A Container-Driven Service Architecture to Minimize the Upgrading Requirements of User-Side Smart Meters in Distribution Grids

Li Liu, Yuemin Ding, *Member, IEEE*, Xiaohui Li, Huaming Wu, *Member, IEEE*, and Lantao Xing, *Member, IEEE*

***Abstract*—Advances in information and communication technologies have significantly influenced the operation of low-voltage distribution grids. As essential elements of distribution grids, user-side smart meters find many smart grid applications, for example to measure electrical energy use and facilitate communications. However, the service models of distribution grids remain under development in association with upgrading of user-side smart meters. These meters are resource constrained, and challenging to upgrade on a large scale. To address this issue, this article describes a container-driven service architecture, in which containers are used to create a virtual dedicated agent (digital twin) for each user-side smart meter. The agent can be deployed either in the cloud or on an edge system, and can be upgraded to support emerging smart grid applications, thus minimizing the future upgrading requirements of user-side smart meters. We built experimental test beds to verify the proposed architecture and evaluated its performance in real-world experiments.**

***Index Terms*—Cloud/edge computing, container, digital twin, meter upgrading, smart meter.**

I. Introduction

SMART grids integrate information and communication technologies with electrical grids, allowing electricity users and distributed energy resources to actively participate in grid operation and maintenance [1]. Low-voltage distribution grids play a more important role today than in the past. They support many innovative smart grid applications, including advanced metering infrastructures, real-time pricing/incentives, user-side demand responses, peer-to-peer energy trading, and integration of renewable energy storage [2]. To meet the requirements of such emerging smart grid applications, major upgrading of existing distribution grid facilities is required.

In a low-voltage distribution grid, user-side smart meters are essential to support smart grid applications [3], [4]. Smart meters not only provide conventional functionalities, such as measurement of electricity use and billing, but are also key for last-mile communication. They serve as a bridge between neighborhood area networks (NANs) and local area networks (LANs) on the demand side (homes, buildings, and factories), enabling bidirectional communication between utility suppliers and electricity users [5]. Thus, smart meters provide essential support for new applications of smart grids, such as demand responses and peer-to-peer energy trading, as shown in Fig. 1.

Many communication protocols and service models for low-voltage distribution grids remain under development (see Fig. 1),

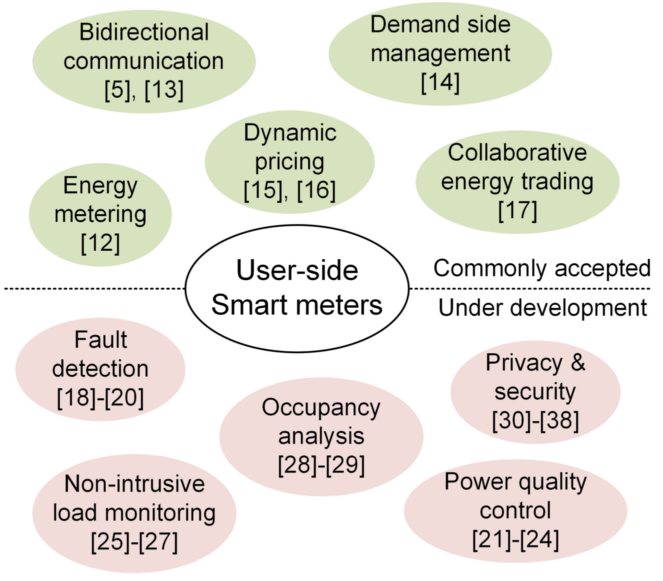
Bidirectional communication [5], [13]
Demand side management [14]
Dynamic pricing [15], [16]
Collaborative energy trading [17]
Energy metering [12]
User-side Smart meters
Commonly accepted
Under development
Fault detection [18]-[20]
Privacy & security [30]-[38]
Occupancy analysis [28]-[29]
Non-intrusive load monitoring [25]-[27]
Power quality control [21]-[24]

Fig. 1. Smart meter-related applications in a smart grid.

Manuscript received February 16, 2021; revised May 3, 2021; accepted May 28, 2021. Date of publication June 11, 2021; date of current version September 29, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61702369 and Grant 62071327, in part by Tianjin Natural Science Foundation under Grant 18JCZDJC30700, and in part by the Open Research Project of the State Key Laboratory of Industrial Control Technology, Zhejiang University, China under Grant ICT2021B56. Paper no. TII-21-0703. *(Corresponding author: Yuemin Ding.)*

Li Liu is with the School of Computer Science and Engineering, Tianjin University of Technology, Tianjin 300384, China (e-mail: 18611429192@163.com).

Yuemin Ding is with the Department of Energy and Process Engineering, Norwegian University of Science and Technology, NO-7491 Trondheim, Norway (e-mail: yuemin.ding1986@gmail.com).

Xiaohui Li is with the Department of Information Science and Engineering, Wuhan University of Science and Technology, Wuhan 430081, China (e-mail: bestxiaohui.li@gmail.com).

Huaming Wu is with the Center for Applied Mathematics, Tianjin University, Tianjin 300072, China (e-mail: whming@tju.edu.cn).

Lantao Xing is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore (e-mail: lantao.xing@ntu.edu.sg).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2021.3088135.

Digital Object Identifier 10.1109/TII.2021.3088135

and both may change or be upgraded in the future. Therefore, as essential smart elements of distribution grids, user-side smart meters must be upgraded occasionally to meet the emerging requirements of smart grid applications, and to adhere to newly approved communication standards [6]. However, it is both costly and technically difficult to upgrade user-side smart meters [7], which are resource-constrained devices with limited computation, communication, and storage capabilities. Thus, it would be valuable to create an easily updatable service architecture to support emerging smart grid applications.

This article describes a container-driven service architecture that minimizes the upgrading requirements of user-side smart meters in distribution grids. Using containers, a dedicated virtual agent (digital twin) is created either in the cloud or on an edge system for each user-side smart meter. Application-oriented communication and implementation of service models are moved from user-side smart meters to dedicated container agents. The user-side smart meters are embedded with only a minimum set of core functions, such as energy measurement and an interface for one-to-one communication with the virtual agent. In this manner, communication protocols or service models can be upgraded by upgrading the dedicated containers; the upgrading requirements of user-side smart meters are minimized. To demonstrate the advantages of the proposed architecture, we built experimental test beds and performed real-world experiments to evaluate system performance in typical smart grid applications. The contributions of this article can be summarized as follows.

1) The problem of user-side smart meter upgrading is clarified; this is an important issue that has rarely been discussed.
2) A service architecture is proposed (based on container technology) to minimize the upgrading requirements of user-side smart meters.
3) Application-oriented communication and service models are realized using virtual dedicated containers, which are much easier to upgrade.

The rest of this article is organized as follows. Section II summarizes related work. Section III details the proposed container-driven service architecture, which minimizes user-side smart meter upgrading. Section IV deals with the experimental implementation and analysis. Section V discusses certain limitations and the business model of the proposed architecture, which requires further study. Finally, Section VI concludes this article.

II. Related Work and Motivation

This article aimed to minimize the upgrading requirements of user-side smart meters in the smart grid. The smart grid has attracted significant interest during the past decade. Many studies have discussed applications of smart grids, including a road map [8], [9], smart grid communication architecture [4], [10], and smart grid-based business models [11]. These studies clearly illustrate the importance of the user-side smart meters.

The remainder of this section reviews the literature on recent advances in user-side smart meters and related applications, following the structure in Fig. 1. Then, the technical issues, which motivated this research, are discussed.

A. Smart Meter Applications

Several reviews have examined the evolution of user-side smart meters and smart grid applications [4], [10]. Smart meters have applications to energy metering [12], bidirectional communication [5], [13], demand-side management [14], dynamic pricing [15], [16], and peer-to-peer energy trading [17]. However, the communication protocols and service models may be upgraded in the future. For example, the OpenADR protocol, which was created in 2002, had been revised until 2018 and approved as an international standard, will be considered for further revision in 2024. Thus, user-side smart meters may have to be modified to support new functionalities.

Many potential applications of smart meters are under investigation, including fault detection, big data analysis, privacy and security enhancement, etc. The voltage measurements of smart meters were used in [18] to detect high impedance faults in distribution systems, overcoming the lack of data availability near-fault points. Trindade and Freitas [19] proposed that smart meter data be used to construct low-voltage zones; this would reduce the number of estimations made by existing impedance-based methods and help locate the fault. In [20], a multiple hypothesis method based on integer programming was used to manage outages by integrating information from smart meters.

In terms of power quality control, smart meter signals were used in [21] to extract features that rapidly identified disturbances. In [22], an unbundled smart meter architecture was used to integrate new functionalities with the data provided by superficial assessment of power quality. In [23], a real-time power quality disturbance detection method for smart meters, based on a support vector machine and practical wavelet filters, was developed. In [24], a deep learning method with practical applicability was used to detect and classify power quality disturbances.

For big data analysis with smart meters, Alahakoon and Yu [25] focused on the use of smart meter data for intelligent applications, particularly nonintrusive load monitoring, and occupant behavior analysis. A load decomposition scheme using metering data was introduced in [26] to monitor and understand load behaviors at the smart meter level, enabling near-real-time analysis. In [27], a nonintrusive demand monitoring and load identification scheme using smart metering data was proposed; particle swarm optimization was used to improve accuracy and computational efficiency. In [28], smart metering data were used to obtain household electricity consumption profiles, facilitating energy-efficient management. In [29], smart metering data were used to monitor both residential and commercial buildings.

The privacy and security issues of smart meters have attracted significant recent interest. In [30], a cost-effective and privacy-preserving dynamic programming framework was designed; this prevents identification of behavior/occupancy patterns. Other studies protected the privacy of smart metering data

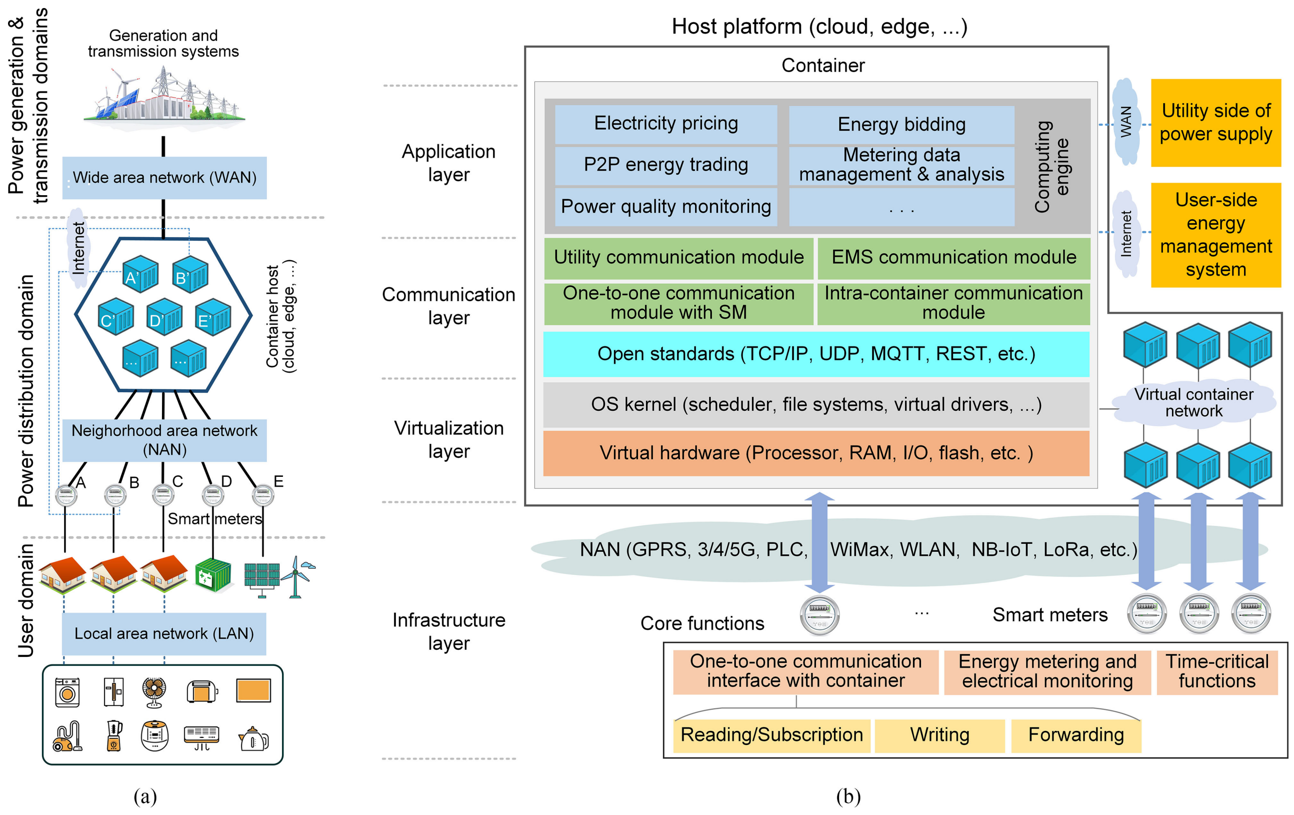
Power generation & transmission domains
Generation and transmission systems
Wide area network (WAN)
Internet
Power distribution domain
Container host (cloud, edge, ...)
A' B' C' D' E'
Neighorhood area network (NAN)
A B C D E
Smart meters
User domain
Local area network (LAN)
(a)
Host platform (cloud, edge, ...)
Container
Application layer
Electricity pricing
Energy bidding
P2P energy trading
Metering data management & analysis
Power quality monitoring
. . .
Computing engine
WAN
Utility side of power supply
Internet
User-side energy management system
Communication layer
Utility communication module
EMS communication module
One-to-one communication module with SM
Intra-container communication module
Open standards (TCP/IP, UDP, MQTT, REST, etc.)
Virtualization layer
OS kernel (scheduler, file systems, virtual drivers, ...)
Virtual hardware (Processor, RAM, I/O, flash, etc.)
Virtual container network
NAN (GPRS, 3/4/5G, PLC, WiMax, WLAN, NB-IoT, LoRa, etc.)
Infrastructure layer
Core functions
...
Smart meters
One-to-one communication interface with container
Energy metering and electrical monitoring
Time-critical functions
Reading/Subscription
Writing
Forwarding
(b)

Fig. 2. Container-driven service architecture. (a) System model. (b) Layered architecture.

and other user-side facilities, such as batteries [31], [32] and water heaters [33]. For security reasons, lightweight message authentication and secure communication schemes were proposed in [34] and [35] to meet the security requirements of smart grid communication. In [36]–[38], key establishment and exchange schemes were introduced for communication between smart meters and service providers; practical implementations and formal verification were discussed.

Although some smart grid applications that require support from user-side smart meters remain under development, many smart meters have already been installed on the user side [6], these may have to be upgraded in future.

B. Technical Gaps and Motivation

Existing studies on user-side smart meters focused principally on new smart grid applications or functionalities, many of which remain under investigation. Both the communication protocols and service models of smart grid applications will likely be revised or changed in future, so there will be a need to upgrade smart meters that have already been deployed to the user side. As smart meters are resource-constrained devices, upgrading would be challenging, but has attracted little attention.

To tackle this issue, this article presents a container-driven service architecture to minimize the upgrading requirements of user-side smart meters. In the proposed architecture, application-oriented communication protocols and service models are created in container-driven virtual agents, maximizing the concision, and stability of user-side smart meters.

III. Container-Driven Service Architecture

This section offers detailed information on the system model and proposes a container-driven architecture to minimize the upgrading requirements of user-side smart meters.

A. System Model

Fig. 2(a) illustrates the system model of the proposed architecture, which has power generation, transmission, distribution, and user components. In addition to the typical information infrastructure of a smart grid, which includes a wide area network (WAN), NAN, and LAN [4], [5], a container host, which can be deployed either in the cloud or on an edge system, is included to minimize the upgrading requirements of user-side smart meters. The container host provides space for containers, which are lightweight and have been successfully used to deploy and manage applications in cloud computing [39].

In the proposed architecture, each container serves as a dedicated communication and service agent for a user-side smart meter; containers A' and B' service smart meters A and B, respectively. The containers are mutually insulated to prevent any disruption of the operation of other containers. With the support of containers, this actually creates a digital twin for each physical smart meter on the user side. To enable collaboration among neighborhood smart meters, a virtual network is used to exchange messages among containers. In addition, the containers communicate with their individual user-side smart meters via a one-to-one communication interface, and exchange information on behalf of the smart meters with application servers, peer smart meters, and users if needed. In this manner, application-oriented functionalities that currently must be implemented by user-side smart meters are moved into dedicated virtual containers. If smart grid applications change, only the dedicated containers, and not the user-side smart meters, require upgrading.

B. Layered Architecture

Fig. 2(b) shows the container-driven service architecture. Five layers are arranged in a bottom-up manner: an infrastructure layer, a virtualization layer, a communication layer, a computation layer, and an application layer. Their functions are described as follows.

The infrastructure layer provides infrastructural support. It is composed principally of NANs and user-side smart meters. A NAN establishes communication between smart meters and the dedicated containers. To reduce upgrading requirements, the functions embedded in smart meters should be as concise as possible. In the proposed architecture, only three types of core functions are implemented: a communication interface, electrical functions (such as energy metering and electrical monitoring), and time-critical functions. The communication interface establishes communication with the dedicated virtual container and exchanges messages with user-side energy management systems (EMSs). All other functionalities are transferred to the virtual containers. The functions of the user-side smart meter are simplified as much as possible.

The virtualization layer is an insulated operating environment; each container serves as a dedicated agent of a user-side smart meter. In this layer, virtual hardware simulates the hardware components of a computing system, such as processing units, memory, input and output devices, and flash drives. The kernel of the operating system, including drivers, a scheduler, and a filing system, lies on top of the virtual hardware. The virtualization layer also hosts the upper layers, i.e., the communication and application layers.

The communication layer interacts with all other elements of the proposed architecture. Depending on the roles played by the user-side smart meters, four types of communication are supported by each virtual container: communication with the utility, one-to-one communication with the user-side smart meter, communication with user-side EMSs, and communication with peer containers. These are described in more detail as follows.

1) When communicating with the utility, a virtual container serves as the dedicated agent of a physical user-side smart meter, and simulates its behaviors. To do this, it follows a standard (such as OpenADR) protocol for communication between a utility and smart meter. The utility is not aware that it is communicating with a virtual container because the container behaves in the same manner as the physical user-side smart meter. No upgrading of the utility is required to support the proposed service architecture, thus guaranteeing the consistency of the entire system.
2) As one virtual container serves as the agent of only one user-side smart meter, one-to-one communication is required between the two. For smart grid applications, only three types of communication are necessary: reading/subscription, writing, and forwarding. Reading/subscription is used by a container to read/subscribe to metering or monitoring data from the smart meter; writing enables the container to send commands (e.g., pertaining to the need to assume a certain configuration) to the smart meter; and forwarding enables a smart meter to directly forward the message to a user-side EMS without processing it locally. In this manner, messages to/from the smart meters are simplified. Security and authentication issues are minimized by the reduced complexity and simplified one-to-one communication with the dedicated container.
3) The communication interface with user-side EMSs enables direct communication between the container and user-side facilities without any support from the smart meter. This reflects the fact that not all smart meters can communicate with user-side EMSs, which, thus, cannot participate in demand response programs. In the user domain, many electrical appliances can be remotely monitored/controlled through the Internet. Enabling communication between the container and user-side facilities allows the facilities to engage in demand-side management, even without the support from a user-side smart meter.
4) The intracontainer communication interface allows for message exchange among containers. Given the power distribution systems of a smart grid, it is essential that electricity users and distributed energy resources collaborate with respect to grid operation and maintenance. In the proposed architecture, this function is transferred to the virtual container. The communication interface implemented in the traditional smart meters is realized as an intracontainer communication interface. Communication among containers adheres to the protocol for communication among smart meters, enabling reuse of existing technologies. Also, smart meters have limited communication bandwidth due to the limitations of NAN technologies. The virtual network that enables communications among virtual containers has much greater bandwidth and data transmission rate.

The application layer accommodates the smart grid applications of distribution grids, including electricity pricing and incentives, energy bidding, peer-to-peer energy trading, power quality monitoring, smart metering data management and analysis, and blockchain. These applications were previously supported by user-side smart meters but this has now shifted to the containers. In addition, a computing engine provides the lightweight computational capabilities required by smart grid applications; these include data encryption, lightweight optimization, and learning-based analysis. The computing engine is designed and implemented by the utility; the availability of multiple options allows users to select what they require.

C. Example Upgrading Process

Using the architecture illustrated in Fig. 2, Fig. 3 shows how to upgrade existing smart grid applications or create new ones.

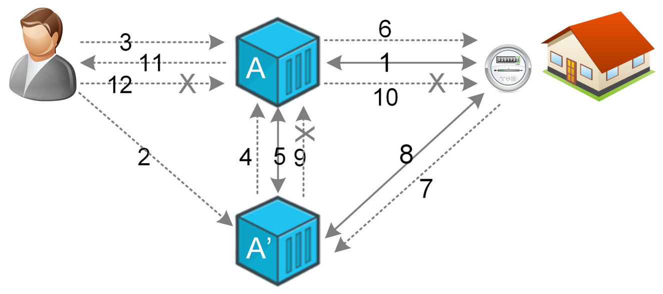

Fig. 3. Example upgrading process.

The solid lines indicate data flow and the dashed lines control flow. In detail, the following conditions hold.

1) Flow 1 indicates the existing data flow (such as energy metering data) between a user-side smart meter and its dedicated container.
2) Flow 2 configures an additional container A' (created by the system manager) with updated or new smart grid applications. This occurs only when a smart grid application requires upgrading or a new smart grid application is introduced.
3) In Flow 3, the system manager informs the dedicated container agent A, that an upgrading process has commenced, and that a new container A', has been created as its successor.
4) Flows 4 and 5 show that the new container A', requests A to synchronize and share existing data.
5) After synchronization is complete, A informs the user-side smart meter that a container with upgraded/new applications has been created (Flow 6).
6) After receiving this information from container A, the user-side smart meter sends a request to the new container A' to establish a connection (Flow 7).
7) In Flow 8, data flow between the user-side smart meter and the new container A' is established. Note that data flow between container A and the user-side smart meter continued up to this point.
8) After receiving data from the user-side smart meter, the new container A' and the user-side smart meter informs container A that they have successfully established a connection. Then, the links with container A are removed (Flows 9 and 10).
9) In Flow 11, container A informs the system manager that upgrading has been completed, after receiving information to that effect from both container A' and the user-side smart meter.
10) Flow 12 indicates that the system manager destroys container A and frees up its memory and storage space; this completes the upgrading process.

This enables the system manager to introduce new smart grid functions, and upgrade existing applications, in a flexible and seamless manner. The user-side smart meters remain unchanged, without any interruption of user-side services. This is realized by moving application-oriented communications and service models from user-side smart meters to virtual containers, which are much easier to upgrade and deploy.

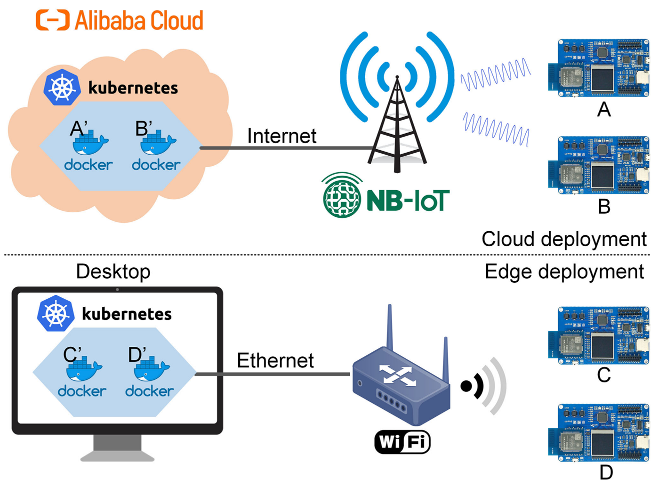

Fig. 4. Experimental setup with the proposed architecture: (a) cloud deployment and (b) edge deployment.

TABLE I
SOFTWARE AND HARDWARE TOOLS USED FOR EXPERIMENTAL VERIFICATION

Tools	Description
Alibaba cloud[1]	Cloud host for containers.
Desktop	Edge host for containers.
Kubernetes[2]	Containers deployment.
Docker[3]	Container environment as virtual hardware.
Alpine Linux[4]	Operating system.
OpenLEADR[5]	Implementation of OpenADR protocol.
ESP8266	Wifi communication module.
BearPi	Hardware of smart meters with integrated NB-IoT.

IV. EXPERIMENTAL SETUP AND ANALYSIS

This section introduces the test bed setup of the proposed architecture and the design of the experimental scenarios. The experimental results are then collected and analyzed in detail.

A. Test Beds

The test beds for the proposed architecture are shown in Fig. 4. The upper part of Fig. 4 uses a cloud platform as the container host and the Narrowband Internet of Things (NB-IoT) as the communication method of user-side smart meters, while the lower part adopts an edge system (a desktop computer) as the container host and WiFi as the communication methodology for the user-side smart meters. Table I summarizes the hardware and software tools used to set up the test beds.

For the cloud deployment shown in Fig. 4, Alibaba Cloud serves as the container host, integrated with Kubernetes and Docker containers. On top of the Docker container, Alpine Linux, a security-oriented, and lightweight Linux distribution, serves as the operating system that creates dedicated virtual

[1][Online]. Available: https://www.alibabacloud.com/
[2][Online]. Available: https://www.kubernetes.io/
[3][Online]. Available: https://www.docker.com/
[4][Online]. Available: https://www.alpinelinux.org/
[5][Online]. Available: https://github.com/OpenLEADR/openleadr-python/

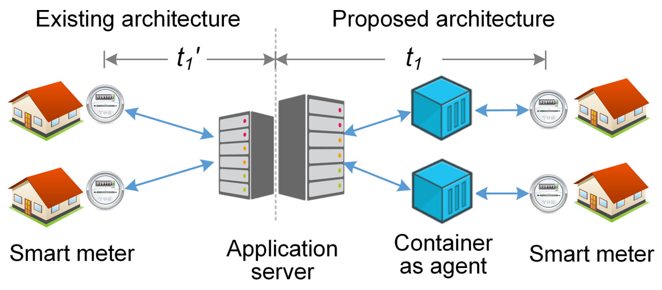

Fig. 5. Experimental scenario for communication between smart meters and application servers.

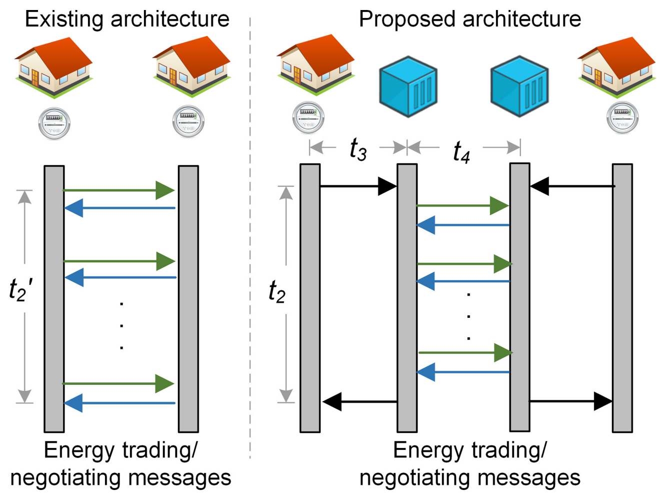
Existing architecture
Proposed architecture
t_3
t_4
t_2'
t_2
Energy trading/ negotiating messages
Energy trading/ negotiating messages

Fig. 6. Experimental scenario for iterative communication of peer-to-peer applications.

agents for smart meters and hosts smart grid applications previously embedded in user-side smart meters. These meters are realized using BearPi, which is integrated with an NB-IoT communication module. Using NB-IoT, BearPi terminals communicate directly with the base station of the network service provider. This base station then connects with the Alibaba Cloud through the Internet. In this manner, each BearPi terminal establishes one-to-one communication (via NB-IoT) with a dedicated container in the cloud; A communicates with A'. The containers communicate/collaborate using the inner network of Alibaba Cloud.

For the edge deployment of Fig. 4, a desktop computer is used to build an edge system, in which Kubernetes and Docker containers are then installed following the instructions of the developers. Here, the user-side smart meters communicate via WiFi technology, which is also widely used for neighborhood communication. The smart meters are based on BearPi with the plugin WiFi module ESP8266. The other configurations are the same as those of the cloud deployment. Note that other platforms, such as Android and embedded Linux systems, can also be used to build edge systems.

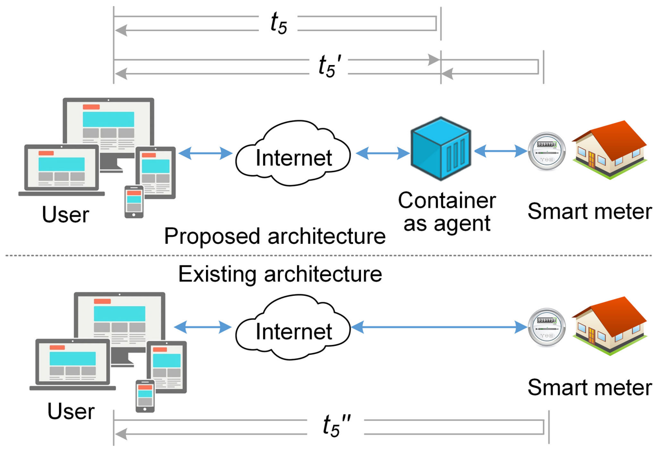

Fig. 7. Experimental scenario for communication of on-demand applications.

B. Experimental Scenarios

To verify the effectiveness of the proposed architecture, and to evaluate its performance in terms of smart grid applications, three experimental scenarios were designed.

Scenario 1 evaluates how the proposed architecture impacts communication between smart meters and application servers, such as the demand response server. As illustrated in Fig. 5, end-to-end delays with and without the proposed architecture, namely t_1 and t_1', are measured for both the cloud and edge deployments. To obtain the end-to-end delay, a round-trip time (RTT) is first measured, and t_1 or t_1' is estimated as half of the RTT value.

Scenario 2 investigates how the proposed architecture performs in terms of iterative communication (peer-to-peer services). A typical smart grid application of this type is energy trading; iterative communication is used to negotiate the energy price and amount (see Fig. 6). The time taken to complete peer-to-peer negotiation is determined for cases with and without the proposed architecture, namely t_2 and t_2' (see Fig. 6). The communication delays between containers and user-side smart meters t_3 and t_4 are also evaluated.

Scenario 3 describes how the proposed architecture affects the communications of on-demand services in a smart grid. A good example is on-demand meter reading with a low latency requirement, as shown in Fig. 7. Using the proposed architecture, two cases are considered: 1) if the requested data are available on the dedicated container, it replies directly to the user request, of which the communication time is t_5 and 2) if the requested data are not available on the dedicated container, it communicates further with the user-side smart meter to obtain the requested data before replying to the user request, leading to a delay of t_5'. However, in the absence of the proposed architecture, the data can be obtained only from the user-side smart meters, leading to t_5''. The communication delays, t_5, t_5', and t_5'' are compared with experiments.

For all the experimental scenarios, the experimental settings are summarized as the following.

1) The application servers are deployed on a cloud platform, which is roughly 120 km from the user-side smart meters.
2) The containers are deployed on the same cloud platform with the application servers for the cloud deployment of

TABLE II
COMMUNICATION REQUIREMENTS OF SMART METER-RELATED APPLICATIONS [40]

Application	Latency	Reliability
Advanced metering Infrastructure	5s-several hours	99.0%-99.99%
On-demand meter reading	$<$ 5s	$>$ 99%
Multi-interval meter reading	$<$ 4hours	$>$ 99%
Demand response management	500 ms-min	$>$ 99%
Electricity pricing	$<$ 1 min	$>$ 98%
Grid-to-Vehicle	2s-5 min	99%-99.99%
Vehicle-to-Grid	$<$ 15s	99%-99.99%

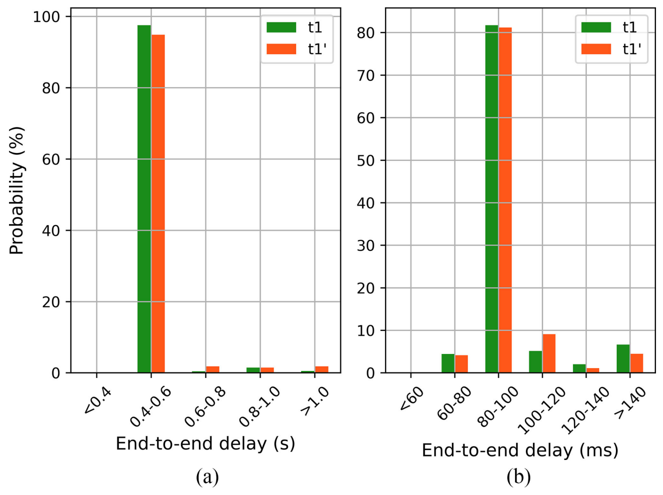

Fig. 8. Statistical probability of end-to-end delays between smart meters and application servers.

the proposed architecture and are deployed on a local desktop computer next to the user-side smart meters for the edge deployment.

3) The users' smart devices are simulated with another cloud platform, which is roughly 500 km from the user-side smart meters and 660 km from the application servers.
4) The communications between containers and application servers/users/peer containers adopt the TCP/IP protocol, and the communications between containers and the user-side smart meters follow the MQTT standard. For both cases, the application data payloads are set to 512 Bytes.
5) Each experimental configuration runs for 24 h with one-minute communication intervals to collect latency data.

The experimental data are analyzed according to the requirements of smart grid applications (summarized in Table II) to verify the effectiveness of the proposed architecture.

C. Results and Analysis

Based on the experimental scenarios described previously, this section presents the experimental results.

1) Results of Scenario 1: Fig. 8 depicts the statistical probabilities of the end-to-end delays for Scenario 1 (see Fig. 5). Note that the statistical probability of t_1 is similar to that of t_1' with cloud deployment using the NB-IoT [see Fig. 8(a)]. A further numerical analysis indicates that the average difference

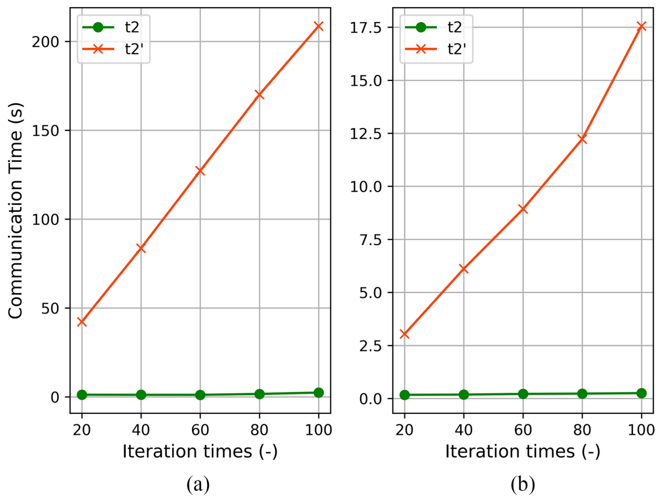

Fig. 9. Communication times of peer-to-peer applications.

between t_1 and t_1' is within 10 ms, which can be easily caused by the dynamic network conditions.

The statistical probabilities of t_1 and t_1' on edge deployment are shown in Fig. 8(b); it can be seen that they are very similar. As the container is deployed on an edge near the user-side smart meter, any impact on communication latency is negligible under dynamic network conditions. Although the proposed architecture may slightly increase the latency between smart meters and application servers given the inclusion of containers, the increment is very small and can be ignored. For both the cloud and edge deployments, t_1 satisfies the time requirements of smart grid applications listed in Table II.

2) Results of Scenario 2: Fig. 9 shows the communication time with the number of iterations for Scenario 2 (see Fig. 6). For the cloud deployment, the communication time of the proposed architecture t_2, is significantly shorter than that without the proposed architecture, t_2'. A peer-to-peer negotiation with 100 iterations requires less than 2.4 s on average using the proposed architecture, compared to 208 s without the architecture. This is because, as dedicated agents of the smart meters, containers can negotiate with peers directly and forward the final decision to the user-side smart meters. This is impossible without the proposed architecture; user-side smart meters must then participate in every iteration of peer negotiation.

For the edge deployment shown in Fig. 9, t_2 is also less than t_2'. t_2 has an average value of 0.2 s for a negotiation with 100 iterations; t_2' is then 17.5 s. Even if the smart meters are located in the same WiFi network and can, thus, communicate directly, the communication time t_2' is still longer than that of the proposed architecture (t_2). In reality, smart meters involved in peer-to-peer negotiation may not be within a one-hop distance. In such a case, the proposed architecture reduces the communication time even further.

To better illustrate why the proposed architecture reduces the time required for iterative peer-to-peer communications, the communication delay between smart meters and containers t_3, and the communication delay between containers t_4, were measured for both the cloud and edge deployments, as shown

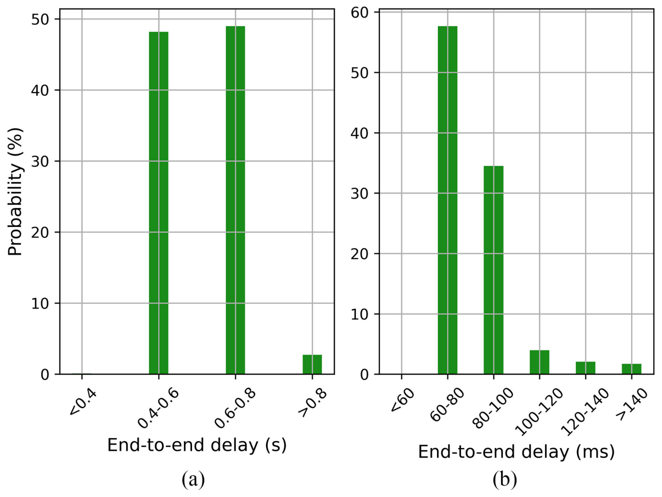

Fig. 10. Communication delays between smart meters and containers.

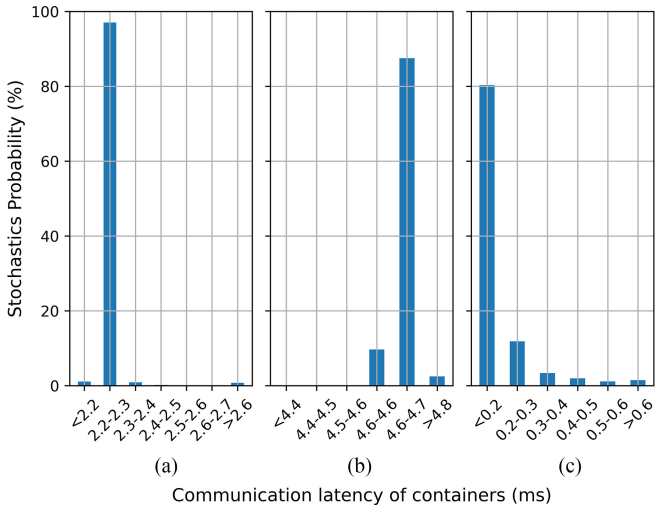

Fig. 11. Communication delays between containers: (a) within the same cloud cluster; (b) in different clusters; and (c) within an edge system.

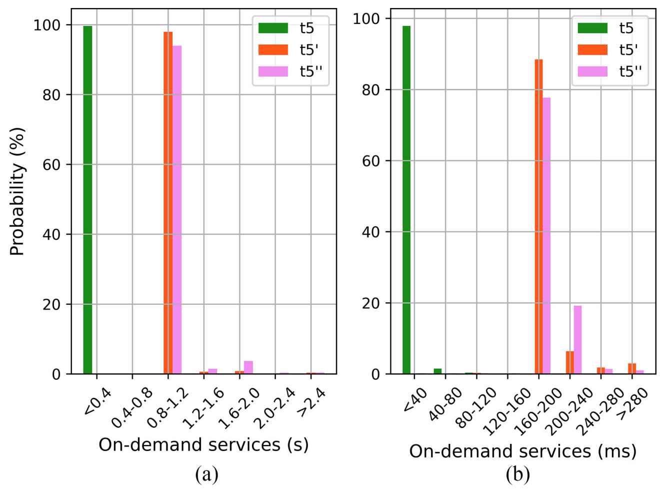

Fig. 12. Communication delays of on-demand services.

TABLE III
PARAMETERS OF THE CONTAINERS

	Container with customized protocol	Container with OpenADR stack
Image size of container (MB)	5	110
Peak running memory (MB)	<1	40

in Figs. 10 and 11, respectively. For t_3, the delay for the cloud deployment is approximately 0.4–0.8 s, and that for edge deployment is mainly about 60–140 ms. For t_4, the communication delay is mainly less than 2.6 ms when the containers are deployed in the same computing cluster, less than 4.8 ms when they are in computing clusters in different cities, and less than 0.6 ms for the edge deployment on the same desktop.

Using the proposed architecture, most iterative communications are between dedicated containers, as illustrated in Fig. 6. The communication time is, thus, much shorter with than without the proposed architecture; in the latter case, user-side smart meters must participate in each peer-to-peer negotiation.

3) Results of Scenario 3: Fig. 12 shows the communication delays in on-demand services. For the proposed architecture with the cloud deployment, in the case that the user requested data were available on the dedicated containers, the on-demand communication delay t_5, is less than 400 ms (less than 40 ms more specifically), as shown in Fig. 12(a). On the other hand, if the requested data were not directly available, the container had to obtain the request data from the user-side smart meter, leading to a higher on-demand communication delay (t'_5) in the range of 0.8–2.4 s. Without the proposed architecture, the on-demand communication delay t''_5, is similar to that of t'_5.

For the edge deployment shown in Fig. 12(b), t_5, t'_5, and t''_5 are mainly less than 280 ms, satisfying the latency and reliability requirements of on-demand services. With the proposed architecture, the communication delay when the requested data were available on the dedicated container t_5, is shorter than that when the requested data had to be obtained from the smart meter t'_5. Without the proposed architecture, the communication delay, t''_5 is similar to the latter case t'_5.

V. Discussion

This section focuses on discussing potential limitations and the business model, facilitating the practical application of the proposed architecture to a smart grid.

A. Discussion on Potential Limitations

The proposed architecture relies on virtual containers to minimize the upgrading requirements of user-side smart meters. However, it is not cost-free to deploy virtual containers on a large scale, on either the cloud or an edge system. Table III lists the parameters of some containers, including one with a customized protocol in C language and one with a full OpenADR stack in Python. These two implementations differ in terms of storage and running requirements. The OpenADR stack implemented here requires the loading of several Python packages to the container. This is generally not problematic, but may limit large-scale deployment of the proposed architecture to some extent. For example, consider a desktop computer with 500 GB of storage and 32 GB of memory. This can accommodate over 30 000 containers with C-based customized protocols, but only 800 containers with the Python-based OpenADR stack. It will be necessary to optimize container implementation to facilitate large-scale deployment and reduce deployment costs in the cloud or on an edge. It will be important to appropriately schedule and allocate resources among containers deployed in the cloud or on an edge.

In the proposed architecture, application-oriented communications and services are performed by dedicated containers in different locations, allied to user-side smart meters. Given the communication delays between the containers and meters, it is not currently possible to support time-critical applications, such as those with hard deadlines. Such applications must still be implemented on user-side smart meters, and can be moved to containers only when new communication technology reduces the latency between containers and meters; the ultrareliable low-latency communication provided by 5G may eliminate the problem.

When cloud deployment is used, the proposed architecture may encounter difficulties with existing cloud providers, including vendor lock-ins, security, and real-time issues. A private cloud is recommended to host the containers.

B. Discussion on the Business Model

The primary stakeholders to deploy the proposed architecture in practical use are the utilities and the customers. First of all, it is the utilities' responsibility to design the business model and conduct investment-payback analysis. It should also be noted that the proposed architecture is not cost-free after deployment. Long-term operation costs apply to the cloud or edge systems hosting virtual containers, as discussed in Section V.A.

Nevertheless, the proposed architecture creates values for the utilities from the following aspects:

1) it reduces the upgrading requirements of user-side smart meters, saving the cost of human resources to upgrade user-side meters and fix upgrading issues;
2) it enables easier and fast deployment of emerging smart grid applications, facilitating the more efficient operation of the smart grid and allowing new ways to make profits for the utilities; and
3) it provides better services to the customers, improving the competitiveness of utilities.

Although sophisticated economic analysis has not been conducted, it can be expected that both the utilities and customers will benefit from the new features introduced by the proposed architecture.

VI. Conclusion

This article presented a container-driven service architecture to minimize the upgrading requirements of user-side smart meters in distribution grids. The system model and layered architecture, and an example of an upgrading procedure, were illustrated to minimize the upgrading requirements of user-side smart meters. To demonstrate the effectiveness of the proposed architecture, two experimental test beds were developed in the cloud and on an edge, and used to verify the performance of the proposed architecture in terms of three types of communication in which user-side smart meters are currently heavily involved: communication with application servers, iterative communication with peers, and on-demand communication with users. The results showed that the proposed architecture satisfied the time requirements of smart grid applications related to user-side smart meters.

In future, it will be necessary to optimize container development by reducing the image size and peak running memory, to facilitate large-scale deployment. It will also be important to determine how the architecture could support time-critical smart grid applications. If cloud deployment is adopted, difficulties with existing cloud providers may require attention. It is also necessary for the utility suppliers to carefully design the business model and conduct investment-payback analysis in order to deploy the proposed architecture in practical smart grid applications.

References

[1] Y. M. Ding, S. H. Hong, and X. H. Li, "A demand response energy management scheme for industrial facilities in smart grid," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2257–2269, Nov. 2014.

[2] N. Shaukat *et al.*, "A survey on consumers empowerment, communication technologies, and renewable generation penetration within smart grid," *Renewable Sustain. Energy Rev.*, vol. 81, pp. 1453–1475, 2018.

[3] Q. Sun *et al.*, "A comprehensive review of smart energy meters in intelligent energy networks," *IEEE Internet Things J.*, vol. 3, no. 4, pp. 464–479, Aug. 2016.

[4] D. B. Avancini, J. J. Rodrigues, S. G. Martins, R. A. Rabêlo, J. Al-Muhtadi, and P. Solic, "Energy meters evolution in smart grids: A review," *J. Cleaner Prod.*, vol. 217, pp. 702–715, 2019.

[5] Y. Ding, Y.-C. Tian, X. Li, Y. Mishra, G. Ledwich, and C. Zhou, "Constrained broadcast with minimized latency in neighborhood area networks of smart grid," *IEEE Trans. Ind. Informat.*, vol. 16, no. 1, pp. 309–318, Jan. 2020.

[6] S. Zhou and M. A. Brown, "Smart meter deployment in europe: A comparative case study on the impacts of national policy schemes," *J. Cleaner Prod.*, vol. 144, pp. 22–32, 2017.

[7] B. K. Sovacool, P. Kivimaa, S. Hielscher, and K. Jenkins, "Vulnerability and resistance in the United Kingdom's smart meter transition," *Energy Policy*, vol. 109, pp. 767–781, 2017.

[8] H. Farhangi, "The path of the smart grid," *IEEE Power Energy Mag.*, vol. 8, no. 1, pp. 18–28, Jan./Feb. 2010.

[9] L. M. Camarinha-Matos, "Collaborative smart grids – A survey on trends," *Renewable Sustain. Energy Rev.*, vol. 65, pp. 283–294, 2016.

[10] J. Lloret, J. Tomas, A. Canovas, and L. Parra, "An integrated IoT architecture for smart metering," *IEEE Commun. Mag.*, vol. 54, no. 12, pp. 50–57, Dec. 2016.

[11] M. Valocchi, J. Juliano, and A. Schurr, "Switching perspectives: Creating new business models for a changing world of energy," in *Smart Grid Appl. Develop.*, London, U.K., 2014, pp. 165–182.

[12] M. Dong, P. C. Meira, W. Xu, and W. Freitas, "An event window based load monitoring technique for smart meters," *IEEE Trans. Smart Grid*, vol. 3, no. 2, pp. 787–796, Jun. 2012.

[13] F. Benzi, N. Anglani, E. Bassi, and L. Frosini, "Electricity smart meters interfacing the households," *IEEE Trans. Ind. Electron.*, vol. 58, no. 10, pp. 4487–4494, Oct. 2011.

[14] M. Eissa, "Developing incentive demand response with commercial energy management system (cems) based on diffusion model, smart meters and new communication protocol," *Appl. Energy*, vol. 236, pp. 273–292, 2019.

[15] P. Samadi, A.-H. Mohsenian-Rad, R. Schober, V. W. Wong, and J. Jatskevich, "Optimal real-time pricing algorithm based on utility maximization for smart grid," in *Proc. 1st IEEE Int. Conf. Smart Grid Commun.*, 2010, pp. 415–420.

[16] C. Feng, Y. Wang, K. Zheng, and Q. Chen, "Smart meter data-driven customizing price design for retailers," *IEEE Trans. Smart Grid*, vol. 11, no. 3, pp. 2043–2054, May 2020.

[17] A. Paudel, K. Chaudhari, C. Long, and H. B. Gooi, "Peer-to-peer energy trading in a prosumer-based community microgrid: A game-theoretic model," *IEEE Trans. Ind. Electron.*, vol. 66, no. 8, pp. 6087–6097, Aug. 2019.

[18] S. Chakraborty and S. Das, "Application of smart meters in high impedance fault detection on distribution systems," *IEEE Trans. Smart Grid*, vol. 10, no. 3, pp. 3465–3473, May 2019.

[19] F. C. Trindade and W. Freitas, "Low voltage zones to support fault location in distribution systems with smart meters," *IEEE Trans. Smart Grid*, vol. 8, no. 6, pp. 2765–2774, Nov. 2017.

[20] Y. Jiang, C.-C. Liu, M. Diedesch, E. Lee, and A. K. Srivastava, "Outage management of distribution systems incorporating information from smart meters," *IEEE Trans. Power Syst.*, vol. 31, no. 5, pp. 4144–4154, Sep. 2016.

[21] F. A. Borges, R. A. Fernandes, I. N. Silva, and C. B. Silva, "Feature extraction and power quality disturbances classification using smart meters signals," *IEEE Trans. Ind. Informat.*, vol. 12, no. 2, pp. 824–833, Apr. 2016.

[22] M. M. Albu, M. Sănduleac, and C. Stănescu, "Syncretic use of smart meters for power quality monitoring in emerging networks," *IEEE Trans. Smart Grid*, vol. 8, no. 1, pp. 485–492, Jan. 2017.

[23] I. Parvez, M. Aghili, A. I. Sarwat, S. Rahman, and F. Alam, "Online power quality disturbance detection by support vector machine in smart meter," *J. Modern Power Syst. Clean Energy*, vol. 7, no. 5, pp. 1328–1339, 2019.

[24] W. L. R. Junior, F. A. Borges, A. F. d. S. Veloso, R. de AL Rabêlo, and J. J. Rodrigues, "Low voltage smart meter for monitoring of power quality disturbances applied in smart grid," *Measurement*, vol. 147, 2019, Art. no. 106890.

[25] D. Alahakoon and X. Yu, "Smart electricity meter data intelligence for future energy systems: A survey," *IEEE Trans. Ind. Informat.*, vol. 12, no. 1, pp. 425–436, Feb. 2016.

[26] H. Ahmadi and J. R. Martı, "Load decomposition at smart meters level using eigenloads approach," *IEEE Trans. Power Syst.*, vol. 30, no. 6, pp. 3425–3436, Nov. 2015.

[27] H.-H. Chang, L.-S. Lin, N. Chen, and W.-J. Lee, "Particle-swarm-optimization-based nonintrusive demand monitoring and load identification in smart meters," *IEEE Trans. Ind. Appl.*, vol. 49, no. 5, pp. 2229–2236, Sep./Oct 2013.

[28] J. P. Gouveia and J. Seixas, "Unraveling electricity consumption profiles in households through clusters: Combining smart meters and door-to-door surveys," *Energy Buildings*, vol. 116, pp. 666–676, 2016.

[29] M. Jin, R. Jia, and C. J. Spanos, "Virtual occupancy sensing: Using smart meters to indicate your presence," *IEEE Trans. Mobile Comput.*, vol. 16, no. 11, pp. 3264–3277, Nov. 2017.

[30] L. Yang, X. Chen, J. Zhang, and H. V. Poor, "Cost-effective and privacy-preserving energy management for smart meters," *IEEE Trans. Smart Grid*, vol. 6, no. 1, pp. 486–495, Jan. 2015.

[31] Z. Zhang, Z. Qin, L. Zhu, J. Weng, and K. Ren, "Cost-friendly differential privacy for smart meters: Exploiting the dual roles of the noise," *IEEE Trans. Smart Grid*, vol. 8, no. 2, pp. 619–626, Mar. 2017.

[32] F. Farokhi and H. Sandberg, "Fisher information as a measure of privacy: Preserving privacy of households with smart meters using batteries," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 4726–4734, Sep. 2018.

[33] D. Chen, S. Kalra, D. Irwin, P. Shenoy, and J. Albrecht, "Preventing occupancy detection from smart meters," *IEEE Trans. Smart Grid*, vol. 6, no. 5, pp. 2426–2434, Sep. 2015.

[34] M. M. Fouda, Z. M. Fadlullah, N. Kato, R. Lu, and X. S. Shen, "A lightweight message authentication scheme for smart grid communications," *IEEE Trans. Smart grid*, vol. 2, no. 4, pp. 675–685, Dec. 2011.

[35] D. Abbasinezhad-Mood and M. Nikooghadam, "An ultra-lightweight and secure scheme for communications of smart meters and neighborhood gateways by utilization of an arm cortex-m microcontroller," *IEEE Trans. Smart Grid*, vol. 9, no. 6, pp. 6194–6205, Nov. 2018.

[36] D. Abbasinezhad-Mood, A. Ostad-Sharif, M. Nikooghadam, and S. M. Mazinani, "A secure and efficient key establishment scheme for communications of smart meters and service providers in smart grid," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 1495–1502, Mar. 2020.

[37] D. Abbasinezhad-Mood and M. Nikooghadam, "Efficient anonymous password-authenticated key exchange protocol to read isolated smart meters by utilization of extended chebyshev chaotic maps," *IEEE Trans. Ind. Informat.*, vol. 14, no. 11, pp. 4815–4828, Nov. 2018.

[38] D. Abbasinezhad-Mood, A. Ostad-Sharif, and M. Nikooghadam, "Novel anonymous key establishment protocol for isolated smart meters," *IEEE Trans. Ind. Electron.*, vol. 67, no. 4, pp. 2844–2851, Apr. 2020.

[39] C. Pahl, A. Brogi, J. Soldani, and P. Jamshidi, "Cloud container technologies: A state-of-the-art review," *IEEE Trans. Cloud Comput.*, vol. 7, no. 3, pp. 677–692, Jul.–Sep. 2019.

[40] Y. Li, X. Cheng, Y. Cao, D. Wang, and L. Yang, "Smart choice for the smart grid: Narrowband internet of things (NB-IoT)," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1505–1515, Jun. 2018.

Li Liu received the B.Eng. degree in metallurgical engineering from the Advanced Engineering School of University of Science and Technology Beijing, Beijing, China, in 2018. He is currently working toward the master's degree in computer science and engineering with the Tianjin University of Technology, Tianjin, China.

His research interests include Internet of Things and smart grid communications.

Yuemin Ding (Member, IEEE) received the doctoral degree in electronic systems engineering from the Department of Electronic Systems Engineering, Hanyang University, Ansan, South Korea, in 2014.

He was an Associate Professor with the School of Computer Science and Engineering, Tianjin University of Technology, Tianjin, China, from 2015 to 2019. From 2017 to 2018, he was also a Visiting Fellow with the Department of Electrical Engineering and Computer Science, Queensland University of Technology, Brisbane, Australia. Since August 2019, he has been with the Department of Energy and Process Engineering, Norwegian University of Technology, Trondheim, Norway. His research interests include Internet of Things (IoT), smart grid communications, user-side energy management, and building digitalization.

Xiaohui Li received the Ph.D. degree in control science and engineering from the Wuhan University of Science and Technology, Wuhan, China, in 2009.

She is currently a Professor with the School of Information Science and Engineering, Wuhan University of Science and Technology. Her research interests include automated demand response in smart grid, home/building automation networks, wireless sensor networks, and routing in complex networks.

Huaming Wu (Member, IEEE) received the B.E. and M.S. degrees in electrical engineering from the Harbin Institute of Technology, Harbin, China, in 2009 and 2011, respectively, and the Ph.D. degree (with the highest hons.) in computer science from Freie Universität Berlin, Berlin, Germany, in 2015.

He is currently an Associate Professor with the Center for Applied Mathematics, Tianjin University, China. His research interests include Internet of things, wireless and mobile network systems, edge/cloud computing, deep learning, and complex networks.

Lantao Xing (Member, IEEE) received the B.Eng. degree in automation from the China University of Petroleum (East China), Dongying, China, in 2013, and the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2018.

He was a Research Fellow with the School of Computer Science, Queensland University of Technology, Australia. He is currently a Presidential Postdoctoral Fellow with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. His research interests include nonlinear system control, event-triggered and quantized control, and distributed control with applications in smart grid.