NB-IoTtalk: A Service Platform for Fast Development of NB-IoT Applications

Yi-Bing Lin, *Fellow, IEEE*, Hung-Chun Tseng, Yun-Wei Lin, and Ling-Jy Chen

***Abstract*—NarrowBand Internet of Things (NB-IoT) is considered as a promising wireless communications technology for Internet of Things (IoT) especially for the outdoor environment. Many outdoor IoT applications involve large numbers of homogeneous NB-IoT devices. It is tedious to specify and accommodate these devices during application development. To resolve this issue, this paper proposes a service platform for fast development of NB-IoT applications called NB-IoTtalk. This platform utilizes a tag mechanism to provide an easy-to-manipulate graphical user interface (GUI) to accommodate a large number of NB-IoT devices in an application and transparently show them in a visual map. Our approach automatically creates and parses the device profile used to interpret the payload of an NB-IoT message. We then use a smart parking lot application as an example to investigate the event-triggered reporting of NB-IoT in terms of the time-to-live report frequency and the outage detection accuracy. Our study provides the guidelines to set the time-to-live interval for event-triggered NB-IoT applications.**

***Index Terms*—event-triggered, Low Power Wide Area Network (LPWAN), NarrowBand Internet of Things (NB-IoT), parking sensor, time-to-live.**

I. Introduction

INTERNET of Things (IoT) is one of the mainstreams in information and communications technology [1]-[3]. Over the past 20 years, outdoor IoT wireless applications have been developed based on the cellular telecommunications technologies. These applications were deployed in either GPRS/LTE based broadband services [4] or WLAN based services [5]. Recently, Low Power Wide Area Network (LPWAN) technologies have been developed for IoT applications with low data rate transmission, e.g., LoRA [6], EC-GSM-IoT [7] and NarrowBand Internet of Things (NB-IoT) [8], [9]. To promote smart campus, National Chiao Tung University (NCTU) is deploying several IoT-based smart campus applications including temperature and PM2.5 monitoring, parking, emergency button, and dog tracking based on an IoT application management platform called IoTtalk [10]-[13]. These applications utilize the NB-IoT devices. Before elaborating on NB-IoT, we first introduce several LPWAN technologies for IoT.

Y.-B. Lin, H.-C. Tseng, and Y.-W. Lin are with the Department of Computer Science, National Chiao Tung University, Hsinchu 300, Taiwan (e-mail: liny@cs.nctu.edu.tw; u9220043@gmail.com; jyneda@gmail.com). L.-J. Chen is with Institute of Information Science, Academia Sinica., Taipei 115, Taiwan (e-mail: cclljj@iis.sinica.edu.tw). This work was supported in part by the Ministry of Science and Technology (MOST) under Grant 106-2221-E-009 -006 and Grant 106-2221- E-009 -049 - MY2, the Ministry of Education through the SPROUT Project-Center for Open Intelligent Connectivity of National Chiao Tung University, Taiwan, and Academia Sinica AS-105-TP-A07.

LoRaWAN is a LPWAN specification intended for battery operated wireless IoT devices, which provides seamless interoperability among IoT devices without the need of complex local installations and gives back the freedom to the users, the developers, and the businesses enabling the roll out of IoTs. In our LoRA deployment at NCTU, the transmission performance is not very good because LoRA is operated in the unlicensed bands. The details can be found in [14].

As an extension technology of GSM, EC-GSM-IoT's uplink/downlink transmission rate ranges from 70 Kbps to 350 Kbps with Gaussian filtered Minimum Shift Keying (GMSK) modulation, and up to 240kbps with 8-Phase-shift keying (8-PSK) modulation. Every GSM base station can accommodate 50,000 EC-GSM-IoT devices, where the standby power of a device is 5Wh. for at least 10 years. Compared with GPRS [4], the radio coverage of EC-GSM-IoT at 33dBm has been increased by 20dB.

While compatible with FDD and TDD LTE [15], 3GPP eMTC has removed some broadband transmission features not needed in IoT to support 100,000 devices. At 3GPP Release 12 [16], eMTC is exercised at a high transmission rate up to 1Mbps using 1.08MHz bandwidth, which has 12dB radio coverage increase as compared with GPRS. eMTC also supports Voice Over LTE (VoLTE) [15], which is not found in other IoT technologies. The standby power of an eMTC device is 5Wh for at least 10 years.

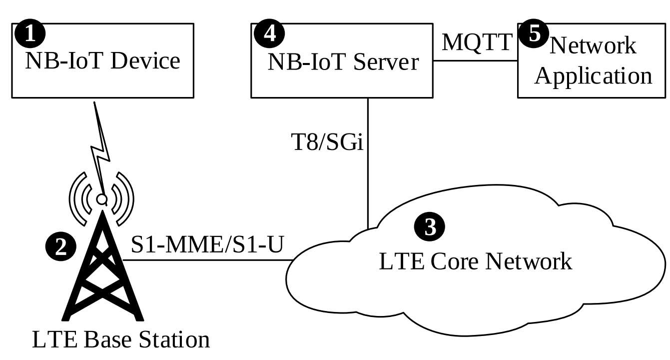

Fig. 1. The NB-IoT Network Architecture.

NB-IoT is exercised with 180 kHz bandwidth under the LTE infrastructure. Fig. 1 illustrates the NB-IoT network architecture. An NB-IoT device (Fig. 1 (1)) consists of the sensors/actuators and the NB-IoT wireless module that communicates with an LTE base station (Fig. 1 (2)), where the unlink/downlink transmission rate is 250Kbps. The LTE base station connects to the LTE core network (Fig. 1 (3)) through the S1-MME/S1-U interface. The core network connects to the NB-IoT Application Server (AS; Fig. 1 (4)) through the T8/SGi interface.

The standby power of an NB-IoT device is 5Wh for at least 10 years. The NB-IoT radio coverage is the same as EC-GSM-IoT (up to 20 Km in diameter), which supports 200,000 devices per base station. Furthermore, NB-IoT is extended with multiple 180 kHz carriers to accommodate up to millions of devices. To achieve better radio penetration, it is recommended to operate NB-IoT at sub-1G spectrums (such as 700MHz, 800MHz or 900MHz).

NB-IoT aims to establish a LPWAN through the existing LTE infrastructure. There are three alternatives for NB-IoT deployment. In the Standalone mode, independent bandwidth is allocated so that NB-IoT will not interfere with the LTE operation. The Guard Band mode utilizes the LTE guard band, which allows the operator to support NB-IoT without requiring new spectrum and with minimal impact to LTE. The In-Band mode directly occupies a LTE bandwidth to co-exist with the LTE operation. In 2017, Chunghwa Telecom (the largest telecom company in Taiwan) deployed Taiwan's first NB-IoT service with the In-Band mode in NCTU.

Based on the above description, NB-IoT has better penetration than GSM, which gives good qualities for both indoor and outdoor transmissions. To develop an NB-IoT service, the developer needs to create a "network application" (Fig. 1 (5); typically an application server in the Internet) that connects to the NB-IoT Server. This network application interacts with NB-IoT devices to provide a specific IoT service. Development of an NB-IoT network application is a tedious task, where a device profile is used to define how a network application (Fig. 1 (5)) communicates with the corresponding NB-IoT devices (Fig. 1 (1)). Specifically, the profile is used by the network application to interpret the payload of a message sent from the NB-IoT device. We will elaborate more on the device profile later.

This paper shows how to develop fast NB-IoT network applications through an application-level platform called NB-IoTtalk. This platform integrates IoTtalk [10]-[13] with a tag mechanism, which allows a developer to easily create IoT applications (e.g., smart parking lot) with a large number of NB-IoT devices through a web page. Such tool has not been found in the literature. From the viewpoint of an NB-IoT system, NB-IoTtalk is a network application, and therefore, can be accommodated by LTE operators worldwide. This paper is organized as follows. Section 2 describes the NB-IoTtalk network architecture, and shows how to automatically create the network applications from an existing device profile. Section 3 proposes the tag mechanism and automatic creation of the device profile. Section 4 uses NCTU parking lot as an NB-IoT application and shows its performance in terms of the time-to-live report frequency and the outage detection accuracy.

II. The NB-IoTtalk Architecture and Automatic Network Application Creation

The philosophy of IoTtalk is centered at two concepts called "device feature" (DF) and "device model" (DM). An input DF is a sensor (such as a temperature sensor) or a controller unit (such as a button). An output DF is an actuator (such as a lamp). A DM is a collection of DFs. In IoTtalk, every IoT device can be partitioned into an input and an output devices. The input device is a subset of the IoT device, which consists of the input DFs of that device. Similarly, the output device consists of the output DFs of that device. In [12], we show how to automatically create Arduino-based applications by considering an Arduino board as an IoTtalk device. In this paper, we further extend this capability to transparently map a group of homogeneous NB-IoT devices to an IoTtalk device called "NB-IoTtalk" device. This device (Fig. 2 (1)) consists of two major components: the NB-IoT Application (Fig. 2 (4)) is responsible for communication with the NB-IoT Server (Fig. 2 (2); see also Fig. 1 (4)) and a Device Application (DA; Fig. 2 (5)) is responsible for interaction with the IoTtalk Server (Fig. 2 (3)) through HTTP. Multiple DAs can be connected to the NB-IoT Application to support various services such as parking, dog tracking and smart home applications (Fig. 2 shows three IoTtalk DAs).

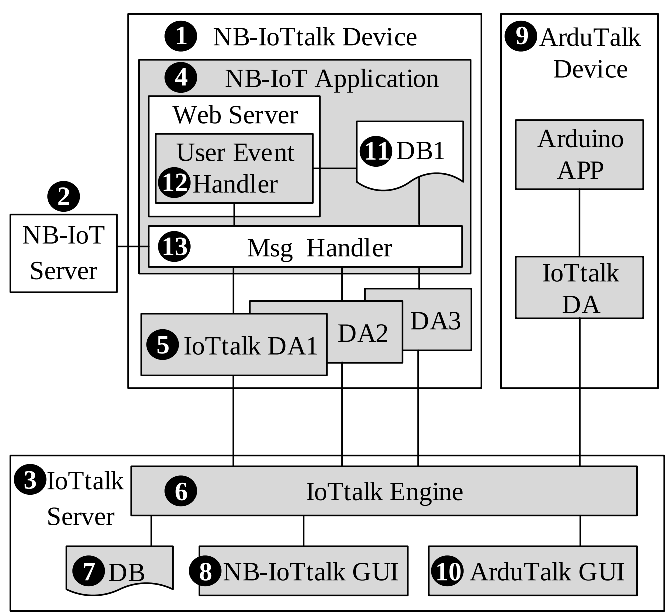

Fig. 2. NB-IoTtalk: Integrating NB-IoT and IoTtalk.

The IoTtalk Server consists of two components: the engine and multiple GUIs tailored for various application systems (such as NB-IoTtalk and ArduTalk). The IoTtalk engine (Fig. 2 (6)) provides MQTT or HTTP based RESTful application programming interfaces for the IoTtalk DA to deliver/retrieve the IDF/ODF information to be saved in a database DB (Fig. 2 (7)). The NB-IoTtalk GUI (Fig. 2 (8)) provides a friendly web-based user interface to quickly establish connections and meaningful interactions among the NB-IoT devices. Through the GUI, a user instructs the IoTtalk engine to execute desired tasks to create or set up device features, functions, and connection configurations. From the viewpoint of the NB-IoT system, the NB-IoTtalk device is a network application (Fig. 1 (4)). From the viewpoint of IoTtalk, the NB-IoTtalk device is an IoTtalk device. We have built other IoTtalk-based application systems using IoTtalk DAs. For example, by installing the IoTtalk IDE (Integrated Development Environment) into an Arduino board, we build an ArduTalk device (Fig. 2 (9)) [13] connected to the IoTtalk engine, which can be accessed and manipulated by the ArduTalk GUI (Fig. 2 (10)). In the GUI window, an input device is represented by an icon placed at the left of the window (Fig. 3 (a), (c), (e)), which consists of smaller icons that represent IDFs (Fig. 3 (1), (2), (5), and (7)). Similarly, an output device is represented by an icon placed at the right-hand side of the window (Fig. 3 (b), (d)), which includes ODF icons (Fig. 3 (3), (4), (6), and (8)). Note that the ParkingStatus device has one IDF and two ODFs, and therefore is represented by two icons (Fig. 3 (b) and (c)). By connecting the IDFs to the ODFs through the line segments (e.g., Joins 1-4), the devices interact with each other without the need of any programing effort. Details of the devices in this example will be elaborated in Section 4.

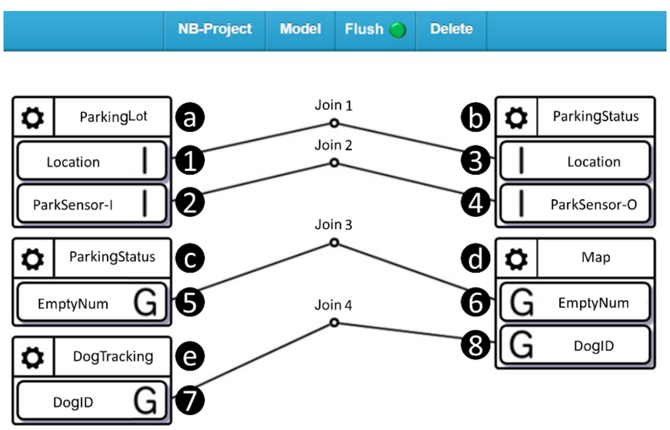

Fig. 3. An example of NB-IoT applications configured in the NB-IoTtalk GUI.

The network application of an NB-IoTtalk device (Fig. 2 (1)) can be automatically created by a device profile, and is illustrated as input and/or output device icons in the NB-IoTtalk GUI. This profile is typically a JSON file specified by the NB-IoT operator or the NB-IoT device manufacturer. Our approach saves this profile in the DB1 database (Fig. 2 (11)). For the device profile given by a third-party NB-IoT device manufacture, the NB-IoT application interacts with the IoTtalk engine to automatically create the device icon shown in the NB-IoTtalk GUI. This profile is preloaded into the database DB1 (Fig. 2 (10)), which is retrieved and parsed by the Web Server (Fig. 2 (12)) to encode/decode the DFs and their values in the payload of every NB-IoT message. For example, the profile for the ParkingLot device is listed below.

```
Line 1. "IMEI": "3588780065690",
Line 2. "IMSI": "460100000001234",
Line 3. "DM": ParkingLot
Line 4. "DFs": [{
Line 5.   "DF": Location
Line 6.   "VALUE": //[float, float]
          },{
Line 7.   "DF": ParkSensor
Line 8.   "VALUE": // boolean
          }]
```

Fig. 4. The device profile for ParkSensor.

In the above JSON code, the device ID, i.e., International Mobile Equipment Identity (IMEI) is given at Line 1. The International Mobile Subscriber Identity (IMSI) number of the SIM card inserted in this device is given at Line 2. Note that different NB-IoT operators or device manufacturers may provide different formats for the profile. For example, in the profile provided by Chunghwa Telecom, the IMEI field is named DEVID and the IMSI field is named SIMID. When the Web Server parses Line 3, it creates the "ParkingLot" device icon (Fig. 3 (a)). Lines 4-8 describe the DFs in the ParkingLot DM. When Lines 5 and 6 are parsed, the Location IDF icon is created (Fig. 3 (1)). Location data are typically produced by a GPS receiver or can be input through a web page (to be elaborated in Fig. 5 (4)). When Lines 7 and 8 are parsed, the ParkSensor IDF icon is created (Fig. 3 (2)). This sensor gives on/off status to indicate if a parking space is occupied.

After the device profile has been parsed, the user can access the details of the devices through the NB-IoTtalk web page illustrated in Fig. 5. This web page shows all NB-IoT devices (Fig. 5 (1)) of a specific device model (Fig. 5 (2)). The user can select the devices in a group, e.g., "NCTU-P2" (Fig. 5 (3)) to indicate the parking sensors installed in NCTU's parking lot 2. In Fig. 5, the first and the third devices are selected.

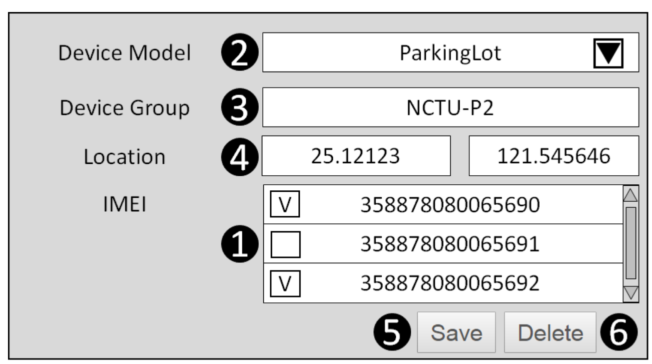

Fig. 5. The NB-IoTtalk web page.

Due to cost consideration, most commercial parking sensors are not equipped with the GPS receivers. In this case, the NB-IoTtalk web page allows manually specifying the location of the parking lot (Fig. 5 (4)).

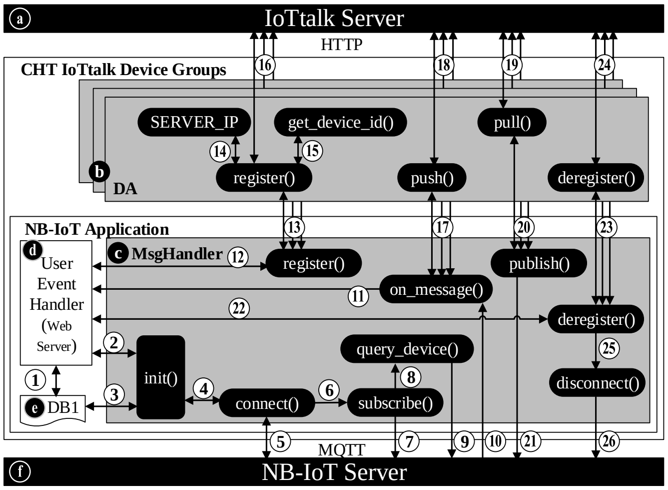

Fig. 6. The functional blocks of the NB-IoTtalk Device.

Fig. 6 illustrates the functional blocks of the NB-IoTtalk Device in Fig. 2 (1). To connect the network application to the NB-IoT Server (Fig. 6 (f)), the following steps are executed.

Steps 1-3: The User Event Handler of the Web Server (Fig. 6 (d)) initializes the MsgHandler (Fig. 6 (c)) to retrieve the MQTT host address of the NB-IoT Server and the related information from DB1 (Fig. 6 (e)).

Steps 4 and 5: The MsgHandler (Fig. 6 (c)) then invokes connect() to establish an MQTT connection with the NB-IoT Server.

Steps 6-9: The MsgHandler subscribes the MQTT topics (i.e., Send/IoTtalk/pub in our example) through subscribe(), and then invokes query_device() to retrieve the IMEIs and the DFs of all NB-IoT devices connected to the NB-IoT Server.

Steps 10 and 11: The NB-IoT Server returns the queried information back to the User Event Handler through on_message(). The information is saved in DB1. At this point, the NB-IoT Application is established, but no DA (Fig. 6 (b)) has been created yet.

After the user has set up an NB-IoT device group, e.g., NCTU-P2, and clicks the "Save" button in Fig. 5 (5), **Steps 12-16** are executed to create the NCTU-P2 DA. Specifically, the User Event Handler catches the "save" event from the browser, and invokes reigster() to create the DA for NCTU-P2, and then registers it to the IoTtalk Server (Fig. 6 (a)) with SERVER_IP.

When an NB-IoT device in NCTU-P2 sends the data to the network, **Steps 10, 17, and 18** are executed: the NB-IoT Server forwards the data to the IoTtalk Server through on_message() in the NB-IoT Application and push() in the NCTU-P2 DA.

When the IoTtalk Server sends the data to an NB-IoT device in NCTU-P2, **Steps 19-21** are executed: the IoTtalk Server sends the data to the NB-IoT Server by invoking pull() in the DA and publish() in the NB-IoT Application.

When the user clicks the "Delete" button of NCTU-P2 in Fig. 5 (6), **Steps 22-24** are executed: The User Event Handler catches the "delete" event from the browser, and invokes deregister() to delete the DA for NCTU-P2 after deregistering it from the IoTtalk Server.

When the user terminates the NB-IoT service, the User Event Handler disconnects the MQTT connection with **Steps 22-26**: the User Event Handler first invokes deregister() to deregister and delete all DAs (**Steps 22-24**), and then calls disconnect() to end the connection to the NB-IoT Server (**Steps 25** and **26**).

III. Automatic Creation of Device Profile

Both the NB-IoT device and the network application should follow the same device profile for a specific NB-IoT device. NB-IoTtalk automatically creates the profile by interaction between the Web Server (Fig. 2 (12)) and the IoTtalk engine (Fig. 2 (6)). The created profile is used as the IDE to be installed in the NB-IoT devices as what we did for Arduino [12]. The IoTtalk engine allows creation of the DFs and accumulates them in the database DB (Fig. 2 (7)). These DFs can be reused by various applications. NB-IoTtalk automatically translates these DFs into JSON format of the device profile. Many important pieces of the DF information may not be specified in the device profile, for example, the range [Min, Max] and the unit of the values (e.g., ℃ or ℉ for temperature). In NB-IoTtalk, such information is stored in DB1 and will be handled by the NB-IoT application. To our knowledge, transparent DF creation and reuse through the GUI has not been found in the literature. To accommodate a large number of homogeneous NB-IoT devices, we propose the "tag" mechanism. In NB-IoTtalk, every DF may include two types of parameters.

- The attribute parameters describe the values of a DF. For example, a PM2.5 sensor is mapped to the PM2.5 DF that has one attribute parameter to produce the particulate matter 2.5 measure in μ g/m3. A GPS receiver is mapped to the Location DF that has two attribute parameters to specify the latitude and the longitude.
- The tag parameters (or tags in short) provide extra information associated with the DF. There are five types of tags: Identity (ID), Geographic Data (GeoData), Time (T), Battery (B), and Privacy (P).

The early IoTtalk version [11], [12] only defined the attribute parameters for a DF. In NB-IoTtalk, the tags are defined to conveniently manipulate the DFs. A tag itself is an attribute parameter in the early IoTtalk version. For example, the GeoData tag is derived from the Location DF (for a GPS sensor or an iBeacon sensor) with two attribute parameters (i.e., the latitude and the longitude), which provides geographic location information of the DF. This tag is used to associate the DF with a visual map.

The Time (T) tag specifies the time when a value of the DF is generated. The Battery (B) tag indicates the battery life of that DF, which is used for energy management of the DF (more precisely, the device of that DF). The Privacy (P) tag gives the privacy level of the DF, which is used when the privacy regulation is enforced in the IoT applications.

In this paper, the most important tag is "Identity" (ID) used to support applications with a large number of homogeneous NB-IoT devices. The ID tag is derived from the Identity DF with one attribute parameter. In this paper, the ID tag specifies the IMEI of an NB-IoT device. When NB-IoTtalk receives a message sent from an NB-IoT device, it associates the IMEI of that device to the corresponding DF(s). In the previous version of IoTtalk, most applications typically involve heterogeneous indoor IoT devices. For these devices, the DFs only need to be specified through their attribute parameters. For example, an indoor PM2.5 measurement application uses a PM2.5 IDF without the ID tag. On the other hand, for long-range wireless technologies such as NB-IoT, LoRA or Sigfox, many applications involve homogeneous outdoor IoT devices. In these applications, the DFs of the homogeneous devices are distinguished by their identities. With the ID tag, the NB-IoTtalk GUI can easily specify and accommodate any number (e.g., thousands) of homogeneous devices in an application and allows the number to dynamically change (i.e., you do not need to specify the number of the NB-IoT devices and can add or remove them). For the outdoor PM2.5 measurement application with thousands of NB-IoT-based PM2.5 sensor devices, these devices are represented by one PM2.5 DF associated with the ID tag in NB-IoTtalk, where the IDs distinguishes multiple homogeneous NB-IoT devices grouped in this application.

The IoTtalk engine provides the Device Feature Window to create and manage the DFs. In this web-based window, the user can edit a new or an existing DF by manipulating their attribute parameters. In Fig. 7, the GUI first displays two radio buttons for IDF/ODF selection (initially, IDF is selected; see Fig. 7 (a)) and the "DF Name" list shows all DFs for the selected IDF/ODF type (Fig. 7 (c)). The first item of the list is "add new DF", which can be clicked to create a new DF (to be elaborated). If the user presses the ODF radio button (Fig. 7 (b)), all IDFs in the "DF Name" list are replaced by all ODFs stored in the DB.

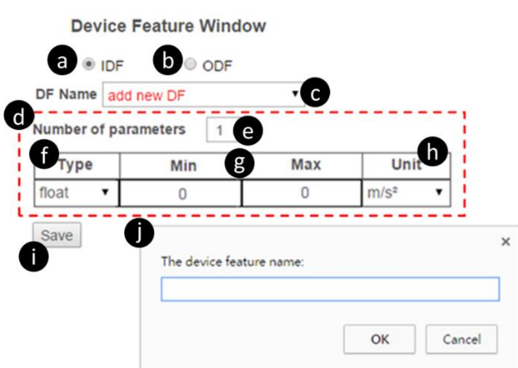

Fig. 7. Device Feature Window.

To create a new DF, the user selects the "add new DF" in the "DF Name" list, and the GUI pops up the Attribute Parameter module (Fig. 7 (d)). The rows of the module are created based on the number of the attribute parameters (Fig. 7 (e)). The default number is one. For each of the parameters, the Attribute Parameter module includes

- the Type (i.e., the data types such as float, string and so on; see Fig. 7 (f)),
- the Min (minimal) and Max (maximal) values (Fig. 7 (g)), and
- the Unit (e.g., cm, m/s2 and so on; see Fig. 7 (h)).

For an IDF, the Min/Max values can be automatically assigned through a dynamic ranging mechanism [11] and the user does not need to fill these fields. For an ODF, if the Min/Max fields are not filled, the ODF-parameters take arbitrary values without range limits. The user edits them according to the characteristics of the ODF provided in the manufacture's data sheet. For example, the status value of the ParkSensor DF has the integer range [0, 1]. When the "Save" button is clicked (Fig. 7 (i)), the GUI pops a dialog box for inputting the name of the new DF (Fig. 7 (j)). Then the IoTtalk engine stores the DF information into the DB (Fig. 2 (7)). For example, in the Device Feature Window, if we select IDF (Fig. 7 (a)), one parameter (Fig. 7 (e)), the integer type (Fig. 7 (f)), the min-max range [0, 1] (Fig. 7 (g)) and NULL unit (Fig. 7 (h)), and give the DF name "ParkSensor" (Fig. 7 (j)), then Lines 7 and 8 in Fig. 4 is automatically generated. At the same time, the min-max range and the unit for ParkSensor are saved in DB1.

The DFs created by the IoTtalk engine are used to build the DMs. In the NB-IoTtalk GUI, every DM is represented by a device icon that illustrates the DFs of the DM (see Fig. 3 (a)-(e)). If a DF of the DM has a tag parameter, then the first letter of that tag is illustrated in the DF icon. For example, the EmptyNum and the DogID DFs are labelled by "G" to indicate that they are associated with GeoData (Fig. 3 (5), (6), (7) and (8)). Similarly, the ParkSensor DFs associated with ID are labeled "I" (Fig. 3 (1), (2), (3) and (4)). If a DF does not have any tag, then no initialized letter is illustrated in the DF icon.

If no DF is associated with ID in the DM, then the user can only connect one device of this DM to NB-IoTtalk by mapping the device to the device icon in the NB-IoTtalk GUI. On the other hand, if any DF of the DM is associated with the ID tag, a group of NB-IoT devices are mapped to that device icon. In this way, we can quickly and conveniently build the applications for NB-IoT devices and replace the devices by others with different tags.

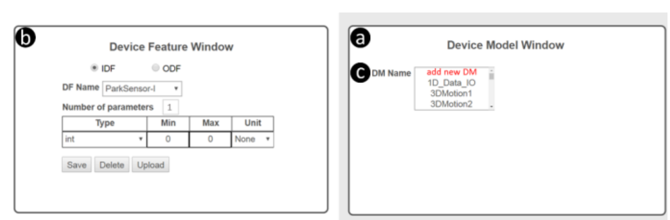

Fig. 8. Device Model Window.

The DMs are manipulated in the Model Management Window. The user can select a DM through the "DM Name" pull-down menu (Fig. 8 (a)) in the Device Model Window. The first item in the list is "add new DM" (Fig. 8 (c)) that can be clicked to create a new model. The Device Feature Window (Fig. 8 (b)) is also shown besides the Device Model Window for the user's benefit: the user may need to know the details of a specific device feature when he/she is configuring a DM.

Whether a DF has any tag or not depends on the DM that includes this DF. The Device Model Window allows the user to manipulate the DM with the Tag Parameter list (Fig. 9 (e); to be elaborated later). To create a new DM, the user selects the "add new DM" in the "DM Name" pull-down menu, and the GUI pops up both the DF module (Fig. 9 (a)) and the "Add/Delete DF" module (Fig. 9 (b)) in the Device Model Window. The DF module lists the DFs of the DM. For a new model, the DF module is empty initially. To add a DF to this DM, the user first selects IDF or ODF from one of the two radio buttons in the "Add/Delete DF" module. For example, "IDF" is selected in Fig. 9 (c). Then the "Add/Delete DF" module shows all IDFs (Fig. 9 (d)) stored in the DB (Fig. 2 (7)). When the user selects a DF in the "Add/Delete DF" module, the DF is automatically displayed in the DF module (Fig. 9 (a)). The user may associate the DF with a tag selected from the Tag Parameter list (Fig. 9 (e)). When the NB-IoT device sends a message, the NB-IoTtalk parses its payload to obtain the DF information, retrieve the IMEI from the message and assign it to the ID tag.

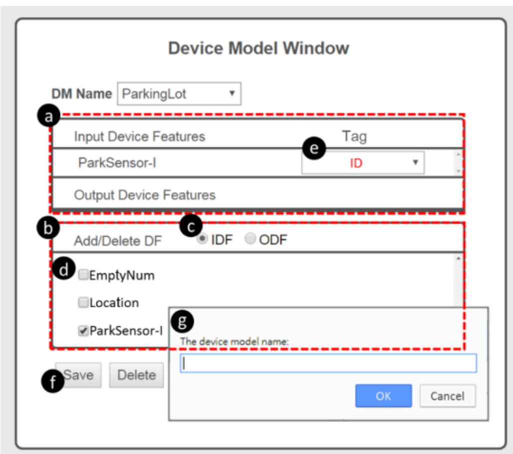

Fig. 9. Device model creation.

The user clicks the "Save" button (Fig. 9 (f)) after he/she has selected all desired DFs. The GUI pops a dialog box for the user to input the name of the new DM (Fig. 9 (g)) to be saved in the DB. At this point, the DM is created, and can be selected and illustrated in the NB-IoTtalk GUI (Fig. 2), which is also shown in the NB-IoTtalk web page (Fig. 5 (2)) that generates Lines 1-3 in Fig. 4.

The user can modify existing DFs and DMs through the Device Feature and the Device Model Windows. The details are omitted. As we just pointed out, when a DM is created through the Device Model Window, the corresponding device profile in the JSON format is also created. This profile can be downloaded into an NB-IoT device as its IDE, and is stored in DB1 (Fig. 2 (11)). The payload of an NB-IoT message is then encoded and decoded at the IDE and DB1.

IV. Configuring the NB-IoT Applications

This section uses smart parking lot and dog tacking as examples to illustrate how NB-IoT applications of NCTU are deployed in NB-IoTtalk. A parking NB-IoT device installed in the NCTU parking lot is surface mounted (Fig. 10), which is 13.5cm in diameter and 3.5cm in height. This device uses a magnetometer where the detection coverage ranges from 500m to 5Km (line-of-sky), and converts the magnetism measure to 0 or 1 to represent the occupancy of the parking space. The device is powered by 3.7V (three 1/2 AA batteries) operated at 900MHz LTE spectrum. We plan to install over 200 parking sensors on campus. At the early stage, 20 NB-IoT parking sensors have been deployed.

Fig. 10. NCTU parking application using NB-IoT.

The NCTU smart parking application can be easily created by configuring three icons in Fig. 3, where ParkingStatus is a display device that receives the location (the Location ODF; Fig. 3 (3)) and the parking sensor status (the ParkSensor ODF; Fig. 3 (4)) from the ParkingLot icon through the links Join 1 and Join 2. In our design, all IDFs connected to ParkingStuatus must associate with the ID tag. ParkingStatus lists the status of each parking sensor and the history line chart for that sensor in a web page (Fig. 11). ParkingStatus also counts the number of empty spaces, which can be accessed through the EmptyNum IDF. This IDF is a counter associated with the GeoData tag. The attribute parameter is an integer representing the number of empty spaces. The EmptyNum IDF is connected to a Map icon (Fig. 3 (d)) through Join 3, and the map shows the number of available parking spaces in the map (see Fig. 12; the dark green rectangle indicates 8 empty parking spaces in this example). All IDFs connected to Map must associate with the GeoData tag.

Parking Status

DeviceID		Status	Time
358878080059196	History	🔴	2018-02-05 18:07:03
358878080059964	History	🟢	2018-02-05 17:48:04
358878080077529	History	🔴	2018-02-05 20:33:10
358878080074724	History	🟢	2018-02-05 15:52:03
358878080060392	History	🟢	2018-02-05 19:01:01

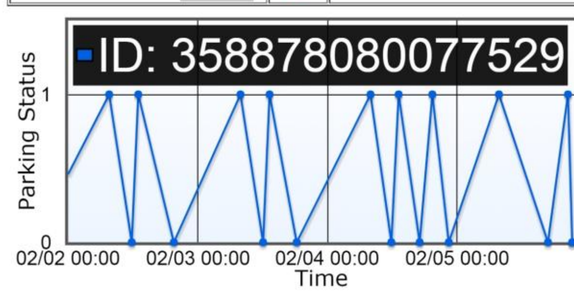

Fig. 11. The web page for ParkingStatus (Green: empty; Red: occupied).

Fig. 3 also includes a DogTracking icon connected to Map through Join 6. The DogID IDF (Fig. 3 (7)) is associated with the GeoData tag to provide the locations of the dogs been tracked (Fig. 13). In this IDF, the dog identities are considered as an attribute parameter. Fig. 12 shows tracking of two dogs represented by green circles marked with the blue number 0 (the first dog) and the orange number 1 (the second dogs) with the tails (colored line segments) indicating the historic traces of the moving dogs. Note that when we first developed the dog tracking application [17], the tag mechanism was not invented, and have to manually create an IDF that includes both DogID and GeoData, which results in extra effort in connecting the dog tracking mechanism to the map.

Fig. 12. The web page for Map.


(a) (b)

Fig. 13. NCTU dog tracking application: (a) a tracker with the GPS receiver; (b) the tracked dog with ID 0.

In the dog tracking application, the tracking sensor (Fig. 13 (a)) periodically reports the location of a dog (Fig. 13 (b)). Modeling of periodic reporting is given in [17], and the details will not be presented in this paper. On the other hand, event-triggered reporting is exercised in the parking lot application, where a parking sensor sends messages only when its status is changed. Event-triggered reporting consumes less power than periodic reporting. The problem of event-triggered reporting is that it is impossible to detect when the sensor fails to send messages to the network. To resolve this issue, a time-to-live (TTL) mechanism is required to detect sensor outage. TTL is similar to periodical reporting but is designed such that message sending is less frequent than periodic reporting. The TTL mechanism is investigated in the next section.

V. Performance Evaluation for Smart Parking Lot

This section uses smart parking lot as an example to elaborate on the performance issue for event-triggered NB-IoT message delivery.

The TTL interval can be determined based on the characteristics of the applications as follows. Let T_p be the period between two consecutive TTL reports. If T_p is small, then short battery life of the sensor is expected. On the other hand, If T_p is large, it takes longer time to detect sensor outage. The tradeoff between TTL reporting frequency and outage detection accuracy can be modeled as follows.

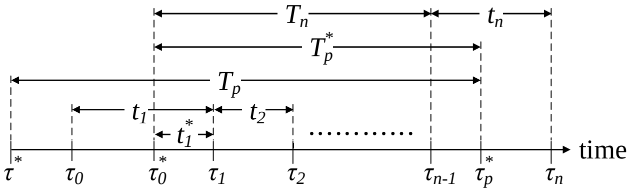

Fig. 14. Timing diagram for modeling sensor outage detection.

Consider the timing diagram in Fig. 14. Suppose that two consecutive TTL reports are sent by a parking sensor at τ^* and τ_p^* respectively, where $\tau^* < \tau_p^*$. Suppose that the NB-IoT communications of the sensor is disconnected at time τ_0^*, where $\tau^* < \tau_0^* < \tau_p^*$. Before τ_0^*, the last status change of the parking sensor occurs at τ_0. For $n \geq 1$, suppose that the parking sensor attempts to send $n-1$ status changes to the network during $[\tau_0^*, \tau_p^*]$ but fails, where the i-th status change occurs at τ_i for $0 < i < n$. If $n = 1$ then there is no status change in $[\tau_0^*, \tau_p^*]$. Let $t_i = \tau_i - \tau_{i-1}$ be i.i.d. random variable with the density function $f(t_i)$. Then $t_1^* = \tau_1 - \tau_0^*$ is the residual life of t_1. The NB-IoT disconnection occurring at τ_0^* can be considered as a random observer of the t_1 period, and from the residual life theorem [18], t_1^* has the density function

$$r_1(t_1^*) = \left\{\frac{1}{\mathrm{E}[t_1]}\right\}\left[1 - \int_{t=0}^{t_1^*} f(t)dt\right] \tag{1}$$

Let $f^*(s)$ be the Laplace Transform of the density function $f(\tau_i)$, then

$$f^*(s) = \int_{s=0}^{\infty} f(\tau_i)\ e^{-s\tau_i} d\tau_i \tag{2}$$

From (1) and (2) the Laplace transform of $r_1(t_1^*)$ is

$$r_1^*(s) = \frac{1 - f^*(s)}{\mathrm{E}[t_1]s} \tag{3}$$

For $n > 2$, let $T_n = \tau_{n-1} - \tau_0^* = t_1^* + \sum_{i=2}^{n-1} t_i$, and $T_2 = t_1^*$. Let $f_n(T_n)$ be the density function of T_n. Then from (3) and the convolution of Laplace transform, we have

$$f_n^*(s) = \left\{\frac{1 - f^*(s)}{\mathrm{E}[t_1]s}\right\}[f^*(s)]^{n-2} \text{ for } n \geq 2 \tag{4}$$

Let $T_p = \tau_p^* - \tau^*$ be the period between two consecutive TTL reports. For $K \geq 1$, suppose that T_p has the K-Erlang density function

$$f_p(T_p) = \frac{\lambda^K {T_p}^{K-1} e^{-\lambda T_p}}{(K-1)!}$$

This distribution is considered because the mixture of the Erlang distributions is widely used in modeling the transmission delay in telecommunications networks. Since the NB-IoT disconnection occurring at τ_0^* is a random observer of the T_p period, from (1), the residual life of T_p is $T_p^* = \tau_p^* - \tau_0^*$, which has the density function

$$\begin{aligned} r_p(T_p^*) &= \left(\frac{\lambda}{K}\right)\left[1 - \int_{t=0}^{T_p^*} f_p(t)dt\right] \\ &= \left(\frac{\lambda}{K}\right)\left[\sum_{k=0}^{K-1} \frac{\lambda^k {T_p^*}^k e^{-\lambda T_p}}{k!}\right] \end{aligned} \tag{5}$$

Let N be the random variable representing that there are $N-1$ status changes in the T_p^* period. Then $\Pr[N = n]$ is the probability that $n-1$ status changes are not received by the network before NB-IoT disconnection is detected. From (5) and for $N>1$, we have

$$\Pr[N=n]$$
$$=\int_{t_n=0}^{\infty}\int_{T_n=0}^{\infty}\int_{T_p^*=T_n}^{T_n+t_n} f(t_n)\, f_n(T_n)\, r_p(T_p^*)\ dT_p^* dT_n dt_n$$
$$=\left(\frac{\lambda}{K}\right)\times$$
$$\int_{t_n=0}^{\infty} f(t_n)\int_{T_n=0}^{\infty} f_n(T_n)\int_{T_p^*=T_n}^{T_n+t_n}\left[\sum_{k=0}^{K-1}\frac{\lambda^k {T_p^*}^k e^{-\lambda T_p^*}}{k!}\right] dT_p^* dT_n dt_n$$
$$=\left(\frac{1}{K}\right)\int_{t_n=0}^{\infty} f(t_n)\int_{T_n=0}^{\infty} f_n(T_n)$$
$$\times\left\{\sum_{k=0}^{K-1}\left[1-\sum_{j=0}^{k}\frac{(\lambda T_p^*)^j e^{-\lambda T_p^*}}{j!}\right]\Bigg|_{T_p^*=T_n}^{T_p^*=T_n+t_n}\right\} dT_n dt_n$$
$$=\left(\frac{1}{K}\right)\int_{t_n=0}^{\infty} f(t_n)\int_{T_n=0}^{\infty} f_n(T_n)$$
$$\times\left(\frac{1}{K}\right)\left\{\sum_{k=0}^{K-1}\left\{\sum_{j=0}^{k}\frac{(\lambda T_n)^j e^{-\lambda T_n}}{j!}\right.\right.$$
$$\left.\left.-\sum_{j=0}^{k}\frac{[\lambda(T_n+t_n)]^j e^{-\lambda(T_n+t_n)}}{j!}\right\}\right\} dT_n dt_n$$
$$=\left(\frac{1}{K}\right)\int_{T_n=0}^{\infty} f_n(T_n)\left\{\sum_{k=0}^{K-1}\left[\sum_{j=0}^{k}\frac{(\lambda T_n)^j e^{-\lambda T_n}}{j!}\right]\right\} dT_n$$
$$-\left(\frac{1}{K}\right)\int_{t_n=0}^{\infty} f(t_n)\int_{T_n=0}^{\infty} f_n(T_n)$$
$$\times\left\{\sum_{k=0}^{K-1}\left\{\sum_{j=0}^{k}\binom{k}{j} {T_n}^j {t_n}^{k-j}\left[\frac{\lambda^j e^{-\lambda(T_n+t_n)}}{j!}\right]\right\}\right\} dT_n dt_n$$
$$=\frac{A-B}{K} \qquad (6)$$

where

$$A=\int_{T_n=0}^{\infty} f_n(T_n)\left\{\sum_{k=0}^{K-1}\left[\sum_{j=0}^{k}\frac{(\lambda T_n)^j e^{-\lambda T_n}}{j!}\right]\right\} dT_n$$
$$=\sum_{k=0}^{K-1}\sum_{j=0}^{k}\left\{\int_{T_n=0}^{\infty} f_n(T_n)\left[\frac{(\lambda T_n)^j e^{-\lambda T_n}}{j!}\right] dT_n\right\} \qquad (7)$$

From the frequency-domain general derivative of Laplace transform, we have

$$\int_{T_n=0}^{\infty} {T_n}^j f_n(T_n) e^{-sT_n} dT_n=(-1)^j\left[\frac{f_n^{*(j)}(s)}{\mathrm{d}s^j}\right] \qquad (8)$$

From (8), (7) is simplified as

$$A=\sum_{k=0}^{K-1}\sum_{j=0}^{k}\left[\frac{(-\lambda)^j}{j!}\right]\left[\frac{f_n^{*(j)}(s)}{\mathrm{d}s^j}\right]\Bigg|_{s=\lambda} \qquad (9)$$

In (6), B is rewritten as

$$B=\int_{t_n=0}^{\infty} f(t_n)\int_{T_n=0}^{\infty} f_n(T_n)$$
$$\times\left\{\sum_{k=0}^{K-1}\sum_{j=0}^{k}\binom{k}{j} {T_n}^j {t_n}^{k-j}\left[\frac{\lambda^j e^{-\lambda(T_n+t_n)}}{j!}\right]\right\} dT_n dt_n$$
$$=\int_{t_n=0}^{\infty} f(t_n)\int_{T_n=0}^{\infty} f_n(T_n)$$
$$\times\left\{\sum_{k=0}^{K-1}\sum_{j=0}^{k}\binom{k}{j}\left(\frac{\lambda^j}{j!}\right)\left[{T_n}^j e^{-\lambda T_n}\right]\left[{t_n}^{k-j} e^{-\lambda t_n}\right]\right\} dT_n dt_n$$
$$=\sum_{k=0}^{K-1}\sum_{j=0}^{k}\binom{k}{j}\left(\frac{\lambda^j}{j!}\right)\left[\int_{T_n=0}^{\infty} f_n(T_n)\, {T_n}^j e^{-\lambda T_n} dT_n\right]$$
$$\times\left[\int_{t_n=0}^{\infty} f(t_n)\, {t_n}^{k-j} e^{-\lambda t_n}\ dt_n\right]$$
$$=\sum_{k=0}^{K-1}\sum_{j=0}^{k}\binom{k}{j}\left[\frac{(-\lambda)^j\lambda^{k-j}}{j!}\right]\left\{\left[\frac{f_n^{*(j)}(s)}{\mathrm{d}s^j}\right]\left[\frac{f^{*(k-j)}(s)}{\mathrm{d}s^{k-j}}\right]\right\}\Bigg|_{s=\lambda} \qquad (10)$$

Substitute (9) and (10) into (6) to yield

$$\Pr[N=n]=\sum_{k=0}^{K-1}\sum_{j=0}^{k}\left[\frac{(-\lambda)^j}{K\,(j!)}\right]$$
$$\times\left\{\left[\frac{f_n^{*(j)}(s)}{\mathrm{d}s^j}\right]\left\{1-\binom{k}{j}\lambda^{k-j}\left[\frac{f^{*(k-j)}(s)}{\mathrm{d}s^{k-j}}\right]\right\}\right\}\Bigg|_{s=\lambda} \qquad (11)$$

For N=1, from (5)

$$\Pr[N=1]=1-\Pr[N>2]$$
$$=1-\int_{t_1^*=0}^{\infty}\int_{T_p^*=t_1^*}^{\infty} r_1(t_1^*)\, r_p(T_p^*)\ dT_p^* dt_1^*$$
$$=1-\int_{t_1^*=0}^{\infty} r_1(t_1^*)\int_{T_p^*=t_1^*}^{\infty}\left[\sum_{k=0}^{K-1}\frac{\lambda^k {T_p^*}^k e^{-\lambda T_p^*}}{k!}\right] dT_p^* dt_1^*$$
$$=1-\int_{t_1^*=0}^{\infty} r_1(t_1^*)\left[\sum_{k=0}^{K-1}\sum_{j=0}^{k}\frac{(\lambda t_1^*)^j e^{-\lambda t_1^*}}{j!}\right] dt_1^* \qquad (12)$$

Similar to the derivation for (7)-(9), (12) is re-written as

$$\Pr[N=1]=1-\sum_{k=0}^{K-1}\sum_{j=0}^{k}\left[\frac{(-\lambda)^j}{j!}\right]\left[\frac{r_1^{*(j)}(s)}{\mathrm{d}s^j}\right]\Bigg|_{s=\lambda} \qquad (13)$$

For $K=1$, (12) is simplified as

$$\Pr[N=n]=f_n^*(\lambda)[1-f^*(\lambda)] \qquad \text{for } N>1 \qquad (14)$$

and (13) is re-written as

$$\Pr[N=1]=1-r_1^*(s)=1-\frac{1-f^*(\lambda)}{\mathrm{E}[t_1]\lambda} \qquad (15)$$

Substitute (4) into (14) to yield

$$\Pr[N=n]=\frac{[1-f^*(\lambda)]^2[f^*(\lambda)]^{n-2}}{\mathrm{E}[t_1]\lambda} \qquad \text{for } N>1 \qquad (16)$$

If t_i has the Gamma distribution with the expected value $\mathrm{E}[t_i]=\alpha/\beta$, we have

$$f(t_i)=\frac{\beta^\alpha {t_i}^{\alpha-1} e^{-\beta t_i}}{\Gamma(\alpha)} \qquad (17)$$

where α is the shape parameter and $1/\beta$ is the scale parameter. The Gamma distribution is a general form of the Erlang distribution, and is widely used in telecommunications network modeling [19], [20]. The Laplace transform of (17) is

$$f^*(s) = \frac{\beta^\alpha}{(s+\beta)^\alpha} \tag{18}$$

Substitute (18) into (16) and (15) to yield

$$\Pr[N=n] = \begin{cases} \left(\frac{\beta}{\alpha\lambda}\right)\left[1-\left(\frac{\beta}{\lambda+\beta}\right)^\alpha\right]^2\left(\frac{\beta}{\lambda+\beta}\right)^{\alpha(n-2)} & \text{for } N>1 \\ 1-\left(\frac{\beta}{\alpha\lambda}\right)\left[1-\left(\frac{\beta}{\lambda+\beta}\right)^\alpha\right] & \text{for } N=1 \end{cases} \tag{19}$$

We validate the boundary conditions of (19) as follows. For $N=1$, as λ approaches ∞, $\Pr[N=1]$ should approach 1. We have

$$\lim_{\lambda\to\infty}\Pr[N=1] = \lim_{\lambda\to\infty} 1-\left(\frac{\beta}{\alpha\lambda}\right)\left[1-\left(\frac{\beta}{\lambda+\beta}\right)^\alpha\right] = 1-0\times 1 = 1$$

For $n>1$, as λ approaches ∞, $\Pr[N=n]$ should approach 0. From (19), we have

$$\lim_{\lambda\to\infty}\Pr[N=n] = \lim_{\lambda\to\infty}\left(\frac{\beta}{\alpha\lambda}\right)\left[1-\left(\frac{\beta}{\lambda+\beta}\right)^\alpha\right]^2\left(\frac{\beta}{\lambda+\beta}\right)^{\alpha(n-2)} = 0\times(1-1)\times 1 = 0$$

As λ approaches 0, $\Pr[N=1]$ should approach 0. In (19), the limit of $\left(\frac{\beta}{\alpha\lambda}\right)\left[1-\left(\frac{\beta}{\lambda+\beta}\right)^\alpha\right]$ has the $\frac{0}{0}$ form. By using the L'Hôpital's rule

$$\lim_{\lambda\to 0}\left(\frac{\beta}{\alpha\lambda}\right)\left[1-\left(\frac{\beta}{\lambda+\beta}\right)^\alpha\right] = \lim_{\lambda\to 0}\left(\frac{\beta}{\alpha}\right)\left\{\frac{\left\{\frac{d\left[1-\left(\frac{\beta}{\lambda+\beta}\right)^\alpha\right]}{d\lambda}\right\}}{\left[\frac{d(\lambda)}{d\lambda}\right]}\right\}$$

$$= \lim_{\lambda\to 0}\left(\frac{\beta}{\alpha}\right)\left[\frac{\left(\frac{\alpha}{\beta}\right)\left(\frac{\beta}{\lambda+\beta}\right)^{\alpha+1}}{1}\right] = 1 \tag{20}$$

Substitute (20) into (19) for N=1, we have

$$\lim_{\lambda\to 0}\Pr[N=1] = \lim_{\lambda\to 0} 1-\left(\frac{\beta}{\alpha\lambda}\right)\left[1-\left(\frac{\beta}{\lambda+\beta}\right)^\alpha\right] = 1-1 = 0$$

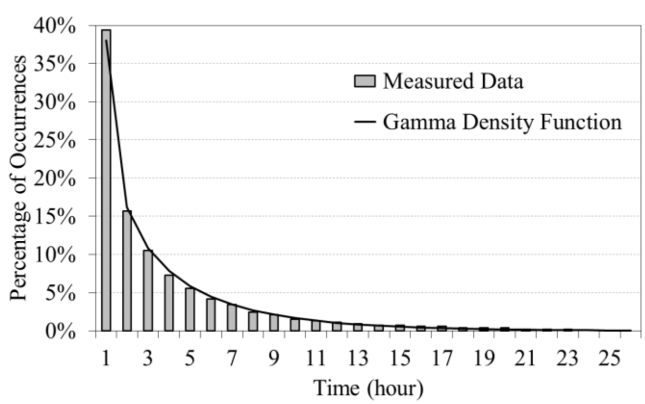

Fig. 15. Histogram of t_i and the Gamma density function.

Fig. 15 plots the histogram of t_i (in percentage of occurrences) for the visitor parking lot of the administration building at NCTU. Most visitors spend roughly 1-4 hours in this building for business. The histogram has the mean $\mathrm{E}[t_i]$ = 3.1874 hours and the variance $\mathrm{V}[t_i]$ = 16.6857 hour^2= 1.64237$\mathrm{E}[t_i]^2$ We approximate the histogram by the Gamma distribution with the shape parameter α =0.60886 and the scale parameter $\beta = 0.19102$. Fig. 15 shows that the Gamma distribution nicely fits the histogram.

The analytic model (i.e., Equation (19)) is used to validate an event-driven simulation we developed in Appendix A. The analytic model and the simulation experiments are compared with various parameter setups. For all cases we considered, the discrepancies are less than 0.4%.

Then we use the simulation experiments and the measured data in Fig. 15 to investigate the relationship between T_p and $\Pr[N=n]$. To save the energy consumption of the NB-IoT devices, a long T_p should be selected. On the other hand, a short T_p should be selected to ensure a small amount of lost sensor data (i.e., a large $\Pr[N=1]$ and a small $\mathrm{E}[N]$). It is clear that T_p, $\Pr[N=1]$ and $\mathrm{E}[N]$ are conflicting output measures.

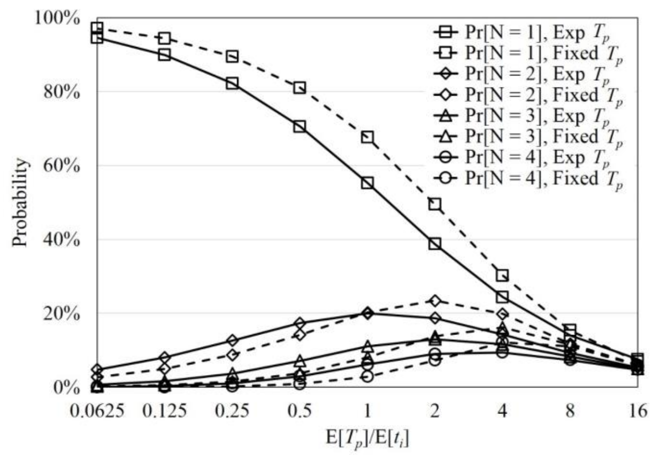

Fig. 16. $\Pr[N=n]$ against $\mathrm{E}[T_p]/\mathrm{E}[t_i]$.

Fig. 16 illustrates the probabilities $\Pr[N=n]$ against $\mathrm{E}[T_p]/\mathrm{E}[t_i]$ (for $1 \le N \le 4$**).** The figure shows that the probability $\Pr[N=1]$ (i.e., the parking status is up to date when connection failure is detected) decreases as $\mathrm{E}[T_p]$ increases. The curve indicates that by decreasing $\mathrm{E}[T_p]$ from $\mathrm{E}[t_i]$ to $0.25\mathrm{E}[t_i]$, the $\Pr[N=1]$ performance is improved by 48.77% for Exponential $\mathrm{E}[T_p]$, and 32.35% for fixed $\mathrm{E}[T_p]$. On the other hand, by decreasing $\mathrm{E}[T_p]$ from $0.25\mathrm{E}[t_i]$ to 0.0625 $\mathrm{E}[t_i]$, the $\Pr[N=1]$ performance is improved by 15% for Exponential $\mathrm{E}[T_p]$, and 8.5% for fixed $\mathrm{E}[T_p]$. In other words, for $\mathrm{E}[T_p] \ge 0.25\mathrm{E}[t_i]$, $\Pr[N=1]$ performance is effectively improved as $\mathrm{E}[T_p]$ decreases. For $\mathrm{E}[T_p] \le 0.25\mathrm{E}[t_i]$, $\Pr[N=1]$ performance is insignificantly improved as $\mathrm{E}[T_p]$ decreases. Therefore, it is appropriate to select a small $\mathrm{E}[T_p] \ge 0.25\mathrm{E}[t_i]$.

For $n \ge 2$, $\Pr[N=n]$ increases and then decreases as $\mathrm{E}[T_p]$ increases. As N increases, the peaks of the curves shift to the right. The $\Pr[N=1]$ values for fixed $\mathrm{E}[T_p]$ is larger than that for Exponential $\mathrm{E}[T_p]$. For $n \ge 2$, When $\mathrm{E}[T_p]$ is small

$\Pr[N = n]$ for fixed $\mathrm{E}[T_p]$ is smaller than that for Exponential $\mathrm{E}[T_p]$. When $\mathrm{E}[T_p]$ is large $\Pr[N = n]$ for fixed $\mathrm{E}[T_p]$ is larger than that for Exponential $\mathrm{E}[T_p]$.

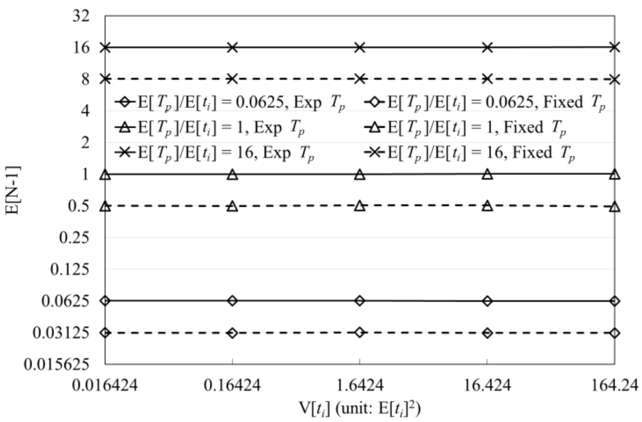

Fig. 17. $\mathrm{E}[N-1]$ against $\mathrm{E}[T_p]/\mathrm{E}[t_i]$ and $\mathrm{V}[t_i]$.

Fig. 17 shows the effect of the variance $\mathrm{V}[t_i]$ on the expected number $\mathrm{E}[N-1]$ of missed status changes before the connection failure is detected. The figure indicates that the $\mathrm{E}[N-1]$ performance for fixed $\mathrm{E}[T_p]$ is better than that for Exponential $\mathrm{E}[T_p]$, which is not affected by the variance $\mathrm{V}[t_i]$.

Fig. 18 illustrates the probability $\Pr[N = 1]$ against the variance $\mathrm{V}[t_i]$ and $\mathrm{E}[T_p]/\mathrm{E}[t_i]$. This figure shows a non-trivial result where $\Pr[N = 1]$ is an increasing function of $\mathrm{V}[t_i]$. That is, for a larger $\mathrm{V}[t_i]$, it is more likely that no status change information is lost when the network failure is detected. This phenomenon is explained as follows. When $\mathrm{V}[t_i]$ is large, more long t_i intervals are observed, and it is more likely that $T_p^* < t_1^*$, and therefore a large $\mathrm{E}[N = 1]$ is expected.

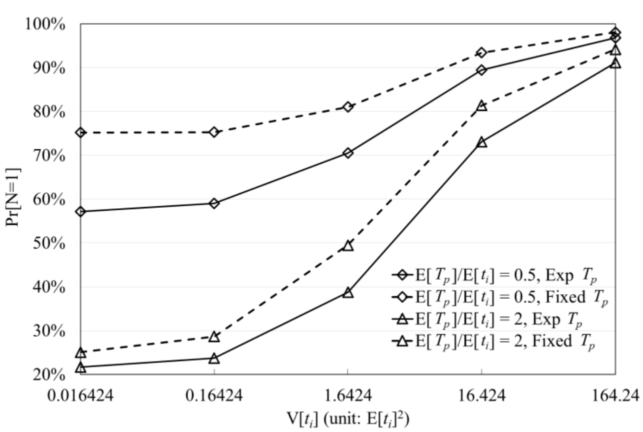

Fig. 18. $\Pr[N = 1]$ against $\mathrm{V}[t_i]$ and $\mathrm{E}[T_p]/\mathrm{E}[t_i]$.

VI. Related Work

This section compares NB-IoTtalk with the related studies. The work [21] proposed an architecture-driven approach based on model-driven architecture (MDA) to enable automatic generation of a Wireless Sensor and Actuator Network middleware tailored to the requirements elucidated by the system architects. Based on a design flow, [22] takes the application software, the hardware specification (communication protocols and sensor network platforms) and the mapping between them as inputs to construct a system model in Behavior Interaction Priority (BIP) framework. Both MDA and BIP provide good paradigms for developing IoT applications, and the details can be found in [21] and [22]. Similar to these two approaches, IoTtalk automatically generates the network applications. The major differences are that IoTtalk develops the whole system centered at the device feature and the device model concepts**,** and a network application is automatically generated to interact between the IDFs and the ODFs. Furthermore, a user friendly GUI is included in IoTtalk that allows the user to modify the network applications without any or with little programming effort. Although NB-IoTtalk enjoys the IoTtalk features mentioned above, it is not incremental improvement of IoTtalk. NB-IoTtalk makes the following major contributions not found in IoTtalk [10][11]:

- Automatic generation of Device Application: In IoTtalk, the creation of DAs for IoT devices are the responsibility of users. In this paper, the DAs for various NB-IoT applications (Fig. 2 (5)) are automatically created.
- Automatic parsing the message payloads for IDFs and ODFs: Different NB-IoT applications send different types of IoT data to the network. Since the IoT data are stored in the payloads of NB-IoT messages in the JSON format (Fig. 4), we can automatically parse the DF values and create the corresponding icons in the IoTtalk GUI (Fig. 3).
- Extending IoTtalk with the tag concept that allows the DF attributes to associate with the tags: Without tags, the previous approaches need to manually create various DFs with the same attribute. With the ID tag, we can handle multiple devices by one DA, and represent all of them by one icon in the IoTtalk GUI. We can also group the NB-IoT devices into subgroups (Fig. 5). With the GeoData tag, multiple NB-IoT devices can be easily shown in the map output device (Fig. 12) without extra programming effort.
- Proposing service platform interworking by developing a DA to bridge other platforms to IoTtalk: We described how NB-IoT platform interworks with IoTtalk (Fig. 2).
- Original modeling of event-triggered NB-IoT message delivery: Most TTL mechanisms [23] were investigated to balance data accuracy against the transmission cost between the servers and the clients. In Apache and Squid, a TTL interval is defined for data entries stored in the mobile devices. The TTL for a data entry is determined based on whether the data entry is modified due to either a mobile query or a server update, which involves the data access times and the TTL expiration time. On the other hand, in event-triggered NB-IoT message delivery, we need to detect if a message does not arrive at the server; i.e., we also need to consider the sensor failure (unavailable) times. Therefore, the analysis in this paper is more complicated than that in [23].

VII. Conclusions

Many outdoor IoT applications involve large numbers of homogeneous NB-IoT devices, and it is tedious to specify and accommodate these devices in application development. This paper proposed the NB-IoTtalk service platform for fast development of NB-IoT applications, which utilizes a tag mechanism to resolve this issue. In our approach, every sensor and every actuator in an IoT application can be associated with a tag. We define five types of tags: Identity (ID), Geographic Data (GeoData), Time (T), Battery (B), and Privacy (P). With this tag mechanism, NB-IoTtalk provides an easy-to-manipulate GUI to accommodate a large number of NB-IoT devices in an application with the ID tag, and illustrate them to a visual map using the GeoData tag. Our approach automatically creates and parses the device profile used to interpret the payload of an NB-IoT message.

We then used a smart parking lot application as an example to show event-triggered reporting in NB-IoT. We conducted the on/off status measurements of the parking sensors in a NCTU parking lot, and developed an analytic model and simulation experiments to investigate the performance of event-triggered reporting in terms of the time-to-live (TTL) report frequency and the outage detection accuracy for the parking lot application. Our study provides the guidelines to set the TTL interval for NB-IoT event-triggered reporting. Specifically, for the NCTU parking lot, we suggest to selected fixed TTL interval $T_p = 0.25\text{E}[t_i]$, where t_i is the interval between two status changes of the parking sensor.

In the future, we will extend multiple tags to associate with a DF, and in particular, employ the Privacy tag to fit privacy regulations of various countries.

APPENDIX A. THE SIMULATION MODEL

We have developed an event-driven simulation to compute $\Pr[N = n]$. Several variables are defined. The event-driven simulation is repeated with 10^6 replications. Let *I* be the number of replications performed so far. The simulation uses a Boolean variable *f* as a flag to indicate if the network is disconnected. An array $N[I]$ is used to store the number of lost status changes before the failure is detected in the *I*-th replication. An event *e* consists of two fields: the timestamp *e*.t when the event occurs and the event type *e*.type. Three event types are defined:

- **Disconnection**: the network is disconnected
- **StatusChange**: the status of the parking sensor changes
- **TTLreport**: the parking sensor issues a TTL report

The flowchart of the simulation is illustrated in Fig. A.1 with the following steps:

Step A.1. The variables *I* and *N*[] are initialized to 0.

Step A.2. The first **Disconnection** event *e1* is created where its timestamp *e1*.t drawn from the Exponential distribution is generated by an Exponential random number generator RNG1(). Therefore, this network disconnection interval is a random observer of the **StatusChange** and the **TTLreport** time intervals. The variable *f* is set to false, and the number *I* of replications is incremented by one. This event is inserted in the event list.

Step A.3. The first **StatusChange** event *e2* is created. The timestamp *e2*.t is drawn from the Gamma distribution (with the shape parameter α and the scale parameter $1/\beta$) produced by a Gamma random number generator RNG2(). This event is inserted in the event list.

Step A.4. The first **TTLreport** event *e3* is created where its timestamp is a fixed value or is drawn from the Erlang-K distribution generated by a random number generator RNG3(). The event is inserted in the event list.

Step A.5. The event *e* with the smallest timestamp is removed from the event list.

Step A.6. The event type *e*.type is checked. If *e*.type is **Disconnection**, **Step A.7** is executed. If *e*.type is **StatusChange**, **Step A.8** is executed. If *e*.type is **TTLreport**, **Step A.11** is executed.

Step A.7 (*e*.type is **Disconnection**)**.** Flag *f* is set to true. The simulation flow goes to **Step A.5**.

Step A.8 (*e*.type is **StatusChange**)**.** The next **StatusChange** event *e2* is created where its timestamp is set to *e2*.t=*e*.t+ RNG2(). The event *e2* is inserted in the event list.

Step A.9. Flag *f* is checked. If *f* is true (i.e., the network does not receive this status change), the simulation flow goes to **Step A.10**. Otherwise, the flow goes to **Step A.5.**

Step A.10. $N[I]$ is incremented by one. The simulation flow goes to **Step A.5**.

Step A.11 (*e*.type is **TTLreport**)**.** Flag *f* is checked. If *f* is true (i.e., the network detects that it does not receive the TTL report), the simulation flow goes to **Step A.13**. Otherwise, the flow goes to **Step A.12.**

Step A.12. The next **TTLreport** event *e3* is created where its timestamp is set to *e3*.t=*e*.t+ RNG3(). The event *e3* is inserted in the event list. The simulation flow goes to **Step A.5**.

Step A.13. If $I > 10^6$, then report $N[I]$ at **Step A.14** and the simulation terminates. Otherwise, the flow goes to **Step A.2** for the next replicated run.

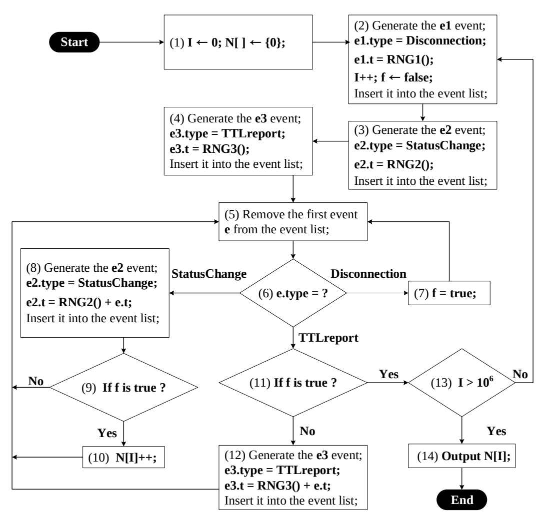

Fig. A.1. Simulation flowchart.

REFERENCES

[1] H. W. Kao, Y. H. Ju and M. H. Tsai, "Two-Stage Radio Access for Group-Based Machine Type Communication in LTE-A", *IEEE*

International Conference on Communications (ICC), pp. 3825-3830, London, June 2015.
[2] X. Li *et al.*, "Smart Community: an Internet of Things Application", *IEEE Communications Magazine*, vol. 49, no. 11, pp. 68-75, 2011.
[3] Y. Y. Shih, A. C. Pang, and P. C. Hsiu, "A Storage-free Data Parasitizing Scheme for Wireless Body Area Networks", *IFIP Networking*, Trondheim, Norway, 2014.
[4] Y.-B. Lin, and A.-C. Pang, "Wireless and Mobile All-IP Networks", John Wiley and Sons, 2005.
[5] J. Xie and I. Howitt, "Multi-Domain WLAN Load Balancing in WLAN/WPAN Interference Environments", *IEEE Transactions on Wireless Communications*, vol. 8, no. 9, pp. 4884-4894, September 2009.
[6] N. Sornin *et al.*, "LoRaWAN™ Specification", LoRA Alliance, 2015.
[7] 3GPP TR 45.820, "Cellular system support for ultra-low complexity and low throughput Internet of Things (CIoT)", v13.1.0, December 2015.
[8] 3GPP TR 23.720, "Study on architecture enhancements for Cellular Internet of Things", v13.0.0, March 2016.
[9] P. Andres-Maldonado *et al.*, "NarrowBand IoT Data Transmission Procedures for Massive Machine Type Communications", *IEEE Network*, pp. 8-15, November/December, 2017.
[10] Y.-B. Lin *et al.*, "EasyConnect: A Management System for IoT Devices and Its Applications for Interactive Design and Art", *IEEE Internet of Things Journal*, vol. 2, no. 6, pp. 551-561, December 2015.
[11] Y.-B. Lin *et al.*, "IoTtalk: A Management Platform for Reconfigurable Sensor Devices", *IEEE Internet of Things Journal*, vol. 4, no.5, pp. 1552-1562, October, 2017.
[12] Y.-W. Lin *et al.*, "IoTtalk-RC: Sensors As Universal Remote Control for Aftermarket Home Appliances", *IEEE Internet of Things Journal*, vol. 4, no. 4, pp. 1104-1112, Aug. 2017
[13] Y.-W. Lin *et al.*, "ArduTalk: An Arduino Network Application Development Platform based on IoTtalk", *IEEE System Journal*, accepted and to appear, 2018.
[14] S.-Y. Wang *et al.*, "Performance of LoRa-based IoT Applications on Campus", *IEEE 86th Vehicular Technology Conference (VTC-Fall)*, Toronto, pp. 1-6, September 2017.
[15] Y.-C. Sung *et al.*, "Voice/Video Quality Measurement for LTE Services", *IEEE Wireless Communications*, accepted and to appear, 2018.
[16] 3GPP TR 36.888, "Study on provision of low-cost Machine-Type Communications (MTC) User Equipments (UEs) based on LTE", v12.0.0, June 2013.
[17] Y.-B. Lin *et al.*, "Location-based IoT applications on campus: The IoTtalk approach", *Pervasive and Mobile Computing*, pp. 660-673, vol. 40, June 2017.
[18] S.-R. Yang, S.-Y. Yan, and H.-N. Hung, "Modeling UMTS Power Saving with Bursty Packet Data Traffic", *IEEE Transactions on Mobile Computing*, vol. 6, no. 12, pp. 1398-1409, December 2007.
[19] S.-R. Yang, P. Lin, and P.-T. Huang, "Modeling Power Saving for GAN and UMTS Interworking", *IEEE Transactions on Wireless Communications*, vol. 7, no. 12, pp. 5326-5335, December 2008.
[20] S.-R. Yang and W.-T. Chen, "SIP Multicast-based Mobile Quality-of-service Support over Heterogeneous IP Multimedia Subsystems", *IEEE Transactions on Mobile Computing*, vol. 7, no. 11, pp. 1297-1310, November 2008.
[21] T. C. Rodrigues *et al.*, "Architecture-driven development approach WASN for applications", *IEEE 13th International Conference on Embedded and Ubiquitous Computing*, pp. 68-75, October 2015.
[22] A. Lekidis *et al.*, "DesignFlow for the Rapid Development of Distributed Sensor Network Applications", *Verimag Research Report*, Tech. Rep. TR-2014-13, 2014.
[23] Y. Fang *et al.*, "TTL Prediction Schemes and the Effects of Inter-Update Time Distribution on Wireless Data Access", *ACM Wireless Networks*, vol. 10, no. 5, pp. 607-619, September, 2004.

Yi-Bing Lin (M'96–SM'96–F'03) is the Vice Chancellor of the University System of Taiwan, and the Lifetime Chair Professor of the National Chiao Tung University (NCTU). During 2014–2016, he was Deputy Minister of the Ministry of Science and Technology, Taiwan. He is also an Adjunct Research Fellow of the Institute of Information Science, and Center for Information Technology Innovation, Academia Sinica, and a member of board of directors, Chunghwa Telecom. He is the Author of the books *Wireless and Mobile Network Architecture* (Wiley, 2001), *Wireless and Mobile All-IP Networks* (Wiley, 2005), and *Charging for Mobile All-IP Telecommunications* (Wiley, 2008). He received the TWAS Prize in Engineering Sciences in 2011 (The World Academy of Sciences). He serves as the Chair of IEEE Taipei Section. He is an AAAS Fellow, an ACM Fellow, and an IET Fellow.

Hung-Chun Tseng received the B.S. degree in Department of Applied Science from National Hsinchu University of Education, Taiwan, in 2007, and the M.S. degrees in Department of Electrophysics from National Chiao Tung University (NCTU), Taiwan, in 2011. He is currently pursuing the Ph.D. degree at the Department of Computer Science, NCTU.

Yun-Wei Lin received the B.S. degree in computer and information science from Aletheia University, Taipei, Taiwan, in 2003, and the M.S. and Ph.D. degrees in computer science and information engi-neering from National Chung Cheng University, Chiayi, Taiwan, in 2005 and 2011, respectively.

He is currently an Assistant Research Fellow with National Chiao Tung University, Hsinchu, Taiwan. His current research interests include mobile ad hoc networks, wireless sensor networks, vehicular ad hoc networks, and IoT/M2M communications.

Ling-Jyh Chen received the BEd degree in information and computer education from National Taiwan Normal University in 1998, and the MS and PhD degrees in computer science from the University of California, Los Angeles, in 2002 and 2005, respectively. He joined the Institute of Information Science of Academia Sinica as an assistant research fellow in 2005, and became an associate research fellow in 2011. His research interests are networked sensing systems, network measurements, and mobile data management. He is a senior member of the IEEE.