Lightweight Management of Resource-Constrained Sensor Devices in Internet of Things

Zhengguo Sheng, *Member, IEEE*, Hao Wang, Changchuan Yin, *Senior Member, IEEE*, Xiping Hu, *Graduate Student Member, IEEE*, Shusen Yang, *Member, IEEE*, and Victor C. M. Leung, *Fellow, IEEE*

***Abstract*—It is predicted that billions of intelligent devices and networks, such as wireless sensor networks (WSNs), will not be isolated but connected and integrated with computer networks in future Internet of Things (IoT). In order to well maintain those sensor devices, it is often necessary to evolve devices to function correctly by allowing device management (DM) entities to remotely monitor and control devices without consuming significant resources. In this paper, we propose a lightweight RESTful Web service (WS) approach to enable device management of wireless sensor devices. Specifically, motivated by the recent development of IPv6-based open standards for accessing wireless resource-constrained networks, we consider to implement IPv6 over low-power wireless personal area network (6LoWPAN)/routing protocol for low power and lossy network (RPL)/constrained application protocol (CoAP) protocols on sensor devices and propose a CoAP-based DM solution to allow easy access and management of IPv6 sensor devices. By developing a prototype cloud system, we successfully demonstrate the proposed solution in efficient and effective management of wireless sensor devices.**

***Index Terms*—Constrained application protocol (CoAP), device management (DM), Internet of Things (IoT), IPv6, wireless sensor networks (WSNs).**

I. Introduction

THE CONCEPT of Internet of Things (IoT) can be traced back to the pioneering work done by Kevin Ashton in 1999 and it is initially linked to the new idea of using RF identification (RFID) in supply chain [1]. Since recently, this term became popular and is well known as a new communication system where the Internet is connected to the physical world via ubiquitous wireless sensor networks (WSNs). Generally, sensing devices are with common features of constrained energy resources, limited processing capability, vulnerable radio conditions, real-time nature of applications, and no direct human interaction, etc. By interconnecting sensor devices using low-cost wireless communication technologies, which is usually named as WSN, a new ecosystem with a large number of smart applications has been formed.

With the development of IoT technologies in the past few years, a number of major standardization alliances are gradually formed based on their interests in technology selections and commercial markets. Technically speaking, current IoT solutions can be categorized as non-Internet protocols (IP)-based and IP-based solutions. Most of off-the-shelf solutions belong to the former, especially for some well-known standard alliances, such as ZigBee [2] and WAVE2M [3] for office and manufacturing automation, and WirelessHart [4] and PROFIBUS [5] for real-time industrial control systems, etc. However, most of these non-IP solutions are isolated within their own verticals, which hinders the IoT development due to the incompatible nature across heterogeneous communication systems.

Motivated by the fact that the transmission control protocol (TCP)/IP protocol is the *de facto* standard for computer communications in today's networked world, IP-based solution could be the future for IoT networks, etc. IP smart object alliance (IPSO) [6] actively promotes IPv6-embedded devices for machine-to-machine (M2M) applications. PROFINET, a promising real-time Ethernet standard, also adapts Ethernet to the next generation of industrial automation [7]. In order to tackle the technical challenges, such as extensive protocol overheads against memory and computational limitations of sensor devices, Internet Engineering Task Force (IETF)[1] takes the lead to standardize communication protocols for resource constrained devices and develop a number of IP, including IPv6 over low-power wireless personal area networks (6LoWPAN) [8], routing protocol for low power and lossy network (RPL) [9], and constrained application protocol (CoAP) [10], etc.

Although a wide range of intelligent and tiny sensing devices have been massively deployed in a variety of application environments, many open challenges remain, which are mostly due to the complex deployment characteristics of such systems and the stringent requirements imposed by various services wishing to make use of such complex systems [11]. In order to well

Manuscript received October 25, 2014; revised February 23, 2015; accepted March 30, 2015. Date of publication April 06, 2015; date of current version September 15, 2015. This work was supported in part by the University of Sussex Start-up Fund, in part by the Canadian Natural Sciences and Engineering Research (NSERC), in part by the NSERC DIVA Strategic Research Network, in part by the NSFC under Grant 61271257, and in part by the China 973 Program under Grant 2012CB315801.

Z. Sheng is with the School of Engineering and Informatics, University of Sussex, Sussex BN1 9RH, U.K., and also with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC V6T 1Z4, Canada (e-mail: z.sheng@sussex.ac.uk).

H. Wang is with Orange International Labs, Beijing 100190, China (e-mail: hao.wang@orange.com).

C. Yin is with the Beijing University of Posts and Communications, Beijing 100876, China (e-mail: ccyin@bupt.edu.cn).

X. Hu and V. Leung are with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC V6T 1Z4, Canada (e-mail: xipingh@ece.ubc.ca; vleung@ece.ubc.ca).

S. Yang is with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3BX, U.K. (e-mail: shusen.yang@liverpool.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JIOT.2015.2419740

[1]The IETF is a large open international community of network designers, operators, vendors, and researchers concerned with the developments and promotions of Internet standards of the Internet protocol suite (TCP/IP).

maintain a large scale of WSNs, e.g., dynamic registration of sensor devices or monitoring sensor performance, IoT device management (DM) authorities should be able to provide a reliable and efficient way to remotely monitor and control sensor devices without consuming significant resources. This also provides a motivation to recent global IoT/M2M-related standardizations. In particular, the oneM2M global initiative [12], [13] has been formed to develop one globally agreed specifications for common service layer, which can be the basis of horizontal management platform. The IoT-A project [14], which is a European research project addressing the reference model of IoT, is to develop IoT architectures in an interoperable manner. The project has derived entities and resources, which are subject for management functions, and provides various functions to orchestrate and manage collaboration of IoT devices.

DM is an integration of network, system, and application managements. In essence, it includes provisioning and management, configuration of network parameters, firmware upgrades, and performance monitoring, etc. As a result, a number of DM standards are emerging, such as TR-069 [15] for automatic configuration of set-top box, ISO/IEC 14543-3 [16] for home and building automation (HBA), and field device integration (FDI) [17] for the unified management for all field devices, etc. However, none of these standards can be directly used on IoT devices, especially on a processing and battery-restricted sensor device.

We are looking at open technologies to tackle device management of WSN, and the IPv6-based solution is a promising one. Motivated by the fact that the CoAP is an application layer protocol which is intended for use in resource-constrained Internet devices and the simple network management protocol (SNMP), we propose a framework of CoAP-based DM solution for WSN, and develop device management-oriented functions, resource identities, and protocol, etc. Specifically, we take an approach to extend the representation state transfer (REST) paradigm [18], in which a lightweight Web server can be embedded in resource-constrained sensor devices, and map DM functions into CoAP methods. In essence, the proposed method not only integrates IoT into the network, but also manages them via the "Web."

The following summarizes our contributions and key results.

1) We implement the IPv6 protocol stack on sensor testbed and integrate the border router of WSN into an open-platform gateway, as well as implement the HTTP-CoAP proxy implementation to the OpenWrt to realize remote access from an ordinary IP terminal to IPv6 sensor devices.
2) We propose a framework of efficient device management solution for WSN based on IETF open standard CoAP and develop DM-oriented functions, naming and addressing for resource identities as well as mappings of CoAP methods to management functions. The performance evaluation shows that the proposed solution can significantly reduce both packet length and loss rate by 74% and 18.75%, respectively.

This paper is organized as follows. Related works are provided in Section II. The RESTful protocol stack is introduced in Section III. The proposed DM solution is discussed in Section IV. The prototype system is presented in Section V and performance evaluation results are shown in Section VI. Finally, concluding remarks and future work are given in Section VII.

II. Related Work

Recent technology trends in the Web services (WS) are primarily separated as Big WS (or WS-*) and RESTful WS. In the latest work of WS in WSN, Kyusakov *et al.* in [19] take an approach to deploy interoperable simple object access protocol (SOAP)[2]-based WS directly on nodes. Moreover, the WS methods are also widely applied in Automation industry. Cucinotta *et al.* in [20] propose a service-oriented architecture (SOA) with enhanced real-time capabilities by allowing for negotiation of the quality-of-service (QoS) requested by clients from WS for industrial automation. However, the above literatures prefer the WS-* architecture, which may bring extensive overheads for resource-constrained devices. Pautasso *et al.* in [21] compare these two architecture choices and argue that the RESTful WS can create a loosely coupled system which is better suited for simple and flexible integration scenarios, whereas WS-* can provide more advanced QoS for enterprise-level usages.

More recent works are dedicated for developing REST-style IoT systems to enable easy access from application servers to wireless sensor devices [22]–[25]. REST, a lightweight WS implementation, is a general design style of Internet resource access protocol. It provides a design concept that all the objects in the Internet are abstracted as resources. REST style can make applications as sharable, reusable, and loose coupling services. Although its simplicity, most of the existing solutions are not IP based, which means that a multiprotocol translation gateway is needed. As discussed in [26], the network protocol translation can bring more complexity than just a packet format conversion, which usually involves semantics translation between different mechanisms and logic for routing, QoS and security [27], etc.

There are recent papers focusing on the implementation of IPv6 protocol stack on various system platforms. Shelby in [28] gives an overview of the Web architecture and introduces the new IETF Constrained RESTful Environments (CoRE) standardization activity. Potsch *et al.* in [29] demonstrate an intelligent container testbed where the CoAP protocol is implemented on the embedded operating system TinyOS.[3] Moreover, a couple of other implementations of CoAP are also available on the Contiki[4] platform [30]–[32]. There are also papers on the protocol mappings from proprietary solutions or SOAP to CoAP, e.g., [33], [34]. However, most of these papers are either considering protocol translation to CoAP or for the purpose of connectivity evaluations on different operation platforms by assuming a virtual gateway.

Although considerable research has been done on different aspects of sensor networks, the management issue for sensor

[2]SOAP is a protocol specification for exchanging structured information in the implementation of WS in computer networks.

[3]TinyOS is an open source software component-based operating system and platform targeting WSN.

[4]Contiki is an open source operating system for the IoT. Contiki allows tiny battery-operated low-power systems communicate with the Internet.

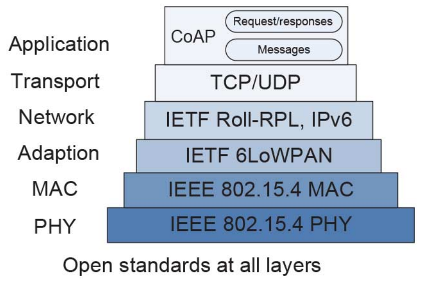


Fig. 1. Overview of IPv6 protocol stack ranging from physical layer up to application layer.

devices is still little explored. Frye and Cheng in [35] propose to incorporate ontology into management of *ad hoc* networks based on existing network management protocol, such as SNMP. Yang *et al.* in [36] propose a basic concept of device management scheme based on IEEE 1451 standard. Different from the previous methods, Hergenroeder *et al.* in [37] propose a hard way method by developing an extra hardware to support energy management. However, those solutions are all implemented using independent management protocols or hardware which are lack of reuse with parallel communication protocols, e.g., CoAP, and may bring extract complexity. Although there are recent works considering to introduce management into CoAP server, i.e., [38] proposes dedicated application protocol on top of CoAP to map all application functions in building automation, [39] utilizes synchronization markup language (SyncML) protocol onto CoAP for DM, and [40] proposes the latest integration of CoAP with SNMP, they all either build management capabilities on top of CoAP or need to support multiple protocols simultaneously, which may bring extra overhead for resource-constrained devices.

Different from the above literatures, our contribution in this paper is to develop an efficient and lightweight device management solution by extending the CoAP protocol without consuming extra resources. To the best of our knowledge, this is the first work that considers DM via CoAP protocol and realizes ease of access and management of WSN.

III. Implemented Protocol Stack for WSN

In this section, we introduce the IETF protocol stack used in resource-constrained networks [41]. Fig. 1 illustrates the protocol stack ranging from physical layer up to application layer.

A. *IEEE 802.15.4*

IEEE 802.15.4 [42] is a radio technology standard for low power and low data rate applications with a radio coverage of only a few meters. It has typically a maximum data rate of 250 kb/s and a maximum output power of 1 mW. The maximum packet size is 127 bytes. Besides, in order to achieve energy savings, radio power management (e.g., duty cycling) is an essential part in media access control (MAC) layer mechanisms. The radio transceiver must be managed so that it can be switched OFF when there is no traffic but switched ON when communication is engaged.

B. *6LoWPAN*

The main focus of 6LoWPAN is on protocol optimization of IPv6 over networks using IEEE 802.15.4. In fact, there are two reasons to apply 6LoWPAN over IEEE 802.15.4. On one hand, consider that the maximum frame size supported by IEEE 802.15.4 is only 127 bytes and significant header space may be taken by other layered protocols (e.g., MAC layer header, IPv6 header, security header, and transport layer), the payload size available for the application layer is very limited. On the other hand, since the minimum value of maximum transmission unit (MTU) specified by IPv6 is 1280 bytes which is larger than the supported size of IEEE 802.15.4, an adaptation layer right above the data link layer to segment the IPv6 packet into small pieces is required by the lower layer.

C. *RPL*

RPL is a distance-vector routing protocol, in which nodes construct a destination-oriented acyclic graph (DODAG) by exchanging distance vectors and root information with a "controller." Through broadcasting routing constraints, the root node (i.e., central control point) filters out nodes that do not meet the constraints and selects the optimum path according to the metrics. In a stable state, each sensor node will have a set of "parents" and will forward packets along its parents to the "root."

D. *CoAP*

CoAP is a specialized Web transfer protocol for resource-constrained nodes and networks. CoAP conforms to the REST style. It abstracts all objects in the network as resources. Each resource corresponds to a unique universal resource identifier (URI) from which the resources can be operated stateless, including GET, PUT, POST, DELETE, and so on. The URI is described as a link in the CoRE link format [43]. The CoRE link format is carried as a payload and is assigned an Internet media type.

Unlike HTTP, CoAP adopts datagram-oriented transport protocols, such as user datagram protocol (UDP). In order to ensure reliable transmission, CoAP introduces a two-layer structure as shown in Fig. 1: the messaging layer is used to deal with asynchronous interactions with UDP, such as confirmable (CON), non-CON (NON), acknowledgment (ACK), and reset (RST) messages, whereas the request/response interaction layer is used to transmit resource operation requests and the request/response data.

IV. CoAP-Based DM for WSN

In this section, we propose a framework of efficient device management solution for WSN based on IETF open standard CoAP and develop DM-oriented functions, resource identities, and protocol, etc.

A. *Management Functions*

The system architecture in Fig. 2 shows the interaction between a IoT client (e.g., via cloud platform) and a

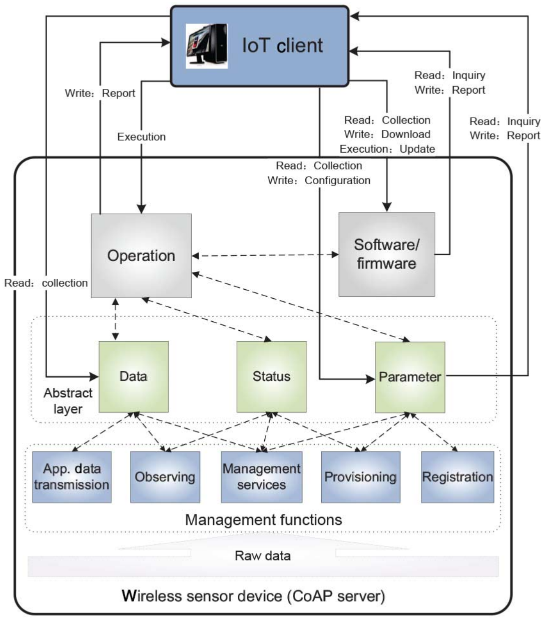


Fig. 2. System architecture of IoT DM, including management functions, abstract layer, and interconnections with client.

sensor device, especially the device components interfaces to IoT client. Due to the requirements imposed to IoT services, such as no direct human interaction, reliable remote management/control and scalable features of applications, we propose five major DM functions which are applicable to WSN.

1) *Registration*: It is a primary function to allow a sensor device to register/deregister with a remote management server, maintain, and update registration information.
2) *Provisioning*: It is to initialize and synchronize essential information of a sensor device with a remote management server.
3) *Management services*: Once the sensor device is well connected with a remote server, a number of essential management services should take in charge to maintain IoT services, such as parameter configuration, connection diagnose, status inquiry, and remote control.
4) *Observing*: It is the unique feature of CoAP to allow sensor devices to "observe" resources, i.e., to retrieve a representation of a resource and keep this representation updated by the remote server over a period of time.
5) *Application data transmission*: It includes any application data that can be collected and delivered to IoT clients, or any other application protocols above CoAP.

Although the management functions can be defined in different manners, they all share common resources on one sensor device and we abstract these resources as parameters (e.g., hardware and software information), status (e.g., dynamic information for connection and faulty diagnose), and data (e.g., information collected from environment), which are defined as abstract layer in Fig. 2. The interactions with IoT client can be directly triggered with these resources via GET, PUT, POST, and DELETE methods provided by CoAP.

TABLE I
NAMING AND ADDRESSING ASSIGNMENT

Class		Resource		Subresource		Method	
Name	ID	Name	ID	Name	ID	Name	ID
Hardware information							(Hex→bit)
HW	1	CHIP_ID	1	Null	0	Get	8→1000
HW	1	SRAM	2	Null	0	Get	8→1000
HW	1	FLASH	3	Null	0	Get	8→1000
HW	1	MAC_addr	4	Null	0	Get	8→1000
HW	1	RADIO	5	Frequency	1	Get	8→1000
HW	1	RADIO	5	Channel	2	Get	8→1000
HW	1	USART0	6	Baudrate	0	Get	8→1000
Operating system information							
SYS	2	CONTIKI_v	1	Null	0	Get	8→1000
SYS	2	NAME	2	Null	0	Get/Put	c→1100
Network protocol information							
NET	3	PANID	1	Null	0	Get	8→1000
NET	3	RDC	2	Null	0	Get	8→1000
NET	3	MAC	3	Null	0	Get	8→1000
NET	3	NETWORK	4	Null	0	Get	8→1000
NET	3	IPv6	5	Prefix	1	Get	8→1000
NET	3	CoAP_v	6	Null	0	Get	8→1000
Onboard resources information							
RES	4	ACTUATOR	1	LEDs	1	Get/Put /Post	e→1110
RES	4	SCREEN	2	Null	0	Get/Put /Post	e→1110
RES	4	SENSOR	3	Humidity	1	Get	8→1000
RES	4	SENSOR	3	Illumination	2	Get	8→1000
RES	4	SENSOR	3	Temperature	3	Get	8→1000

B. Naming and Addressing of Resource Identities

We define a simple resource model in which resources are logically organized into class. A class defines a group of resources, e.g., the hardware class contains all the resources that can be used for provisioning purposes. A resource is identified by the following path:

$$\sim /\{\mathsf{ClassID}\}.\{\mathsf{ResourceID}\}.\{\mathsf{SubresourceID}\}.\{\mathsf{MethodID}\}$$

where the class ID, resource ID, and subresource ID are with size of 1 byte. The Method ID[5] is to represent access methods available to a resource. The method ID is 4 bits and each bit from the most significant bit (MSB) represents an authorized operation in a sequence of GET, PUT, POST, and DELETE. The value "1" means authorized and "0" means nonauthorized. Table I shows the detailed naming and addressing assignment for resources on our sensor testbed. Each class is assigned a unique identity. If a resource does not support multiple subresources, the subresource ID is set as 0. For example, for retrieving the operating system version of the testbed, the resource identity should follow CoRE link format [43]. Through the resource discovery process GET $</$`.well-known`$/$`core`$>$, the IoT device should response with resources information, e.g., $<\sim /2.1.0.1000>$. It is noted that the proposed resource model can widely support any resources in practical implementation.

[5]The CoAP server may assign different method IDs to a same resource as long as clients' access levels are different. For example, administrator may have the full access rights of the whole IoT system, whereas some clients may only have "read" access to sensor devices.

TABLE II
COAP METHODS MAPPING TO DM FUNCTIONS

Function	Direction	Logical operation	CoAP method	Uri-path opt.	Location-path opt.
Registration	Uplink	Register	POST	√	√
	Uplink	Deregister	DELETE	√	√
	Uplink	Update	PUT	√	√
Provisioning	Down/ uplink	Configuration	GET	√	
Management services	Down/ uplink	Read	GET	√	
		Write	PUT	√	
		Execute	POST	√	
		Creat	POST	√	
Observing	Downlink	Observe	GET with observe opt.	√	
	Uplink	Notify	Response to observe	√	√
App. data transmission	Downlink	Collect	GET	√	

C. CoAP-Based Management Protocol

The proposed DM protocol is fully based on IETF-defined CoAP protocol with newly defined resource identities to identify management resources and mappings of CoAP methods to management functions.

The CoAP is based on the exchange of short messages which, by default, are transported over UDP (i.e., each CoAP message occupies the data section of one UDP datagram). The protocol has a registered scheme of $<$ `coap` : // $\sim>$ with a default port of 5683. Reliability over the UDP transport is provided by the built-in retransmission mechanism of CoAP, e.g., CON message defined by the Type field in the header. It could also be used over other transport protocols such as TCP or SMS. CoAP messages are encoded in a simple binary format. The message format starts with a fixed-size 4-byte header. This is followed by a variable-length token value which can be between 0 and 8 bytes long. Following the token value, it comes a sequence of zero or more CoAP options in type-length-value (TLV) format, optionally followed by a payload which takes up the rest of the datagram.

Table II shows the detailed CoAP methods mapping to DM functions. We should note that each management function can be abstracted as a recall process to conduct with resources on sensor device, thus the RESTful approach provided by CoAP protocol can be adopted as a lightweight method to access from application servers to wireless sensor devices. Especially, the uri-path option is to indicate management resource identities and the location-path option is to indicate the address of remote registration server for future update and delete operations.

V. PROTOTYPING SYSTEM

In this section, we present our prototyping system to implement the RESTful DM methods to IPv6 WSN. The network topology is shown in Fig. 3, where a laptop acts as a client to retrieve sensor resources via the RESTful gateway.

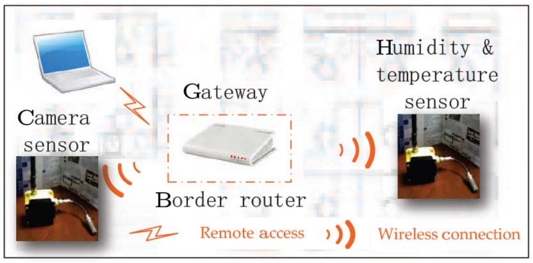


Fig. 3. Network topology of the prototype system: a laptop acts as a client to retrieve sensor resources via the RESTful gateway.

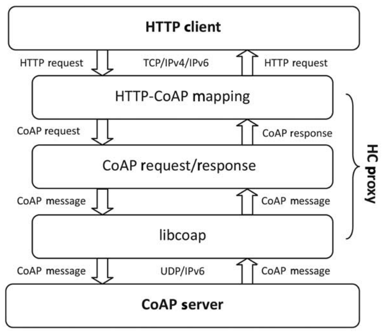


Fig. 4. Interaction process of HC proxy between a client and a server, which includes libcoap, CoAP request/response, and HTTP-CoAP mapping.

A. Sensor Nodes

We deploy wireless sensor devices to monitor indoor environment. The sensor platform is equipped with CC2530 MCU with 8051 CPU core running at 32 MHz, 8 KB SRAM, and 256 KB flash block to support IEEE 802.15.4-compliant radio transceiver. To support IPv6 connectivity, all sensor devices are running Contiki v2.6 operating system with implementation of 6LoWPAN, IPv6, and RPL protocols based on IEEE 802.15.4. The WS running on the sensor devices relies on the application protocol CoAP.

B. RESTfull Gateway

In order to ease the access from Internet applications to sensor resources, especially for those of Internet users who cannot speak CoAP, we integrate IEEE 802.15.4 connectivity into an open-platform gateway and port the HTTP-CoAP (HC) proxy to the OpenWrt, the operation system of the gateway, to access from an ordinary IP terminal to an IPv6 sensor device.

In our prototype gateway, the HC proxy is implemented based on libcoap [44], which is an open-source C-implementation of CoAP and conforms to GPL v2 or higher licenses. The interaction process of the HC proxy is shown in Fig. 4. Specifically, for each of the HC proxy layers, we have the following implementations.

1) libcoap Layer: It defines message structure and methods to implement the CoAP messages layer based on UDP.

2) CoAP Request/Response Layer: CoAP request/response layer encapsulates the data structure and methods relevant to CoAP requests and responses. It is to transmit CoAP request

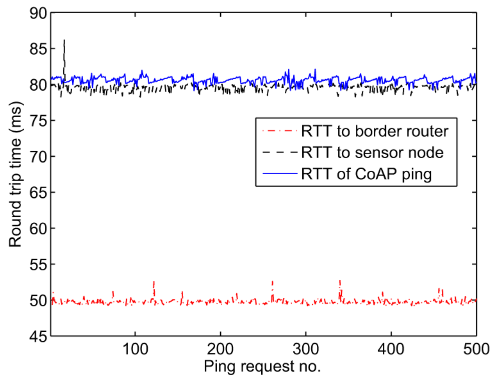


Fig. 5. RTT comparison between CoAP ping to sensor node and ICMP pings to both border router and sensor node.

in the form of CoAP message through the messages layer and generate CoAP response based on received CoAP messages.

3) HTTP-CoAP Mapping Layer: It is to implement mapping from HTTP request to CoAP request and vice versa. When converting a HTTP request to a CoAP request, the HC proxy needs to convert the HTTP request method, URI, header/option and payload, respectively. If a proxy encounters an error, it has to generate corresponding error response.

VI. Performance Evaluation

In this section, we provide evaluation results to illustrate the performance of the proposed CoAP-based DM solution and its application in IoT system.

A. System Configuration

Our prototype system is composed of sensor devices, one HC proxy gateway and another laptop for initiating tests. We deploy the prototype system in an open office area. The HC proxy gateway and sensor devices are connected wirelessly via IEEE 802.15.4 and using channel 26. The laptop client is connected to the gateway through the Wi-Fi channel. The network topology as shown in Fig. 3 is built with a maximum two hops, where the camera sensor and humidity&temperature sensor are connected to the gateway with one hop distance.

B. Latency Performance of CoAP Protocol

In order to manage a large-scale deployment of WSN, it is necessary to keep the protocol overhead as small as possible. In this evaluation, we compare the round trip time (RTT) of CoAP ping to a sensor node with that of Internet control message protocol (ICMP)[6] ping messages to both the border router and sensor node using Linux ping6 command. The ContikiMAC is configured as no sleep mode. The IPv6 packet size is fixed as 52 bytes. Fig. 5 shows the RTT comparison between the three methods over 500 independent measurements. The average RTT to the border router and sensor node using ICMP are 49.75 and 79.54 ms, respectively, and the average RTT of CoAP ping to sensor node is 80.5 ms. It is noted that the RTT to the sensor node is much higher than that to the border router, this is because of the processing time imposed at the gateway as well as one extra hop to transmit to the sensor node. However, the RTT of CoAP ping is very closed to the ICMP ping to the same sensor node, which only claims a 1.2% increases. The result tells that the overhead imposed by CoAP protocol is negligible and thus the CoAP-based DM is a promising solution for IoT.

[6]ICMP is an Internet layer protocol to control and report message error between a host server and a gateway to the Internet. ICMP uses IP datagrams, but the messages are processed by the IP software and are not directly apparent to the application user.

TABLE III
URI Length Comparison Between Standard CoAP Method and the Proposed DM Method

Resource	Standard URI	Bytes	Proposed URI	Bytes
CoAP_version	CoAP_v	45	3.6.0.8	8
LEDs	Actuators/LEDs	76	4.1.1.e	8
Illumination	Sensor/illumination	51	4.3.2.8	8
Temperature	Sensor/temperature	45	4.3.3.8	8
Humidity	Sensor/humidity	42	4.3.1.8	8
Screen	Screen	57	4.2.0.e	8

C. Packet Length, Latency, and Packet Loss Performance of the Proposed DM Solution

To further evaluate the performance of the proposed CoAP-based DM solution, we compare it with the standard CoAP method in terms of packet length, latency, and packet loss rate. Table III shows the onboard resources defined by both standard CoAP method (human-readable string) and the proposed method. The URI length is calculated from the space occupied in the RAM. It is clear that the proposed URI representation takes far less memory space than the standard URI representation in which the main space is consumed by "Attributes". Through the resource discovery GET $<$ /`.well-known`/`core` $>$, we can receive a response with a list of available resources as shown in Table III and the total length of transmission packets for both methods is 420 and 109 bytes, respectively. Since the CoAP response will be transmitted in a block-wise fashion, the standard method takes six blocks, whereas the proposed method only takes two blocks. The memory saving of 311 bytes is composed of URI savings and four extra CoAP block headers. The total transmitting packets can be reduced by 74%.

To further evaluate the latency and packet loss performance of the proposed DM solution, we setup a test to send GET request (i.e., $<$ /`.well-known`/`core` $>$) to retrieve onboard resources. Fig. 6 shows the histogram of the request/response delay and retransmission delay over 50 independent measurements. As can be seen from Fig. 6(a),[7] the proposed method shows a tendency of smaller delay with very high probability that the request/response delay is below 1 s. However, the performance of the standard method is varied from 1 to 10 s, because of the unreliable radio environment and a larger number of transmitting packets. Fig. 6(b) shows

[7]To best plot the histogram using MATLAB, we set the x-axis bin internal as 0.4 s, which means that a delay will be counted on the same bar if its value is within the same interval.

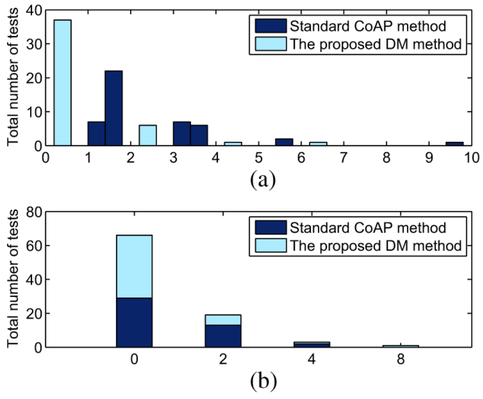


Fig. 6. Comparisons of request/response delay and retransmission delay. (a) Request/response delay (s). (b) Retransmission delay caused by packet loss (s).

the retransmission delay caused by packet loss; the proposed method shows smaller delay with a higher probability that the retransmission can be ensured as 0. A careful reader may notice that there are only even numbers of retransmission delay on *x*-axis, since once a blockwise packet or a CON message is missing, the CoAP server will postpone a retransmission (delay) in 2^n s, where n is the number of retransmission. As a result, the average request/response delay and retransmission delay for the standard method are 3.04 and 1.44 s, respectively, whereas the results for the proposed method are only 0.8 and 0.36 s, respectively.

At the end, we evaluate the packet loss rate over the 50 measurements and have the following results:

$$p_{\text{STD}} = 38/(38 + 50 \times 6) = 11.2\% \tag{1}$$

where 38 is the total number of retransmission and 50×6 is the total blocks of transmission. The result for the proposed solution is

$$p_{\text{Proposed}} = 10/(10 + 50 \times 2) = 9.1\% \tag{2}$$

where 10 is the total number of retransmission and 50×2 is the total blocks of transmission. Therefore, compared to (2) with (1), an improvement of 18.75% can be achieved by using the proposed solution.

In order to evaluate the performance of a large-scale network, we set up another test to evaluate the packet loss rate in a multihop environment. The test is carried out in an open office area with strong Wi-Fi background noise and lowest possible WSN radio frequency output power to ensure a multihop fashion, which makes a sensor device that can only communicate to each other within around 30 cm.

Since the sensor devices are with only 8 KB RAM and 256 KB flash, a maximum number of six hops can be obtained by optimizing the communication system. To retrieve the same onboard resources via GET request (i.e., $<$ /`.well-known/core` $>$) over the same number of measurements, Table IV shows the packet loss rate in a multihop scenario. We can observe that the packet loss rate increases

TABLE IV
PACKET LOSS RATE IN A MULTIHOP NETWORK

		Hop 2	Hop 3	Hop 4	Hop 5	Hop 6
Proposed DM method	Received	684	602	462	439	405
	Lost	43	126	262	289	321
	Packet loss rate (%)	5.91	17.30	36.18	39.69	44.21
Standard CoAP method	Received	2020	1704	1173	1112	944
	Lost	161	474	1003	1068	1234
	Packet loss rate (%)	7.38	21.76	46.09	48.9	56.65

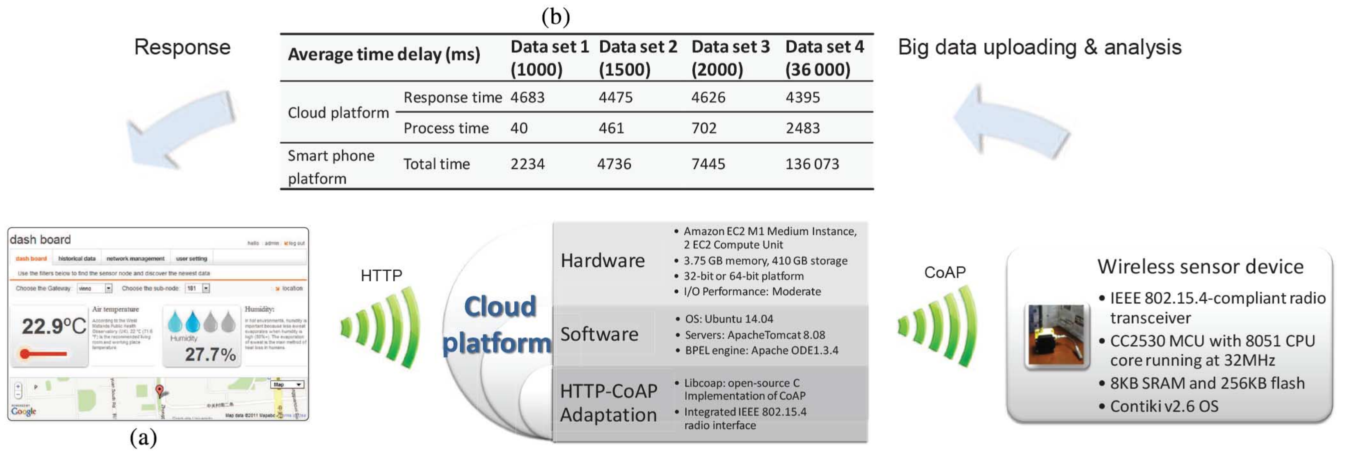
dramatically with an increasing number of hops, because of severe environmental interference and channel congestions, etc. Moreover, additional configurations to ensure a multihop transmission, such as one way communication, low output power, and RPL settings, also contribute to the high loss. However, the result does not affect the truth that the proposed solution can always achieve better performance than the standard method.

D. IoT Application via the Cloud Platform

To further validate the applicability of the proposed DM method, we integrate it into our prototype IoT cloud system by connecting senor devices via the cloud platform using the proposed CoAP-based management protocol. The snapshot of the management portal is shown in Fig. 7(a). Through the predefined CoAP APIs, interactions with application data can be easily managed and retrieved in a unified manner without remembering all string URIs.

Considering the limited resource of sensor devices, diverse contextual data need to be uploaded to the IoT cloud platform for further processing. Such data collected from independent IoT sources often have implicit but disparate assumptions of interpretation. We use a lightweight ontology, which contains a modifier used to capture additional information that affects the interpretations of generic concepts. More details about the setting of the cloud platform, e.g., the BPEL engine presented in Fig. 7, can be found in our previous work [45].

We evaluate the system performance of the IoT cloud in terms of time efficiency by setting up a simple test environment in which five sensor devices are used to upload computing tasks to the cloud platform with a total average rate of $\text{E} = 5/\text{min}$. The OpenGALEN ontology [46] is adopted as benchmark, and the computing tasks are to index and calculate the similarities of concepts on this ontology under the condition of four different size assertions (1000, 1500, 2000, 36 000). The average results are shown in Fig. 7(b). The time delay when performing the task via cloud consists of: 1) response and communication time between the remote IoT cloud platform and the sensor device and 2) processing time of the task. The results show closed performance of response time with an average of 4.5 s, while the process time mostly depends on the size of the data set. It is worth noting that depending on specific scenarios of IoT applications and computing capacities of IoT devices, we could choose different size of data set for real-world deployments.



Average time delay (ms)		Data set 1 (1000)	Data set 2 (1500)	Data set 3 (2000)	Data set 4 (36 000)
Cloud platform	Response time	4683	4475	4626	4395
	Process time	40	461	702	2483
Smart phone platform	Total time	2234	4736	7445	136 073

(b)
Response
Big data uploading & analysis
dash board
22.9°C
27.7%
(a)
HTTP
Cloud platform
Hardware
• Amazon EC2 M1 Medium Instance, 2 EC2 Compute Unit
• 3.75 GB memory, 410 GB storage
• 32-bit or 64-bit platform
• I/O Performance: Moderate
Software
• OS: Ubuntu 14.04
• Servers: ApacheTomcat 8.08
• BPEL engine: Apache ODE1.3.4
HTTP-CoAP Adaptation
• Libcoap: open-source C Implementation of CoAP
• Integrated IEEE 802.15.4 radio interface
CoAP
Wireless sensor device
• IEEE 802.15.4-compliant radio transceiver
• CC2530 MCU with 8051 CPU core running at 32MHz
• 8KB SRAM and 256KB flash
• Contiki v2.6 OS

Fig. 7. User-cloud-sensor interactions in the proposed IoT system and its performance. (a) Illustration of management Web portal. (b) Overall system performance of the IoT cloud.

VII. Conclusion and Future Work

We have proposed a CoAP-based DM solution and developed a prototype system to implement some basic functions on IPv6 sensor nodes. By integrating IEEE 802.15.4 connectivity and HTTP-CoAP proxy into an open-platform gateway, the remote access and management from an Internet device to an IPv6 sensor device can be realized in a unified manner. Through the performance evaluations, we have shown the simplicity and efficiency of the proposed solution, which is promising to drive IoT development.

In the future work, a more robust and reliable device management system for IoT needs to be built. Especially, the following research issues need to be considered with higher priorities. 1) *Real-time management* is a challenging issue for resource constrained sensor networks. The requirements of real-time communication comprises, in general, of response time in the required range and a small need of device resources, e.g., processor load and memory use [47]. The proposed CoAP-based method can potentially provide a lightweight solution to cope with stringent industrial applications. 2) *Security, trust, and privacy* [48] are also important issues to be considered in practical applications. In our case, the CoAP-based management principle can utilize the transport layer bindings of UDP or SMS protocols. Thus, the security mechanisms of these channel bindings can be utilized to implement access control and policy enforcement for IoT systems. 3) *Dynamic registration, bootstrap, and management* will be particularly considered for a large-scale deployment with devices coming in and out and changing their characteristics and functionalities.

References

[1] K. Ashton, "That 'Internet of Things' thing," *RFID J.*, Jul. 2009 [Online]. Available: http://www.rfidjournal.com/articles/view?4986

[2] A. Wheeler, "Commercial applications of wireless sensor networks using ZigBee," *IEEE Commun. Mag.*, vol. 45, no. 4, pp. 70–77, Apr. 2007.

[3] A.-B. García-Hernando, J.-F. Martínez-Ortega, J.-M. López-Navarro, A. Prayati, and A. P. L. Redondo-López, *Problem Solving for Wireless Sensor Networks*. New York, NY, USA: Springer, 2008.

[4] J. Song *et al.*, "WirelessHART: Applying wireless technology in real-time industrial process control," in *Proc. IEEE Real-Time Embedded Technol. Appl. Symp. (RTAS)*, Apr. 2008, pp. 377–386.

[5] J. Kjellsson, A. Vallestad, R. Steigmann, and D. Dzung, "Integration of a wireless I/O interface for PROFIBUS and PROFINET for factory automation," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, pp. 4279–4287, Oct. 2009.

[6] IP Smart Object Alliance (IPSO) [Online]. Available: http://www.ipso-alliance.org

[7] J. Jasperneite and J. Feld, "PROFINET: An integration platform for heterogeneous industrial communication systems," in *Proc. 10th IEEE Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2005, vol. 1, pp. 8–822.

[8] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Transmission of IPv6 packets over IEEE 802.15.4 networks," Internet Engineering Task Force, RFC 4944, 2007.

[9] T. Winter *et al.*, "RPL: IPv6 routing protocol for low-power and lossy networks," IETF, RFC 6550, 2012.

[10] Z. Shelby, K. Hartke, and C. Bormann, (2014 Jun.) *Constrained Application Protocol (CoAP)*, Internet Draft [Online]. Available: http://datatracker.ietf.org/wg/core/charter

[11] J. Stankovic, "Research directions for the Internet of Things," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 3–9, Feb. 2014.

[12] J. Song, A. Kunz, M. Schmidt, and P. Szczytowski, "Connecting and managing M2M devices in the future Internet," in *Mobile Networks and Applications*, vol. 19. New York, NY, USA: Springer, 2014, pp. 4–17.

[13] J. Swetina, G. Lu, P. Jacobs, F. Ennesser, and J. Song, "Toward a standardized common M2M service layer platform: Introduction to oneM2M," *IEEE Wireless Commun. Mag.*, vol. 21, no. 3, pp. 20–26, Jun. 2014.

[14] IoT-Architecture [Online]. Available: http://www.iot-a.eu/public

[15] H. Rachidi and A. Karmouch, "A framework for self-configuring devices using TR-069," in *Proc. Int. Conf. Multimedia Comput. Syst. (ICMCS)*, Apr. 2011, pp. 1–6.

[16] M. Ruta, F. Scioscia, E. Di Sciascio, and G. Loseto, "Semantic-based enhancement of ISO/IEC 14543-3 EIB/KNX standard for building automation," *IEEE Trans. Ind. Informat.*, vol. 7, no. 4, pp. 731–739, Nov. 2011.

[17] S. Yugang and W. Hong, "Research on field device integration model," in *Proc. Int. Conf. Meas. Technol. Mechatron. Autom. (ICMTMA)*, Mar. 2010, vol. 3, pp. 79–82.

[18] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, Dept. Inf. Comput. Sci., Univ. California at Irvine, Irvine, CA, USA, 2000.

[19] R. Kyusakov, J. Eliasson, J. Delsing, J. van Deventer, and J. Gustafsson, "Integration of wireless sensor and actuator nodes with IT infrastructure using service-oriented architecture," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 43–51, Feb. 2013.

[20] T. Cucinotta *et al.*, "A real-time service-oriented architecture for industrial automation," *IEEE Trans. Ind. Informat.*, vol. 5, no. 3, pp. 267–277, Aug. 2009.

[21] C. Pautasso, O. Zimmermann, and F. Leymann, "RESTful web services vs. 'big' Web services: Making the right architectural decision," in *Proc. 17th Int. Conf. World Wide Web (WWW)*, Apr. 2008, pp. 805–814.

[22] W. Qin, Q. Li, L. Sun, H. Zhu, and Y. Liu, "RestThing: A RESTful web service infrastructure for mash-up physical and web resources," in *Proc. IFIP 9th Int. Conf. Embedded Ubiq. Comput. (EUC)*, Oct. 2011, pp. 197–204.

[23] Z. Sheng, C. Zhu, and V. C. M. Leung, "Surfing the Internet-of-Things: Lightweight access and control of wireless sensor networks using industrial low power protocols," *EAI Endorsed Trans. Ind. Netw. Intell. Syst.*, vol. 14, no. 1, p. 12, 2014.

[24] D. Guinard and V. Trifa, "Towards the Web of Things: Web mashups for embedded devices," in *Proc. Int. World Wide Web Conf. (WWW)*, Apr. 2009.

[25] D. Guinard, V. Trifa, and E. Wilde, "A resource oriented architecture for the Web of Things," in *Proc. Internet of Things (IoT)*, Dec. 2010, pp. 1–8.

[26] J.-P. Vasseur and A. Dunkels, *Interconnecting Smart Objects With IP: The Next Internet*. San Mateo, CA, USA: Morgan Kaufmann, 2010.

[27] C. Hennebert and J. Dos Santos, "Security protocols and privacy issues into 6LoWPAN stack: A synthesis," *IEEE Internet Things J.*, vol. 1, no. 5, pp. 384–398, Oct. 2014.

[28] Z. Shelby, "Embedded web services," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 52–57, Dec. 2010.

[29] T. Potsch, K. Kuladinithi, M. Becker, P. Trenkamp, and C. Goerg, "Performance evaluation of CoAP using RPL and LPL in TinyOS," in *Proc. 5th Int. Conf. New Technol. Mobility Secur. (NTMS)*, May 2012, pp. 1–5.

[30] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki—A lightweight and flexible operating system for tiny networked sensors," in *Proc. 29th IEEE Int. Conf. Local Comput. Netw.*, Nov. 2004, pp. 455–462.

[31] M. Kovatsch, S. Duquennoy, and A. Dunkels, "A low-power CoAP for Contiki," in *Proc. IEEE 8th Int. Conf. Mobile Adhoc Sens. Syst. (MASS)*, Oct. 2011, pp. 855–860.

[32] J. Schönwölder, T. Tsou, and B. Sarikaya, "Protocol profiles for constrained devices," Tech. Rep., Mar. 2011 [Online]. Available: https://www.iab.org/wp-content/IAB-uploads/2011/03/Schoenwaelder.pdf

[33] O. Bergmann, K. Hillmann, and S. Gerdes, "A CoAP-gateway for smart homes," in *Proc. Int. Conf. Comput. Netw. Commun. (ICNC)*, Feb. 2012, pp. 446–450.

[34] G. Moritz, F. Golatowski, C. Lerche, and D. Timmermann, "Beyond 6LoWPAN: Web services in wireless sensor networks," *IEEE Trans. Ind. Informat.*, vol. 9, no. 4, pp. 1795–1805, Nov. 2013.

[35] L. Frye and L. Cheng, "A network management system for a heterogeneous, multi-tier network," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Dec. 2010, pp. 1–5.

[36] M. Yang, S. S. So, S. Eun, B. Kim, and J. Kim, "Sensos: A sensor node operating system with a device management scheme for sensor nodes," in *Proc. 4th Int. Conf. Inf. Technol.*, Apr. 2007, pp. 134–139.

[37] A. Hergenroeder, J. Wilke, and D. Meier, "Distributed energy measurements in WSN testbeds with a sensor node management device (SNMD)," in *Proc. 23rd Int. Conf. Archit. Comput. Syst. (ARCS)*, Feb. 2010, pp. 1–7.

[38] M. Jung, J. Weidinger, W. Kastner, and A. Olivieri, "Building automation and smart cities: An integration approach based on a service-oriented architecture," in *Proc. 27th Int. Conf. Adv. Inf. Netw. Appl. Workshops (WAINA)*, Mar. 2013, pp. 1361–1367.

[39] N. Gligoric, S. Krco, D. Drajic, S. Jokic, and B. Jakovljevic, "M2M device management in LTE networks," in *Proc. 19th Telecommun. Forum (TELFOR)*, Nov. 2011, pp. 414–417.

[40] P. van der Stok and B. Greevenbosch. (2014). "CoAp management interfaces (draft-vanderstok-core-comi-04)," IETF [Online]. Available: https://datatracker.ietf.org/doc/draft-vanderstok-core-comi/

[41] Z. Sheng *et al.*, "A survey on the IETF protocol suite for the Internet of Things: Standards, challenges, and opportunities," *IEEE Wireless Commun.*, vol. 20, no. 6, pp. 91–98, Dec. 2013.

[42] *Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*, I. C. Society, IEEE Standard 802.15.4-2003, 2003.

[43] Z. Shelby. (2012 Aug.) "Constrained RESTful environments (CoRE) link format," RFC 6690 [Online]. Available: http://tools.ietf.org/html/rfc6690

[44] O. Bergmann. (2014 Feb.) *Libcoap: C-Implementation of CoAP* [Online]. Available: http://libcoap.sourceforge.net

[45] X. Hu, X. Li, E.-H. Ngai, V. Leung, and P. Kruchten, "Multidimensional context-aware social network architecture for mobile crowdsensing," *IEEE Commun. Mag.*, vol. 52, no. 6, pp. 78–87, Jun. 2014.

[46] A. Rector, J. Roger, P. Zanstor, and E. Haring, "OpenGALEN: Open source medical terminology and tools," *Amer. Med. Informat. Assoc.*, vol. 2003, p. 982, 2003.

[47] J. Feld, "PROFINET–Scalable factory communication for all applications," in *Proc. IEEE Int. Workshop Factory Commun. Syst.*, Sep. 2004, pp. 33–38.

[48] S. L. Keoh, S. Kumar, and H. Tschofenig, "Securing the Internet of Things: A standardization perspective," *IEEE Internet Things J.*, vol. 1, no. 3, pp. 265–275, Jun. 2014.

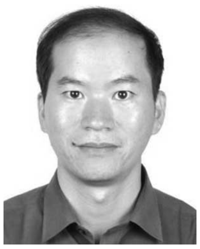
Zhengguo Sheng (GSM'07–M'10) received the B.Sc. degree from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2006, and the Ph.D. and M.S. degrees (with distinction) from Imperial College London, London, U.K., in 2011 and 2007, respectively.

He is currently a Lecturer with the School of Engineering and Informatics, University of Sussex, Sussex, U.K. He is also a Visiting Faculty Member with the University of British Columbia (UBC), Vancouver, BC, Canada. Previously, he was with UBC as a Research Associate, and with France Telecom Orange Labs as the Senior Researcher and Project Manager in M2M/IoT. He was also a Research Intern with the IBM T. J. Watson Research Center, Yorktown Heights, NY, USA, and U.S. Army Research Labs. His research interests include IoT/M2M, cloud/edge computing, vehicular communications, and power line communication (PLC).

Dr. Sheng was the recipient of the Auto21 TestDRIVE Competition Award in 2014 and Orange Outstanding Researcher Award in 2012.

Hao Wang received the B.S. and M.S. degrees in electrical engineering from Tsinghua University, Beijing, China, in 2005 and 2007, respectively.

He is currently a Senior Engineer with Orange Labs, Beijing, China. His research interests include the Internet of Things, device development, and communication protocol standardization and implementations.



Changchuan Yin (M'98–SM'15) received the Ph.D. degree in telecommunication engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 1998.

In Fall 2004, he held a visiting position with the College of Sciences and Technology, University of Sydney, Sydney, N.S.W., Australia. From 2007 to 2008, he held a visiting position with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, USA. He is currently a Professor with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications. His research interests include wireless networks and statistical signal processing.

Dr. Yin has served for various IEEE conferences as a Technical Program Committee (TPC) member. He was the corecipient of the IEEE International Conference on Wireless Communications and Signal Processing Best Paper Award in 2009.

Xiping Hu (GSM'15) is currently working toward the Ph.D. degree in electrical and computer engineering at the University of British Columbia (UBC), Vancouver, BC, Canada.

In 2011, he joined the UBC. He is currently a Key Member in several research projects, like Web service security identification at Tsinghua University, Beijing, China, the SAVOIR project at National Research Council of Canada, Institute for Information Technology (NRC-IIT), and NSERC DIVA strategy research network at UBC. He is the Co-Founder and Chief Technical Officer (CTO) of Bravolol, Hong Kong, a leading language learning mobile application company with over 40 million accumulated downloads and 6 million monthly active users. His research interests include mobile social networks, mobile computing, and crowdsourced-sensing.

Dr. Hu was the recipient of Silver Prizes in national Olympic competitions in mathematics and physics in China, and a Microsoft certified specialist in Web applications, .NET framework, and SQL server. As first author, his research contributions have been published and presented in around 20 international conferences and journals, such as the IEEE HICSS, the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING (TETC), *IEEE Communications Magazine*, and *ACM MobiCom*.

Shusen Yang (M'14) received the Ph.D. degree in computer science from Imperial College London, London, U.K., in 2013.

He is currently a Lecturer with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, U.K. Before joining the University of Liverpool, he was a Research Associate with Imperial College London, and the Intel Collaborative Research Institute (ICRI) for Sustainable Connected Cites. His research interests include adaptive, sustainable, scalable, and secure solutions to distributed wireless networks and sensing systems, including the Internet of Things, fog networking and computing, cyber-physical systems, and energy harvesting networks.

Dr. Yang is a member of ACM.

Victor C. M. Leung (S'75–M'79–SM'97–F'03) received the B.A.Sc. (Hons.) and Ph.D. degrees in electrical engineering from the University of British Columbia (UBC), Vancouver, BC, Canada, in 1977 and 1982, respectively.

He is currently a Professor of electrical and computer engineering and holder of the TELUS Mobility Research Chair with UBC. He has co-authored more than 700 technical papers in archival journals and refereed conference proceedings. His research interests include wireless networks and mobile systems.

Dr. Leung is a Fellow of the Royal Society of Canada, the Canadian Academy of Engineering, and the Engineering Institute of Canada. He is serving/has served on the Editorial Boards of the IEEE JOURNAL ON SELECTED AREAS OF COMMUNICATIONS, the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS LETTERS, and several other journals. He has provided leadership to the technical committees and organizing committees of numerous international conferences. He was the recipient of an APEBC Gold Medal, NSERC Postgraduate Scholarships, a 2012 UBC Killam Research Prize, and an IEEE Vancouver Section Centennial Award. He was also the recipient of several Best Paper Awards.