This is a repository copy of *An integrated architecture for software defined and virtualized radio access networks with fog computing*.

White Rose Research Online URL for this paper:
http://eprints.whiterose.ac.uk/104882/

Version: Accepted Version

Article:

Liang, K., Zhao, L., Chu, X. orcid.org/0000-0003-1863-6149 et al. (1 more author) (2017) An integrated architecture for software defined and virtualized radio access networks with fog computing. IEEE Network, 31 (1). pp. 80-87. ISSN 0890-8044

https://doi.org/10.1109/MNET.2017.1600027NM

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other users, including reprinting/ republishing this material for advertising or promotional purposes, creating new collective works for resale or redistribution to servers or lists, or reuse of any copyrighted components of this work in other works.

Reuse

Unless indicated otherwise, fulltext items are protected by copyright with all rights reserved. The copyright exception in section 29 of the Copyright, Designs and Patents Act 1988 allows the making of a single copy solely for the purpose of non-commercial research or private study within the limits of fair dealing. The publisher or other rights-holder may allow further reproduction and re-use of this version - refer to the White Rose Research Online record for this item. Where records identify the publisher as the copyright holder, users can verify any specific terms of use on the publisher's website.

Takedown

If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk/

AN INTEGRATED ARCHITECTURE FOR SOFTWARE DEFINED AND VIRTUALIZED RADIO ACCESS NETWORKS WITH FOG COMPUTING

Kai Liang, Liqiang Zhao, *Member, IEEE*, Xiaoli Chu, *Member, IEEE*, and
Hsiao-Hwa Chen, *Fellow, IEEE*

[†]Corresponding Author's Address:
Hsiao-Hwa Chen
Department of Engineering Science
National Cheng Kung University
1 Da-Hsueh Road, Tainan City, 70101 Taiwan
Tel: +886-6-2757575 ext. 63320
Fax: +886-6-2766549
Email: `hshwchen@ieee.org`

Kai Liang (email: `klxdu@hotmail.com`) and Liqiang Zhao (email: `lqzhao@mail.xidian.edu.cn`) are with the State Key Laboratory of Integrated Service Networks, Xidian University, China. Xiaoli Chu (email: `x.chu@sheffield.ac.uk`) is with the Department of Electronic and Electrical Engineering, the University of Sheffield, UK. Hsiao-Hwa Chen (email: `hshwchen@ieee.org`) is with the Department of Engineering Science, National Cheng Kung University, Taiwan.

The paper was submitted on January 30, 2016, and revised on April 10, 2016, and accepted on July, 20, 2016. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Nei Kato.

Abstract

Today, billions of communication devices connecting to wireless networks impose serious challenges in network deployment, management, and data processing. Amongst all emerging technologies tackling these challenges, software defined networks (SDNs) decouple control plane from data plane to provide network programmability and virtualization, which can share network and radio resources among various applications. On the other hand, fog computing offloads computing services from cloud to the edge of networks, offering real time data services to nearby data terminals. In this paper, we present an integrated architecture for software defined and virtualized radio access networks with fog computing. We propose a design of software-as-a-service (SaaS) called OpenPipe, which enables network level virtualizaiton. To integrate SDNs and network virtualization with fog computing, we adopt a hybrid control model with two hierarchical control levels, where a SDN controller forms the higher level, and local controllers comprise the lower level. Typical user cases of the proposed network architecture are validated through laboratory demonstrations.

Index Terms

Software defined network; Network virtualization; Fog computing; OpenPipe; Hybrid control model.

I. INTRODUCTION

Wireless networks experienced a rapid evolution in the past decade, radically affecting people's everyday life. However, with the current architecture of wireless networks, configuration and maintenance of wireless networks and devices with diverse protocol stacks and vendor-specific interfaces have become extremely complex and expensive. Moreover, there is a lack of openness, flexibility, and scalability in the current wireless networks [1]. Today's networks even using cloud data centers for centralized provisioning of data processing, management, and services still suffer a fairly low network resource utilization and long delays.

New technologies, such as software defined networks (SDNs) [2], network virtualization [3], and fog computing [4], have emerged to tackle these problems. SDNs detach control plane from data plane. Network virtualization enables sharing of network resources (such as storage, computing, and services), radio resources (such as spectrum and time slots), and network elements among different applications. Fog computing moves a majority of computation tasks and services from the cloud to the edge of networks, relieving the overloaded cloud data centers.

So far, the above three technologies have largely been studied separately in the literature. In [7], the authors studied SDNs for enterprise wireless local area networks (WLANs). In [8], a SDN called MobileFlow was proposed for mobile core networks. SoftRAN abstracts a cluster of base stations (BSs) as a virtual BS consisting of a central controller and radio elements [9]. The cloud radio access network (CRAN), as an example on the radio access network (RAN) side to achieve SoftRAN, decouples baseband processing from radio frequency (RF) transmission of remote radio heads (RRHs) and centralizes it in a cloud-based pool of baseband units (BBUs) [10]. In [11], the authors provided a device-to-device communication based algorithm in software defined multi-tier long term evolution (LTE) networks. The research on fog computing is in its infancy stage with most efforts to specify definitions, architecture, and user cases. In [4], fog computing was compared with cloud computing in the context of Internet of things (IoTs). In [14], the authors proposed an architecture of SDN-based fog computing for vehicular ad hoc networks. In [15], a green cloudlet network powered by hybrid energy sources was proposed, supporting seamless services with low latency. More efforts are needed to explore the synergy in combining these three novel technologies for better RAN performance and user experience.

In this paper, we present an integrated architecture for software defined and virtualized RANs (SDVRANs) with fog computing. The main contributions of this paper include:

- We present a network architecture to integrate fog computing with a SDVRAN by implementing a hybrid control model, an extended OpenFlow (exOF) protocol, and network level virtualization.
- We propose a network example of software-as-a-service (SaaS) called OpenPipe, which enables network level virtualizaiton and user control of network operation with network management applications.
- We propose a hybrid control model with two control levels to support fog computing in SDVRANs.

The rest of the paper can be outlined as follows. In Sections II and III, we review the basic principles of SDNs, network virtualization, and fog computing. In Section IV, we present the integrated network architecture. A laboratory demonstration and user cases are presented in Section V, followed by the conclusion of this article.

II. SDNs and Network Virtualization

A typical SDN architecture consists of three logical layers, i.e., infrastructure layer, control layer, and application layer [2]. SDN provides application programming interfaces (APIs) to facilitate programmability of network operations. The northbound API connects the control layer and the application layer. The southbound API defines the way that the controller interacts with the infrastructure layer. OpenFlow [5] is the most popular protocol promoted by the Open Networking Foundation (ONF). It enables applications to program the forwarding switches' (or OpenFlow switches') flow tables, each of which is a list of flow entities. A flow entity consists of match fields, counters, and a set of instructions to match to the incoming packets, collect statistics regarding the current flow, and handle a matched packet. However, the OpenFlow protocol was proposed originally for wired networks and is not suitable for wireless networks, because of the distinctive properties of wireless channels, such as time variations, attenuation, mobility, and broadcast nature.

Network function virtualization (NFV) [6] leverages the virtualization technology to abstract various network devices onto general-purpose high volume servers, switches, and storage, which are normally deployed in the operator's data center. Using SDN functions running in the data center, we know that NFV changes network architectures and the ways the networks operate, as software based network functions can be moved to or instantiated in the required places in the networks, without the installation of new equipments. According to different requirements, virtual resources can be classified into different levels [3], such as spectrum level slicing, where the spectra is virtualized by multiplexing techniques; infrastructure level slicing, i.e., virtualization of physical network elements; and network level slicing, where all the entities of the network are sliced to form a virtual network.

III. FOG COMPUTING

Fog computing was defined as an extension of cloud computing from the core to the edges of the network by Cisco. The distributed intelligence fits to the cases of frequent services, real time applications, and mobile big data analysis. The necessity for the emerging fog computing is three folds, including

- Scalability: Smart devices are pushing the current networks to their limits. The scalability of network monitoring and management remains to be a big challenge to the current centralized network architecture.
- Real time data analytics: Large delays will make the cloud servers easily overloaded and affect the user experience of latency-sensitive applications. Fog computing can reduce the data processing delays by offloading computation tasks to the nearest fog nodes.

- Saving network resources: Fog nodes undertake most of data processing tasks at the remote sites, and only forward necessary information to the cloud, thus reducing the use of network resources and the traffic burden on the cloud.

The integration of SDN, virtualization, and fog computing benefits the current networks. With fog computing, the computation and transmission burdens on the controller of the SDN and the transmission backbone can be reduced, respectively. Meanwhile, the fog network can be implemented by SDN and virtualization to reduce the management and configuration costs and to improve the scalability and resource utilization. However, there are some challenges to be addressed to integrate the fog computing into SDNs. The SDN with fog computing is a distributed SDN, which should meet the requirements (latency, storage, energy consumption, etc.) of fog computing. Besides, the cooperation schemes among various controllers (i.e., the interactions between cloud SDN controllers and fog controllers) should be carefully designed. Challenges in operation strategies include how to make a computation offloading or data storage decision between fog and cloud, and how to achieve the cooperation between fog and cloud computing. In the following section, we propose a network architecture that supports the integration of fog computing into SDNs, the cooperation among multiple controllers, and the operation of SDVRANs with fog computing.

IV. AN INTEGRATED NETWORK ARCHITECTURE

A. System Design

Fig. 1 shows the proposed network architecture. To better support wireless networks, we propose an exOF protocol (see Section IV-C for more details), which is backward compatible with the standard OpenFlow protocol) and can be implemented in wireless forwarding devices (including access points (APs) and base stations (BSs)), switches in the transport backbone, and the communication channels between the controller and forwarding devices in the proposed network. Note that, without losing the generality, we use a WLAN scenario to describe the proposed architecture for the sake of simplicity.

The fundamental elements of this proposed network architecture include:

- The SDN controller, which has the global intelligence of the whole network and can control network devices.

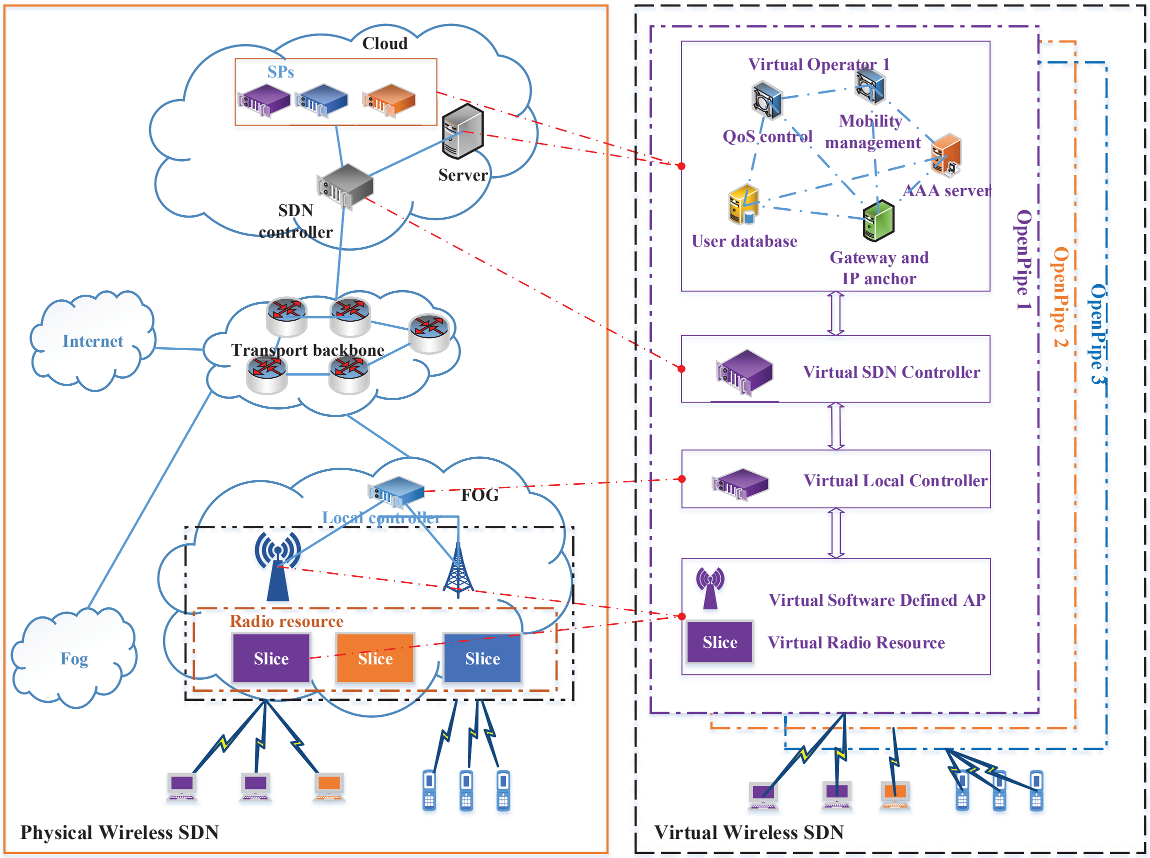


Fig. 1. The architecture of software defined and virtualized RANs with fog computing.

- Local controllers, which are exOF-enabled and controlled by the SDN controller. They are responsible for forwarding data, and fulfilling the demands of real-time and latency-sensitive applications.
- The exOF enabled forwarding devices, including the APs, BSs, and switches in the transport backbone.
- Users, including mobile phones, laptops, and other wireless devices.
- Cloud servers, which store user data and run network functions, such as mobility management, QoS control, authentication, authorization, and accounting (AAA).
- Service providers (SPs), which provide services to their subscribers.

B. A Hybrid Control Model

A SDN controller can be categorized into three control models, including centralized model (such as Odin [7]), distributed model (such as Flowvisor [12]), and hybrid model (such as

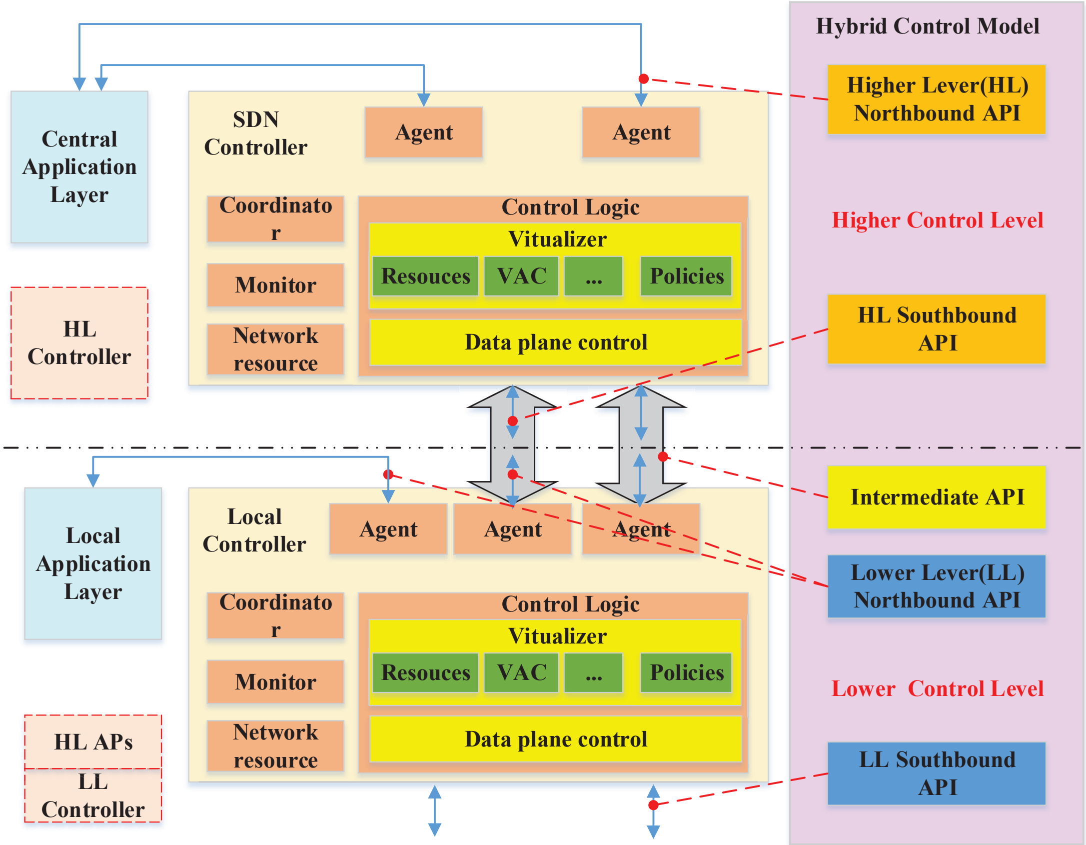


Fig. 2. A hybrid control model.

Kandoo [13]). In order to integrate fog computing, the proposed architecture adopts a hybrid control model, where the SDN controller and the local controllers form a two-level hierarchical control plane of the network. Fig. 2 shows the high level hybrid control model with the internal architecture of SDN and local controllers and the APIs, which define the interactions between a SDN controller and a local controller. The SDN controller (in the higher level) defines specific policies and sends these policies to the local controllers (in the lower level) to instruct how specific applications or behaviours should be processed by local controllers based on their local information. The control plane functions can be programmed directly in these controllers or in the SDN controller as applications. From the higher control level's perspective, the SDN controller and local controllers can be considered as the controller and agents (or switches). From the lower control level's perspective, they can be viewed as applications (or clients) and controllers (or servers), respectively. An intermediate interface is needed to connect two control levels, which will be described in the next subsection.

The benefits of implementing a hybrid control model lie in two folds. First, the SDN controller and local controllers are located in the cloud and the fog networks, respectively. A local controller working together with a set of exOF-enabled devices forms a fog and behaves as a sub-SDN, which can run simple applications and can store a small amount of data. Apart from simplifying the management of the fog, the sub-SDN reduces the burden on the transmission backbone and the SDN controller in the cloud. Moreover, local controllers in fog sub-SDNs can be managed by the SDN controller, forming a higher level SDN. Such a hybrid control model simplifies the management and configuration, and improves the scalability of the proposed network.

C. OpenPipe

We propose a virtual resource chain (or network level virtualization), called OpenPipe. The inner architecture of OpenPipe is transparent to its users. The interface architecture of OpenPipe is shown in Fig. 3, which consists of three APIs and three virtual layers. The three APIs are:

- A southbound API between the infrastructure layer and the control layer. We propose an exOF protocol to meet the requirements of wireless SDNs. Two major changes have been made in the exOF as compared to the original OpenFlow. First, the matching field and action sets of the flow entity have been expanded through OpenFlow Extensible Match (OXM) mechanism by adding wireless type, parameters and the corresponding actions in oxm_class, oxm_field and action sets, respectively. Second, controller/AP messages have been expanded for the connections between the wireless controller and APs through the following steps. After expanding the type of OpenFlow multipart messages (including wireless port, CPU, storage, wireless channel status, and AP status), we have added the description of available resources, wireless channel status, and AP status in the multipart messages.
- An intermediate interface between the SDN controller and local controllers. From the viewpoint of the higher control level, this interface can be viewed as a southbound API, while it acts as a northbound API from the viewpoint of the lower control level. Since the northbound API suffers a lack of standards and can be created using any programming languages, we use the exOF protocol as the intermediate API for the sake of simplicity.
- A northbound API between the control layer and the application layer. The operators can implement programmed applications and communicate with the SDN controller through

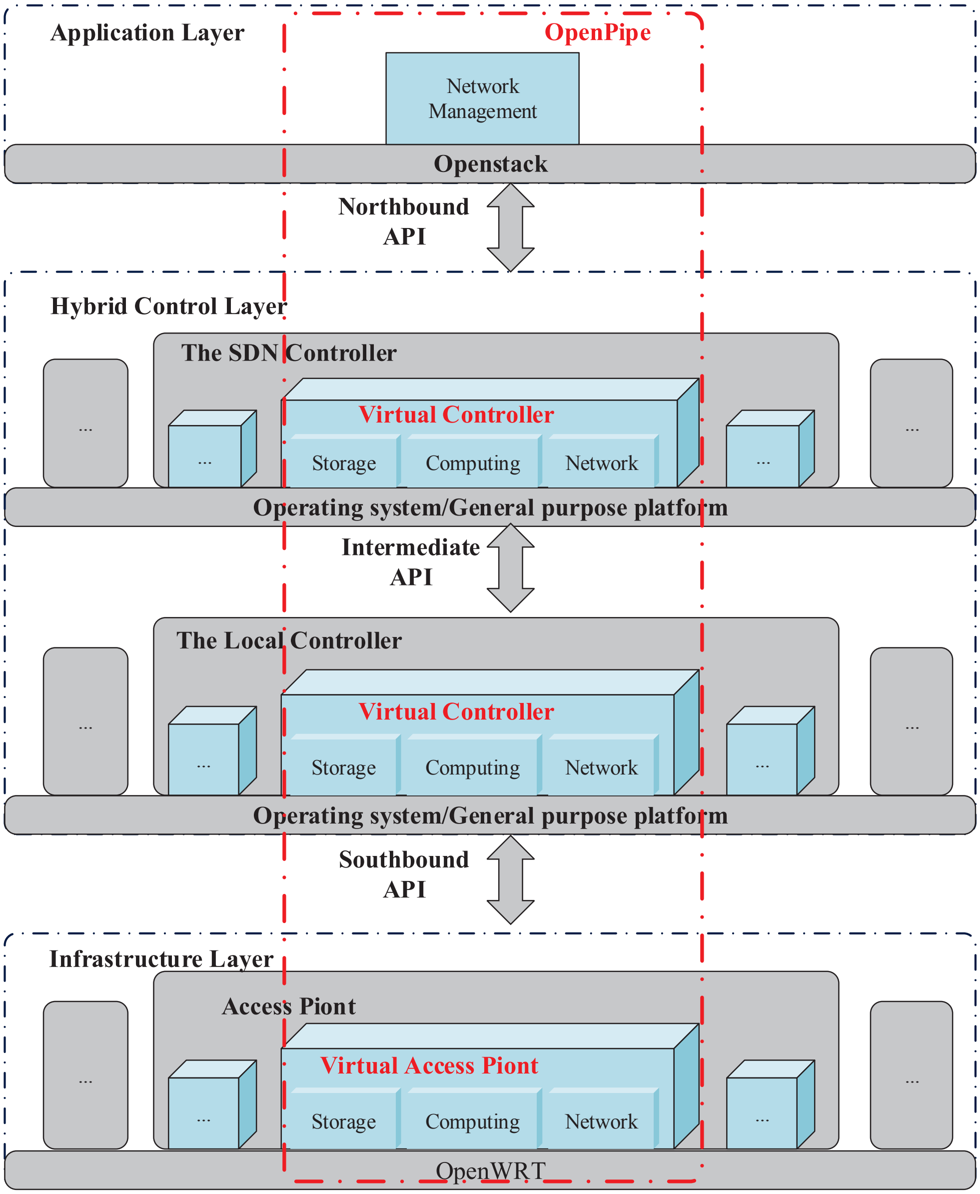


Fig. 3. The interface architecture.

northbound APIs, thereby sharing the network resources, adaptively allocating resources according to specific rules, and accessing the underlying forwarding devices.

The three virtual layers of OpenPipe are infrastructure layer, hybrid control layer, and application layer. Based on network virtualization techniques, the network resources (like forwarding devices) and radio resources can be shared with different applications.

The infrastructure elements include APs, BSs, and switches in the transport backbone. Physical devices and network resources in the infrastructure layer are abstracted and sliced into several

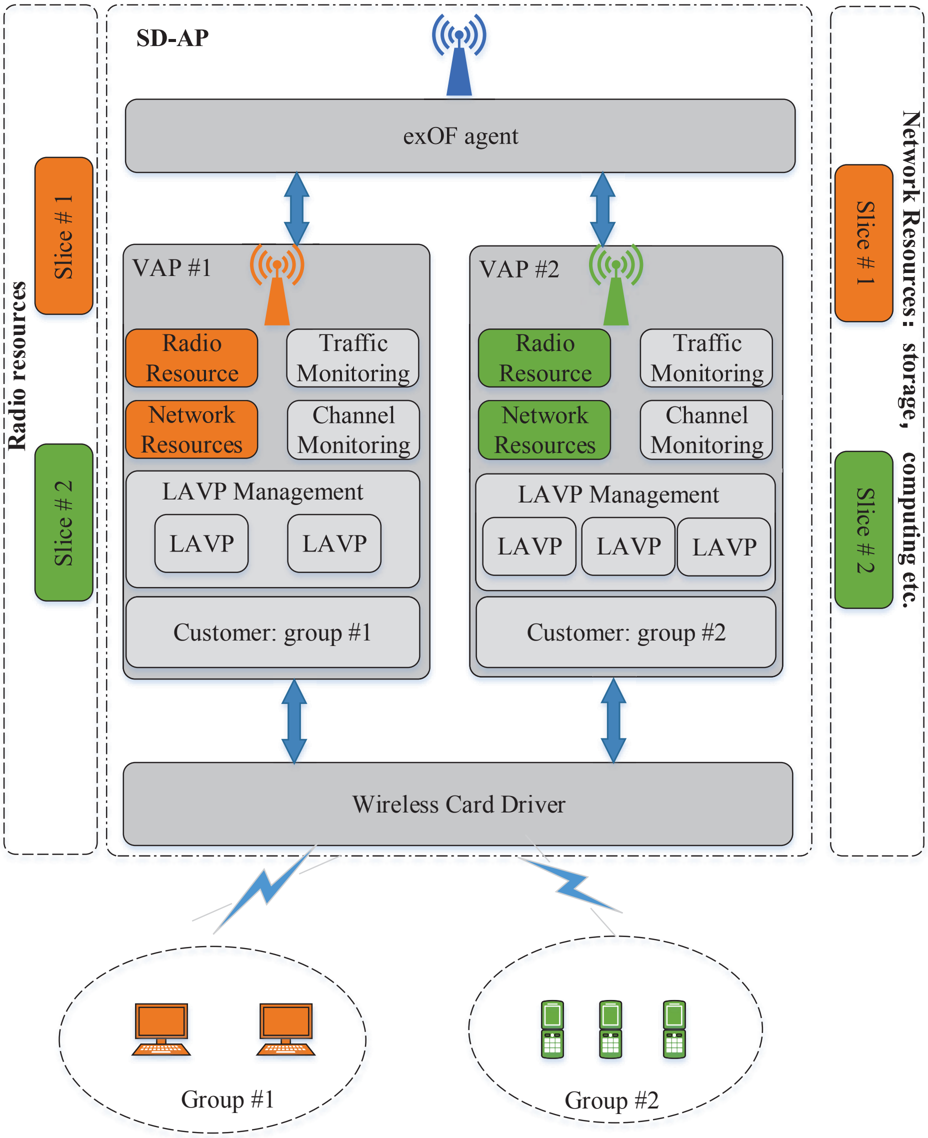


Fig. 4. The software defined AP.

isolated virtual resources, and shared by multiple applications. Following the emerging concept of anything-as-a-service (XaaS) in cloud computing, we can view the virtual APs (VAPs) as an example of infrastructure-as-a-service (IaaS), which outfits each user or application with one or more VAPs running on a physical AP. Meanwhile, radio resources can also be sliced and allocated to different VAPs. Fig. 4 shows the structure of a software defined AP with the virtualization of infrastructure, network resources, and radio resources. Light VAPs (LVAPs) proposed in the Odin framework [7] are used in this paper to achieve seamless mobility and load balance.

The virtual controller (VC) is an example of platform-as-a-service (PaaS). This includes an environment for developing online applications that run on the physical SDN controller or local controllers. Users do not have to manage and control VCs and VAPs. Once applications are created, one or more VCs are formed automatically as needed to run all the applications.

The most significant advantage of OpenPipe is that it achieves full network level virtualization of the proposed SDN with a hybrid control model. A single OpenPipe is a universal wireless SDN and be transparent to users, as it includes all the virtual resources, such as virtual SDN controller, virtual local controllers, VAPs, virtual radio resources, and virtual cloud server, sliced from all the elements of the SDN. Each OpenPipe is isolated from the others and can be assigned to different applications. This enables multiple applications to be implemented and operated at the same time. It is particularly useful for testing experimental functions without affecting the normal network functions. Besides, OpenPipe is an example of software-as-a-service (SaaS). Therefore, user can access OpenPipe online whenever they need it through a pay-per-use revenue model, rather than leasing it for a long term. This facilitates the users who have limited financial resources and prefer to use OpenPipe in a cost-effective way, and benefits the OpenPipe providers by increasing the revenue from pay-per-use users.

D. Fog Computing

In the proposed network, local controllers make the decisions on whether to provide local services to end users or to forward data to the SDN controller, based on specific policies or rules established by and received from the SDN controller. In the following, we discuss the metrics that need to be considered when offloading services from the SDN controller to local controllers:

- Latency: Transmission, signal processing, and control signalling latencies will have significantly impact on some latency-sensitive applications, which require fast network response and service provisioning within their delay tolerance levels.
- Energy consumption: It includes not only transmission energy cost but also network equipment energy cost (such as signal processing, circuit energy loss, etc.) and affiliated costs (such as cooling systems).
- Overhead: Considering signalling overheads, applications with frequent transmissions of small data packets (such as traffic monitoring) should be provided locally wherever possible.

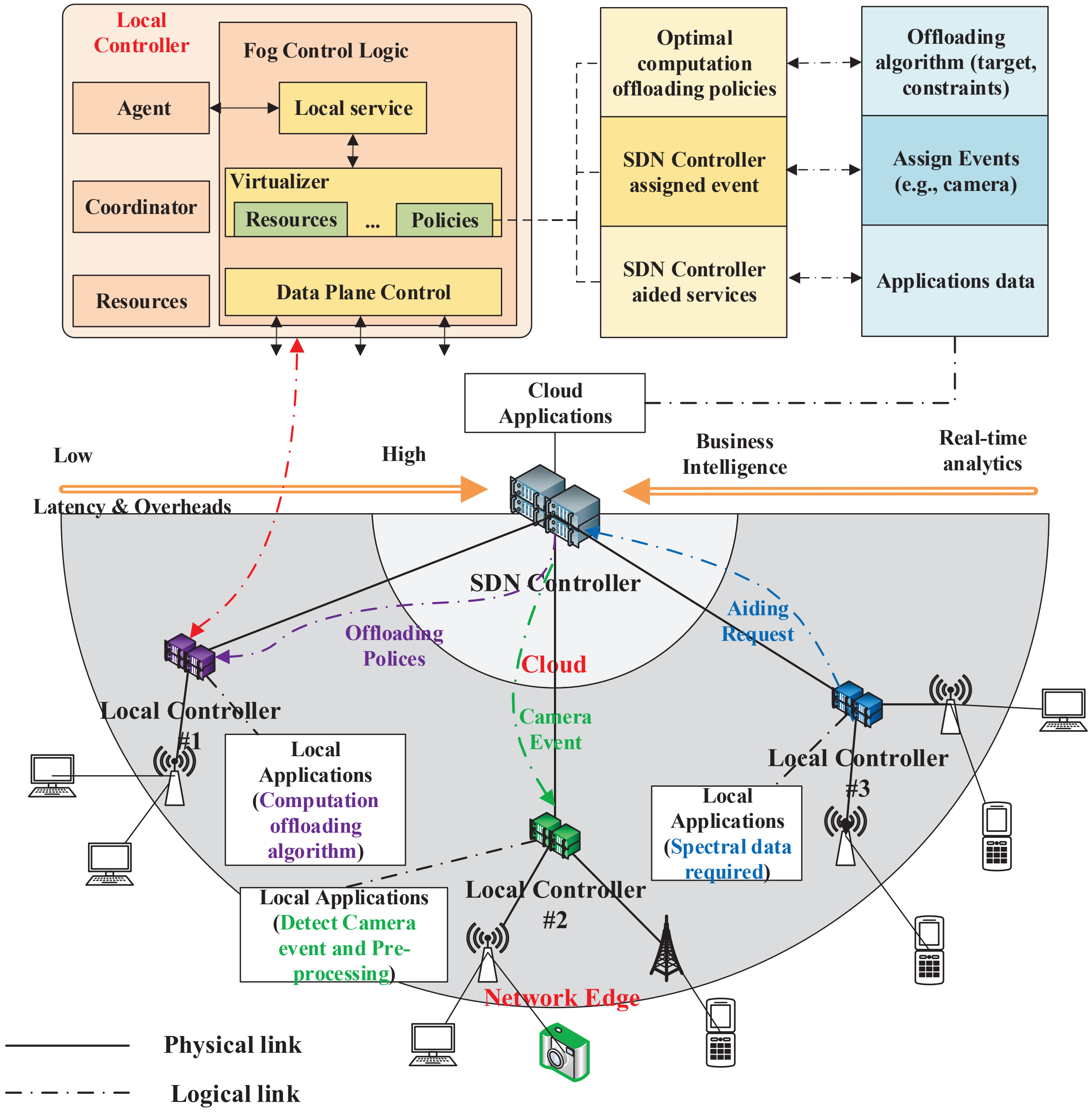


Fig. 5. SDNs with fog computing.

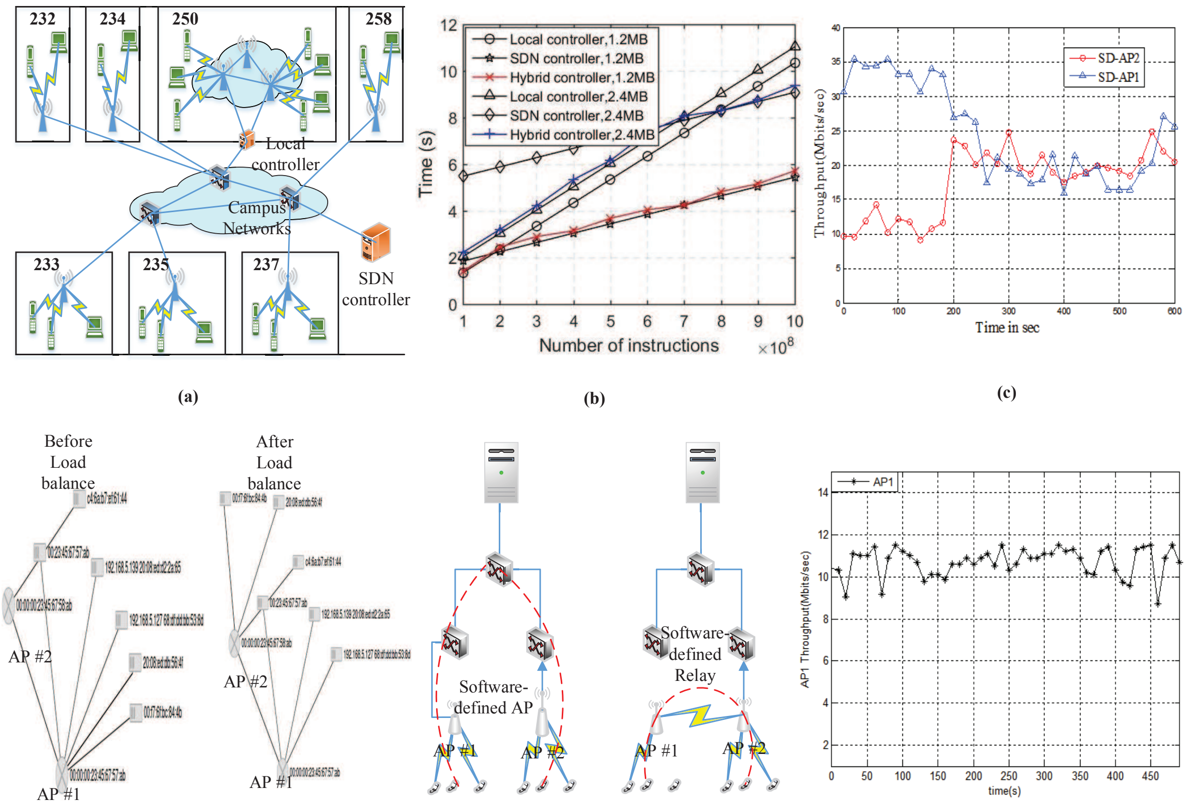
- Wireless connectivity: The performance of a wireless network will be affected when too many devices choose the same wireless channels to offload their computation tasks to local controllers.

Fig. 5 shows a high level architecture of a SDN with fog computing, consisting of a SDN controller located in the cloud and three local controllers deployed in the network. The SDN controller with a global view of the network manages local controllers, which have local views of the fog network. Local controllers control APs, BSs, and switches directly through installing flow-entities to these exOF-enabled forwarding devices. Local controllers may be considered as exOF-enabled forwarding devices by the SDN controller. The SDN controller can deliver flow-entities to the lower level controllers to control switches indirectly. In this case, the SDN controller views itself as the applications of lower level controllers. By registering itself as a lower level controller, the SDN controller can also directly control some switches for specific purposes. The SDN controller sends the control policies to local controllers, e.g., for offloading computation, assigning a specific event, or sending data required by local controllers. These user cases will be discussed in detail later.

E. User Cases

The proposed architecture can offer traditional services of SDN. Let each VAP serve one user, and each physical AP host multiple VAPs. First, load balance can be realized based on the collected network state information by shifting some VAPs from a high traffic AP to a low traffic AP. Second, when a handover event is triggered by a user, the mobility management application moves a number of VAPs from the serving AP to an AP that can provide the user with a better QoS without any additional message exchanges between the user and APs. Third, power management classifies APs into two types, namely, masters that are always on and slavers that are at the disposal of power management, and then formulates the problem as, e.g., throughput maximization of the network by optimizing the transmission power of APs.

The SDN controller and local controllers can provide services jointly. First, cooperative computation offloading can be provided. In Fig. 5, local controller 1 decides whether to provide local services or to offload computation to its higher level controller based on local system states and the computation offloading policies delivered from the SDN controller. For example, the policy can be formulated as an optimization problem aiming to minimize the total energy consumption under specific latency and physical resource constraints. The solution could be the optimal resource allocation and the optimal distribution of computation loads between local applications and higher level applications. Second, the SDN controller assigns an event (such as a camera event) to local controller 2, which creates a camera detecting application. Once the assigned event is received, local controller 2 pre-processes the event according to the SDN controller's policy with the help of local applications and then forwards the pre-precessed event to the SDN controller. Third, the local controllers can also send requests to the SDN controller



(a) (b) (c) (d) (e) (f)

Fig. 6. (a) Laboratory demonstration; (b) Computation offloading performance versus different control models; (c) Load balance performance; (d) User association; (e) Network topology management; (f) Network topology management throughput performance of AP 1 after relaying.

when some higher level application data, such as global topology, network energy consumption, and network revenue, are needed by local applications. In Fig. 5, local controller 3 can receive higher level application data after the SDN controller has received the requests from the lower level.

V. LABORATORY DEMONSTRATIONS

We implemented the proposed network architecture in our laboratory, where tower servers act as controllers, exOF components include switches, gateways, and APs, and users include laptops and mobile phones. The open SDN controller software, i.e., Floodlight (version 1.1), runs on the servers. The operation system, i.e., OpenWrt, runs in APs, and OpenvSwitch 2.3.2 was installed in these APs to support the exOF protocol based on OpenFlow 1.3. Fig. 6 (a) shows the architecture of our laboratory demonstration deployed across a number of rooms on the same floor, where a local controller was placed in Room 250, and the SDN controller was connected through the campus network.

Based on the collected network states, computation offloading, load balancing, power management, and QoS management can be carried out to offload computation services, to distribute loads across the network, to reduce the energy consumption, and to guarantee specific QoS levels through solving some optimization problems with certain objectives and constraints, respectively. For example, Fig. 6 (b) shows the computation offloading performance (event servicing time) for different control models. An event should be processed locally or at the SDN controllers. The event servicing time includes event transmission time and processing time for the single level controller. There is an additional offloading decision time at the local controller for the hybrid control model. The computing capability for an event at the local and the SDN controllers are set as 0.1×10^9 and 0.25×10^9 instructions per second, respectively. The computation quantity of offloading events (to be run on local or SDN controllers) ranges from 0.1×10^9 to 10^9 instructions. Two sizes of transmission data were considered: 1.2 Mbits and 2.4 Mbits. The hybrid model can estimate the servicing times of local controller and SDN controller, and adaptively choose the controller that requires less serving time to offload computation services, reducing the serving time of single level control model. For example, when the computation quantity is low (10^8 instructions), the SDN control spends a longer time on serving this event, and thus the hybrid model chooses local control to offload computation services. Fig. 6 (c) shows the performance of load balancing. The local controller observes that AP 1 serves five users and has a very high traffic loads, while AP 2 serves only one user with a relatively low traffic demand. Based on the collected network state information, the controller shifts some VAPs from the high traffic AP 1 to the low traffic AP 2. The demo network achieves load balancing at about 200 seconds. Before that, the throughput of AP 1 is more than three times of that of AP 2. In contrast, the figures for APs 1 and 2 remain at the same level (about 20 Mb/s). Fig. 6 (d) shows that before load balancing, AP 1 serves five users and AP 2 serves one user, while they each serves three users after load balancing. Fig. 6 (e) shows the application of network topology management. If the physical link from AP 1 to the gateway gets congested or interrupted, data of AP 1 cannot be transmitted without relaying through AP 2. In this case, the controller will send flow entities indicating that AP 2 can act as a relay node to forward the data of AP 1. Fig. 6 (f) shows the throughput of AP 1 with AP 2 acting as its relay to the gateway. In contrast, without relaying, no data can be transmitted to the gateway through AP 1.

VI. CONCLUSIONS

We have proposed an integrated architecture for software defined and virtualized RANs with fog computing, where SDN techniques were used to split up the control and data planes and provide network programmability. Network virtualization slices network resources and shares these sliced resources to diverse applications. Fog computing offloads computing services from the core network to the edges of the network. We presented an example of SaaS called OpenPipe, which can be viewed as network level virtualization. With OpenPipe, the underlying infrastructure becomes transparent to applications. To support fog computing in SDNs, a hybrid control model with two control levels was used, and the related control policies were discussed. Finally, we presented a lab demo to validate typical user cases of the proposed network architecture.

ACKNOWLEDGMENTS

This work was supported in part by National Natural Science Foundation of China (No. 61372070), Natural Science Basic Research Plan in Shaanxi Province of China (No. 2015JM6324), Ningbo Natural Science Foundation (2015A610117), Hong Kong, Macao and Taiwan Science & Technology Cooperation Program of China (No. 2015DFT10160), the 111 Project (No. B08038), Taiwan Ministry of Science and Technology (No. 104-2221-E-006-081-MY2), and Taiwan Industrial Technology Research Institute (No. M0-10309-6).

REFERENCES

[1] F. Granelli *et al*, "Software defined and virtualized wireless access in future wireless networks: scenarios and standards," *IEEE Commun. Mag.*, vol. 53, no. 6, 2015, pp. 25-34.

[2] B. A. A. Nunes *et al*, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," *IEEE Commun. Surv. Tut.*, vol.16, no. 3, 2014, pp.1617-1634.

[3] C. Liang and F. Richard Yu, "Wireless Network Virtualization: A Survey, Some Research Issues and Challenges," *IEEE Commun. Surv. Tut.*, vol. 17, no.1, 2015, pp. 358-380.

[4] S. Sarkary, S. Chatterjee, and S. Misra, "Assessment of the Suitability of Fog Computing in the Context of Internet of Things," *IEEE Trans. Cloud Computing*, DOI: 10.1109/TCC.2015.2485206, 2015.

[5] A. Lara, A. Kolasani, and B. Ramamurthy, "Network Innovation using OpenFlow: A Survey," *IEEE Commun. Surv. Tut.*, vol.16, no.1, 2014, pp. 483-512.

[6] ETSI, "Network Functions Virtualization Introductory white paper," v. 2, http://portal.etsi.org/NFV/NFV_White_Paper2.pdf

[7] L. Suresh *et al*, "Towards programmable enterprise wlans with Odin," *in Proceedings of the first workshop on Hot topics in software defined networks. ACM*, 2012.

[8] K. Pentikousis, Y. Wang, and W. Hu, “MobileFlow: Toward Software-Defined Mobile Networks,” *IEEE Commun. Mag.*, vol. 51, no.7, 2013, pp. 44-53.

[9] A. Gudipati, *et al*, “SoftRAN: Software defined radio access network,” *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, ACM, 2013.

[10] C-L I *et al*, “Rethink Fronthaul for Soft RAN,” *IEEE Commun. Mag.*, vol. 53, no. 9, 2015, pp. 82-88.

[11] J. Liu *et al*, “Device-to-Device Communications for Enhancing Quality of Experience in Software Defined Multi-Tier LTE-A Networks,” *IEEE Network*, vol. 29, no. 4, pp. 46-52, Jul. 2015

[12] R. Sherwood *et al.*, “Carving research slices out of your production networks with openflow,” *ACM SIGCOMM Computer Commun. Review*, vol. 40, no. 1, 2010, pp. 129-130.

[13] S. H. Yeganeh and Y. Ganjali, “Kandoo: a framework for efficient and scalable offloading of control applications,” In Proc. *1st workshop on Hot topics in software defined networks*, HotSDN 12, pp. 19-24, New York, NY, USA, 2012. ACM.

[14] N.B. Truong, G. M. Lee, and Y. Ghamri-Doudane, “Software Defined Networking-based Vehicular Adhoc Network with Fog Computing,” *2015 IFIP/IEEE symposium on interated Network Managment,* 2015.

[15] X. Sun, N. Ansari and Q. Fan, “Green Energy Aware Avatar Migration Strategy in Green Cloudlet Networks” *2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)*, Vancouver, BC, 2015, pp. 139-146, 2015.

Kai Liang is now a Ph.D. student in Xidian University and majors in communication and information systems. He received the funding from the program of China Scholarships Council to visit the Network Convergence Laboratory, University of Essex, UK, during September 2014CSeptember 2015. His current research interests include broadband wireless access, wireless network design, coordinated multipoint-to-multiuser communication systems, and fog computing.

Liqiang Zhao received his Ph.D. degree in information and communications engineering from Xidian University, China, in 2003. He is a full professor with Xidian University. His current research focuses on broadband wireless communications, software defined network and fog computing. He has more than 70 published in authorized academic periodicals. He has hosted/participated a great many research projects, such as the National Natural Science Foundation, 863 program, the EU FP6, FP7 plans, and the Huawei fund.

Xiaoli Chu is a lecturer at the University of Sheffield. She received her PhD degree from Hong Kong University of Science and Technology. She has published more than 90 peer-reviewed journal and conference papers. She is leading editor/author of Heterogeneous Cellular Networks (Cambridge University Press, 2013). She was Co-Chair of the Wireless Communications Symposium at IEEE ICC 2015, and Guest Editor for IEEE Transactions on Vehicular Technology and ACM/Springer Journal of Mobile Networks and Applications.

Hsiao-Hwa Chen [F] is currently a distinguished professor in the Department of Engineering Science, National Cheng Kung University, Taiwan. He obtained his B.Sc. and M.Sc. degrees from Zhejiang University, China, and a Ph.D. degree from the University of Oulu, Finland, in 1982, 1985, and 1991, respectively. He served as the Editor-in-Chief of IEEE Wireless Communications from 2012 to 2015. He is a Fellow of IET and an elected Member at Large of the IEEE Communications Society.