M.-Z, *et al.*: 'Maptalk: mosaicking physical objects into the cyber world', *Cyber-Phys. Syst.*, 2018, **4**, (3), pp. 1–19
[11] Lin, Y.-B., Wang, S.-Y., Huang, C.-C., *et al.*: 'SDN approach for aggregation/disaggregation of sensor data', *Sensors (ISSN 1424-8220)*, 2018, **18**, (7), doi:10.3390/s18072025
[12] 'Onem2m. Standards for M2M and the internet of things'. 2016. Available at http://www.onem2m.org/, accessed 13 July 2017
[13] 'Allseen alliance'. 2016. Available at https://allseenalliance.org/, accessed 13 July 2017
[14] Huang, D.-W., Lin, P., Gan, C.-H.: 'Design and performance study for a mobility management mechanism (WMM) using location cache for wireless mesh network', *IEEE Trans. Mobile Comput.*, 2008, **7**, (5), pp. 546–556
[15] Hong, C.-Y., Pang, A.-C.: '3-approximation algorithm for joint routing and link scheduling in wireless relay networks', *IEEE Trans. Wireless Commun.*, 2009, **8**, (2), pp. 856–861
[16] Tsai, S.-Y., Tsai, M.-H., Sou, S.-I.: 'Reducing energy consumption by data aggregation in M2M networks', *Wireless Pers. Commun.*, 2014, **74**, (4), pp. 1231–1244
[17] Lin, P., Wu, S.-H., Chen, C.-M., *et al.*: 'Implementation and performance evaluation for a ubiquitous and unified multimedia messaging platform', *ACM Wireless Netw.*, 2009, **15**, (2), pp. 163–176