MobileFlow: Toward Software-Defined Mobile Networks

Kostas Pentikousis, Yan Wang, and Weihua Hu, Huawei Technologies

ABSTRACT

Mobile carrier networks follow an architecture where network elements and their interfaces are defined in detail through standardization, but provide limited ways to develop new network features once deployed. In recent years we have witnessed rapid growth in over-the-top mobile applications and a 10-fold increase in subscriber traffic while ground-breaking network innovation took a back seat. We argue that carrier networks can benefit from advances in computer science and pertinent technology trends by incorporating a new way of thinking in their current toolbox. This article introduces a blueprint for implementing current as well as future network architectures based on a software-defined networking approach. Our architecture enables operators to capitalize on a flow-based forwarding model and fosters a rich environment for innovation inside the mobile network. In this article, we validate this concept in our wireless network research laboratory, demonstrate the programmability and flexibility of the architecture, and provide implementation and experimentation details.

INTRODUCTION

Carrier networks rely on pricy, tightly integrated, and monolithic machinery which is neither easy to configure optimally nor troubleshoot. Carrier equipment follows widely accepted network standards, such as those introduced by the Third Generation Partnership Project (3GPP). However, typical implementations rely on vendor-specific hardware platforms. Operators often end up in vendor lock-ins as entire areas must be covered with equipment from the same vendor for maximum efficiency. These problems are manifested in the unlicensed spectrum wireless industry too. For example, the control and provisioning of wireless access points (CAPWAP) standard (RFC 5415) enables a single access controller to manage numerous WiFi access points, possibly procured from a variety of vendors. In practice, CAPWAP multivendor interoperable deployments are uncommon. More critically, the closed nature of modern network equipment prevents the research community from exploring new paradigms using real-world equipment. For mobile operators, although new revenue opportunities arise, such as machine-type communication (MTC), standardization takes years from concept definition to commercial equipment that can realize the service becoming widely available. In short, the carrier network paradigm we have come to trust has reached its limits.

On these grounds, software-defined networking (SDN) emerged, aiming at a shift toward a flow-centric model that employs inexpensive hardware [1], a logically centralized network controller, and assorted applications that utilize controller-exposed information to orchestrate service delivery in the network [2]. SDN takes cues from the way we run networks today: modern networks, be they campus, data center, enterprise, or carrier networks, follow domain-centered deployment and operation. In this paradigm, each domain could develop its own network services, addressing specific user needs. Presently, this can only be done on an end-to-end basis (i.e., on top of IP), as only vendors can modify the highly sophisticated but primarily hardware-based network elements. Instead, SDN advocates the use of simple but software-programmable network switches. This model can take advantage of modern agile programming methodologies, which enable network software to be developed, enhanced, and upgraded at much shorter cycles than what we experience today with state-of-the-art hardware-centric network machinery.

The first generation of SDN development is closely associated with OpenFlow [2], but the lasting contribution of this early phase work is ushering in radical new thinking in the way we design and realize networks. SDN explicitly separates the control and data planes in a manner more daring than other carrier-grade architectures, such as the 3GPP Evolved Packet Core (EPC). Moreover, SDN has triggered significant interest in network function virtualization (NFV), as indicated by the latest developments in standardization organizations such as the Internet Engineering Task Force (IETF) and European Telecommunications Standards Institute (ETSI). In this context, we advance the state of the art in SDN through three contributions. First, we introduce the software-defined mobile network (SDMN) architecture, a blueprint for carrier-grade flow-based forwarding and a rich environment for innovation at the

0163-6804/13/$25.00 © 2013 IEEE

core of the mobile network. The first publication of concept validation laboratory results is our second contribution. The hindsight and implementation details provided should be of great interest to the research community at large. Finally, backed by experimental testing, we explore the programmability, flexibility, and openness of the proposed architecture, detailing how mobile networks can be developed and deployed in the future.

The remainder of this article is organized as follows. After reviewing pertinent related work, we provide a brief background on EPC. We then present SDMN, detailing the core components, and explain the different models supported. Finally, we illustrate our research laboratory experiments and conclude the article.

Related Work

Since the introduction of OpenFlow [2], prominent SDN work has addressed several topics such as network virtualization [1, 3], data center and cloud networking [4, 5], acceleration in value-added network service development [6], and network management and control platforms [7], taking an implementation- and experimentation-oriented approach from the beginning. Wireless networking has also been considered, although salient work in this area, to date, has focused primarily on campus-like wireless deployments. The feasibility of mobility management with vertical handovers between IEEE 802.11 and IEEE 802.16 networks was explored in [8]. Although this work is indicative of future developments in wireless networking, it does not address the challenges mobile operators face when extending a model originally designed for wired networks to the wireless domain. Specifically, carrier-grade software-defined mobile networks have not received the attention they deserve given their ubiquity and importance in connecting the majority of Internet users around the world.

Mobile operators are interested in mature limited-risk technologies that are highly optimized to make the most out of scarce (licensed) wireless resources, scalable to handle billions of users as today's networks can, and ready to foster innovation inside the network. A recent position paper on software-defined cellular networks [9] makes the case that SDN can simplify cellular networks and lower management costs. We concur with Li *et al.* [9] that as one introduces SDN in a mobile network, the challenges lying ahead are significant. Unfortunately, the authors do not provide an architectural blueprint detailing their proposal; nor do they report experimental results. On the other hand, Kempf *et al.* [10] present a detailed study on the evolution of EPC toward a model where the control plane can be "lifted up" from the core network elements and henceforth reside in a data center. The authors review the current EPC architecture and document their implementation, which is based on OpenFlow 1.2 protocol extensions, at a sufficient level of detail so that it can be replicated by others in the SDN community. The article, which we view as complementary to our work, emphasizes the role that data centers can play in future carrier networks and lists possible architectural modifications to EPC, but is very terse on prototype details. Indeed, the authors' experience with using the prototype appears as a work in progress, and the article does not provide experimentation results. The work presented in this article can coincide with that described in [10], but takes a different approach that is not as closely knit to the use of data centers while deploying the proposed architecture. NFV is currently expected to be a driving force in future carrier network standardization, and there are many different paths one can follow in both the system design phase and actual deployment. We argue in this article that SDMN is demonstrably flexible enough to accommodate numerous ways forward.

This article advances the state of the art in the public peer-reviewed SDN literature by describing a concrete architectural proposal that applies the core SDN principles to mobile carrier networks. We detail the development and use of a proof-of-concept prototype to validate the key components of the architecture, following the SDN tradition of implementation-oriented experimentation.

Mobile Broadband Today

EPC is a special-purpose, flat, all-IP architecture standardized by 3GPP [11–13]. EPC, which along with the evolved Universal Mobile Telecommunications System (UMTS) terrestrial radio access network (E-UTRAN) forms the foundation of the Evolved Packet System (EPS) and fourth generation (4G) networks, is currently deployed by dozens of operators around the world. EPC made significant steps forward in terms of embracing packet-switched networking, aiming at reduced complexity and increased scalability through data and control plane separation. EPC acts as a unifying routing fabric in the core network used by different 3GPP-standardized radio technologies as well as non-3GPP access technologies such as IEEE 802.11, among others.

EPC is a highly optimized mobile service overlay that capitalizes on IP- or Ethernet-based transport to deliver a diverse range of services through secure communications with quality of service (QoS) guarantees and seamless mobility support [11–13]. Figure 1 illustrates the key entities, functional elements, and interfaces involved in typical EPS operation during a voice call, video streaming, or online access to web content. We do not delve into the details of EPS operation due to space constraints (interested readers should consider [11–13, references therein]), but note that traffic from/to the user equipment (UE) is effectively directed through a set of general packet radio service (GPRS) Tunneling Protocol (GTP)/Proxy Mobile IP (PMIP) tunnels from the attached eNB to the serving gateway (SGW) to the packet data network (PDN) gateway (PGW) and onward to the corresponding PDN; see [11] for more details. As illustrated in Fig. 1, user packets are encapsulated/decapsulated and processed successively by different functional elements associated with operational and management functions such as

EPC is a special-purpose, flat, all-IP architecture standardized by 3GPP. EPC, which along with E-UTRAN, forms the foundation of the EPS and 4G networks, is currently deployed by dozens of operators around the world.

A top-level requirement for SDMN is to provide maximum flexibility, openness, and programmability to future carriers without mandating any changes in UE. In this way, operators can innovate inside their domain without having to depend on either over-the-top (OTT) service providers or UE vendors to support their innovations.

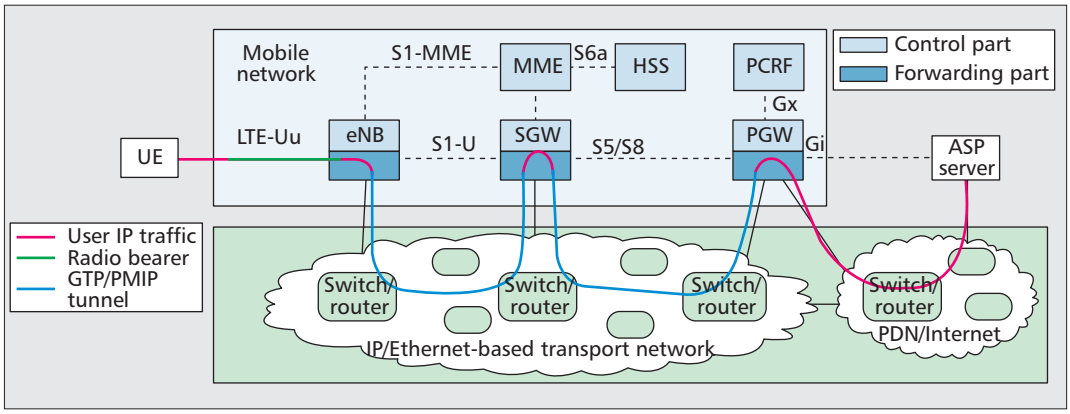

Figure 1. *Evolved packet system.*

charging, policing, authentication, authorization, and accounting (AAA) operations, mobility management, and so on. Each "box" in this architecture typically stands for a physical network device, often powered by specialized software and hardware, adhering to carrier-grade operation requirements. This orientation toward specialized hardware stems, in part, from the evolution of telecommunication networks [14, references therein].

This mobile network architecture has been in general successful as manifested by the number of mobile subscribers and the multibillion dollar industry around it. However, as new technologies such as content distribution and cloud computing enter the mobile operator domain, the complex nature of mobile broadband technologies starts to become an impediment to sustainable future growth. EPC is a fit-for-purpose closed system based on standardized interfaces, where every component performs specific functions, and each of the dozens of interfaces has a unique definition. Further evolution of the mobile broadband architecture along this line of thinking has certain drawbacks. First, operators will have to deal with higher capital and operational expenditures (CAPEX/OPEX) at a time when average revenue per user (ARPU) is decreasing. Progress down this evolutionary path, based on this so-called stovepipe design model, is bound to incur an increasing rate of costs in the future. Meanwhile, operations staff have to maintain many types of network elements and follow the evolution of dozens of interfaces. Second, as higher CAPEX/OPEX forces some operators to refrain from investing further, those who do invest face long time-to-market periods as it is taxing to add new features. Interested operators must push a whole industry to standardize these new features, and then wait for vendors to actually implement them. This process does assure quality standards development, but prevents operators who are willing to endeavor first into new domains from doing so in a fast pace. Finally, a direct consequence of the specialization of each network element, which is defined solely for EPC, is limited openness and flexibility. Once purchased, EPC elements can be controlled through standard interfaces, but cannot be extended through the use of open interfaces or application programming interfaces (APIs). If the operator expects that new kinds of network services would appeal to its subscribers, existing equipment may be of little use in developing and deploying said services.

Software-Defined Mobile Network

In order to address the challenges introduced in the previous sections, and motivated by the core principles of the SDN paradigm, we set out to define what we refer to as the software-defined mobile network (SDMN). A top-level requirement for SDMN is to provide maximum flexibility, openness, and programmability to future carriers without mandating any changes in UE. In this way, operators can innovate inside their domain without having to depend on either over-the-top (OTT) service providers or UE vendors to support their innovations. More important, SDMN aims at making the forwarding substrate fully software-driven, enabling the creation of any network type on demand (as seen later in this article), and opens the mobile network to innovation through new service enablers.

Figure 2 illustrates the central elements of SDMN, contrasting it visually with today's EPC (Fig. 1). The key enablers in our architecture consist of the MobileFlow forwarding engine (MFFE) and MobileFlow controller (MFC). MFFEs are interconnected by an underlying IP/Ethernet transport network. Recall from Fig. 1 that in EPC, the SGW and PGW are user plane elements, while the mobility management entity (MME) is a control plane element. Figure 2 illustrates a new split that decouples mobile network control from all user plane elements. This way the (new) user plane (i.e., the MFFE) becomes simple, stable, and high-performing, while the control plane (i.e., the MFC and mobile network applications) can be implemented in a logically centralized manner. Forwarding in the MFFE can be fully defined in software, while the control software can flexibly steer user traffic to different service enablers (e.g., deep packet inspection [DPI], video caching, and optimization) that can be distributed throughout the mobile network, as shown in Fig. 2. This design, which follows in the tradition first explored in [6], permits us to facili-

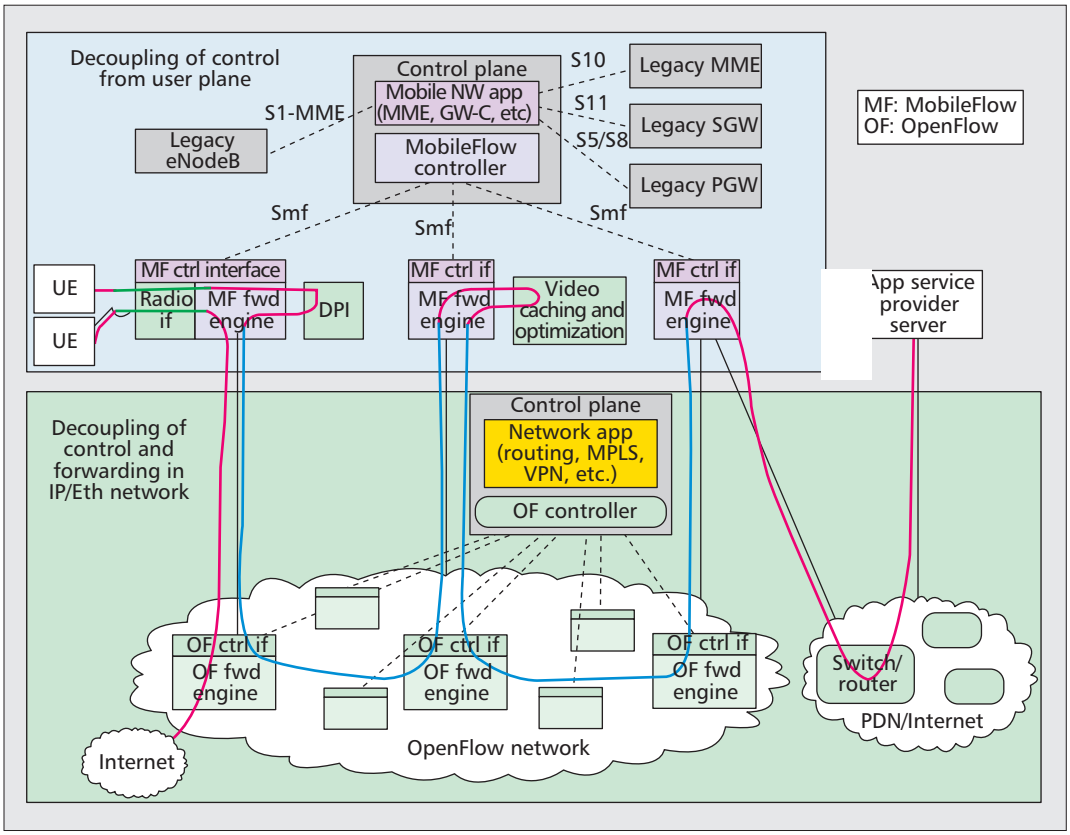

Figure 2. *Software defined mobile network.*

MFFEs include standard mobile network tunnel processing capabilities, such as, for instance, GTP-U and GRE encapsulation/ decapsulation facilities. Consequently, MFFEs, with the support of a MobileFlow controller, can be integrated with legacy EPC equipment.

tate swift network innovation through network function virtualization. For example, in this case, the carrier may opt to either reuse existing DPI equipment or employ virtualized functionality in a data center.

Note also that the figure illustrates two UE devices. The traffic from one of them is handled by a set of MFFEs, while the traffic from the second one is offloaded to the Internet primarily using the underlying OpenFlow transport network. Moreover, the depiction of virtualized network functions, such as DPI, and their interconnection with specific MMFEs is only indicative.

Effectively, we introduce a new stratum based on a logically centralized MobileFlow controller that can steer forwarding. MFFEs employ network virtualization, are suitable for multitenant mobile networks, and can support advanced and security-related network functions. MFFEs are more complex than an OpenFlow switch, but much simpler than a router or a PGW, as the majority of control plane functionality is factored out to the MFC and the associated MobileFlow applications. In other words, significant parts of the functionality of today's EPC can be centralized into a carrier-grade control plane (upper part of Fig. 2). Figure 2 also depicts the OpenFlow-based decoupling of the IP/Ethernet transport network, contrasting it with the aforementioned mobile network control- and forwarding-plane decoupling. As can be surmised from Fig. 2, OpenFlow alone is not sufficient for implementing a carrier-grade mobile network user plane. We distinguish an SDMN from an OpenFlow-based network, as MFFEs are not switch-level equipment. MFFEs must support carrier-grade functionality such as network layer (L3) tunneling and flexible charging.

Our MobileFlow architecture can easily interact with legacy EPC network elements. In the control plane, the mobile network applications running on top of an MFC can function as an MME or the gateway control part of a physical box (indicated as GW-C), and thus can interoperate with other legacy MMEs, SGWs, or PGWs using the well established 3GPP-defined interfaces, as shown in Fig. 2. In the user plane, the MFFE can be interconnected with legacy EPC elements with a common IP/Ethernet transport network. The control messages can be transmitted via the MFFE in an in-band manner or via a dedicated out-of-band network.

Next, we look into the details of each of the architectural components of Fig. 2 and provide implementation insights based on our experience with an SDMN in our research laboratory.

MobileFlow Forwarding Engine

MFFEs include standard mobile network tunnel processing capabilities (e.g., GTP-U and GRE encapsulation/decapsulation facilities) [11]. Consequently, MFFEs, with the support of an MFC, can be integrated with legacy EPC equipment. An MFFE may also include the key functionality of a wireless access node, such as a radio interface to manage radio bearers. As illustrated in Fig. 2, an SDMN does not mandate any changes in UE. From a deployment perspective, one could envision wireless access MFFEs in parallel operation with existing eNodeBs, for example, as

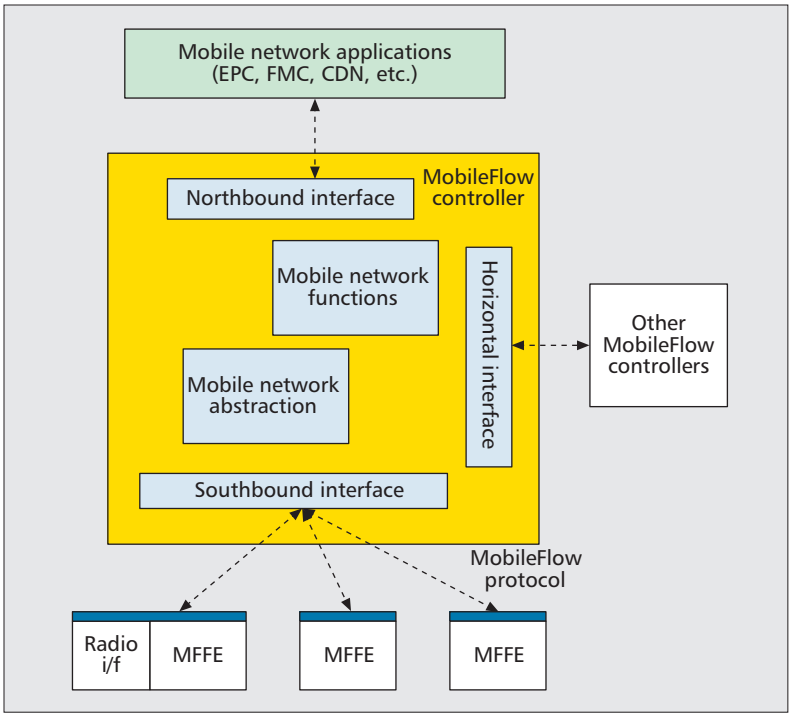

Figure 3. *MobileFlow controller.*

well as the incremental use of MFFEs in the core network as SGW/PGWs are retired.

Each MFFE communicates with an MFC through a lightweight protocol that implements the MobileFlow control interface (Smf). Due to space limitations we cannot delve into the details of this protocol, but suffice it to say that the MFFEs use the flow rules received from the MFC to encapsulate/decapsulate user traffic and/or forward packets to the transport network. Our current protocol implementation borrows from the flow-entry model advocated by the Open Networking Forum (ONF). Although we are still at a proof-of-concept stage, and more work is foreseen in the coming months, the performance we are observing in the laboratory is very promising, since the burdensome gateway control part is factored out of the user plane. Eventually, each MFFE featuring the lightweight Smf-driven control layer will be a high-performance, highly reliable network element that can handle a much larger number of flow entries than commodity OpenFlow switches do. This is because a future carrier mobile network should be able to handle very fine-grained policies, at the very minimum on par with what is now available for commercial PGWs. The comparative advantage is that with MFFEs, one can innovate much faster and, if so desired, can aim for different optimalities than what is possible today.

From an implementation perspective, MFFE can be based on a stripped-down mobile gateway that can handle network-layer (layer 3, L3) tunnel processing in the data plane with all other tunnel control processes moved up to the MFC. Alternatively, one can extend an OpenFlow switch to support the MobileFlow functions mentioned above. A hybrid model can be used as a combined forwarding element in the same IP/switch network level to facilitate 3GPP-IP convergence. For example, in Fig. 2, the MFFE can be combined with an OpenFlow switch, while the MFC can be combined with the OpenFlow controller.

MobileFlow Controller

Figure 3 zooms in on the MFC and illustrates its main functional blocks: the mobile network function, the mobile network abstraction, and the functional blocks corresponding to three interfaces. The MFC southbound interface controls MFFE operation. The horizontal interface is used for communication with other MFCs, and can be employed for intra- and trusted inter-domain cooperation. Operators with very large infrastructure typically segment their network according to administrative, regulatory, technical, or other reasons. In contrast to previous work [8, 9], our architecture specifies this interface from the beginning as we aim at very large carrier-grade deployments with high demands on reliability and availability that can scale out. Finally, the northbound interface facilitates fast network service and application development, as we discuss below.

The MFC relies on network-level abstraction. The corresponding functional block in Fig. 3 includes topology auto-discovery, topological resource view, network resource monitoring, and network resource virtualization. The network functions block includes tunnel processing, mobility anchoring, routing, and charging, among others. For instance, support for creating and establishing GTP-U tunnels is one of the network functions implemented in our testbed prototype, described later in this article. These abstractions correspond to high-level descriptions and are agnostic to the particular MFFE implementation. Thus, an operator can freely use MFFEs from different vendors. In addition, the operator can use MFFEs to adopt and deploy novel mobile network architectures that do not rely on tunneling.

A cornerstone of the SDN paradigm is the possibility of extending current and creating new functionality through programmatic APIs. An SDMN is compatible with advances in the OpenFlow-based ecosystem. Moreover, Fig. 3 illustrates the open interfaces and associated APIs introduced by the SDMN. Note that the SDMN controller does not have an explicit interface to legacy infrastructure. Any legacy interfacing can be handled efficiently by MobileFlow applications. The SDMN northbound interface, for example, can be used by MobileFlow applications to implement different network functionality. The MobileFlow stratum can be engaged in managing user traffic selectively depending on several criteria.

Returning to Fig. 2, recall that some traffic can be handled directly by the underlying OpenFlow-based transport network, while other traffic may require special treatment or support from advanced services. For example, mobility management, DPI, video optimization, and fine-grained QoS support can be selectively activated and handled at the MobileFlow stratum for premium traffic, while offloaded traffic can be handled by the transport network. We detail in the "Implementation" section later in this article

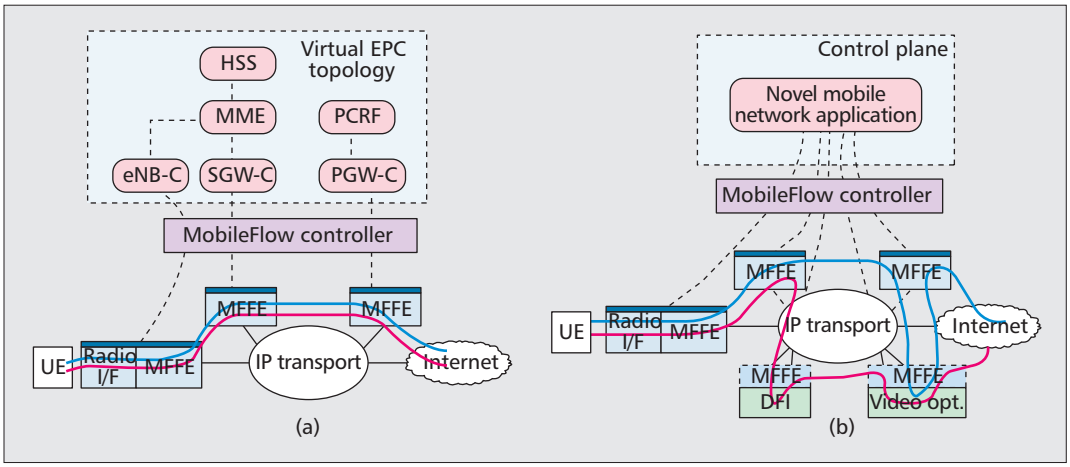

Figure 4. *MobileFlow application models.*

how this is employed to realize different cellular architectures.

Mobile Network Applications

Mobile network applications can be developed by taking advantage of the SDMN northbound interface. Figure 4 illustrates the two models we explored while developing a fully EPS-compliant network based solely on an SDMN. The first model, illustrated in Fig. 4a, adopts a network function virtualization approach: a MobileFlow application implements the control part functionality of each of the EPS elements (e.g., eNodeB-C, SGW-C, PGW-C, S/PGW-C, and GGSN-C). Then each of these applications, which can run in a cloud computing environment similar to what is proposed in [10], interacts with the MFC to manage a respective MFFE, effectively implementing a virtual gateway. This model features a 1:1 mapping between the virtual gateway and MFFE via the MobileFlow controller. In other words, an EPC virtual topology is constructed in the MobileFlow stratum, as shown in Fig. 4a, according to the relationship between the corresponding reference architecture (Fig. 1) network elements (eNB-C, SGW-C, PGW-C, MME, PCRF, and HSS). Value-added functionality (e.g. video optimization) can be introduced by implementing the corresponding application and mapping it to an MFFE. This model is akin to the one introduced in [3], and although control may be centralized, application interaction is logically distributed.

The second model, shown in Fig. 4b, adopts a 1:*m* mapping between one all-encompassing MobileFlow application and several MFFEs. The application uses the SDMN northbound interface to access the substrate resource topology from the MobileFlow controller. In addition, it dictates the traffic paths and operations within the MFFE topology based on the network abstraction obtained from the MFC (Fig. 3). This model does not have to follow currently standardized interfaces and can therefore provide fertile ground for developing novel mobile network systems. For example, we can develop a virtual EPC gateway which aggregates functionality by factoring in that is currently distributed between several network elements, thus optimizing control signaling, while incorporating various SDMN-based service enablers, as shown in Fig. 4b.

Both models support *m*:1 mapping between MobileFlow applications and MFFEs, as each application belongs to a different control plane, therefore enabling multitenancy in a straightforward manner. Note that the first model is particularly suitable for integrating legacy infrastructure transparently into an SDMN system. The operator is free to employ the former model for one part of the network and the latter one for another part in which it can experiment with and readily deploy new network services. In fact, through our support for multitenancy we expect that it is possible to have both models in the same part of the network, although this is left as future work with respect to prototype implementation. Regardless of which model is preferred, SDMN enables flexible on-demand mapping between MobileFlow applications and their MFFE counterparts, as we explain in the following section.

Mobility Management

Supporting mobility in an SDMN, whether based on existing standards by 3GPP and IETF or newly proposed by the research community, can be realized by introducing the corresponding applications above the northbound interface of the MFC. For instance, as shown in Fig. 4a, where we take EPS as an illustrative case, we can use SDMN to implement the current generation of control functionality in a 3GPP mobile network. The wireless access MFFE with its eNB-C acts as an eNodeB. The MFFE with its associated SGW-C acts as the SGW, and the MFFE with its associated PGW-C acts as the PGW. In this case, mobility is managed by the virtualized but standard-adhering MME. The interfaces between MME and eNB-C/SGW-C are as defined by 3GPP (i.e., S1-MME and S11). The interface between SGW-C and PGW-C also follows the 3GPP-defined GTP-C part of S5/S8. The corresponding MFFEs handling the user traffic are controlled by their "-C" control counterparts with respect to processing GTP-U packets. The main difference from current deployments is that the -C virtualized function

Supporting mobility in SDMN, whether based on existing standards by 3GPP and IETF or newly proposed by the research community, can be realized by introducing the corresponding applications above the northbound interface of the MFC.

We validated the SDMN architecture through research prototyping using COTS x86 based general-purpose servers and OpenFlow-based components. To showcase the flexibility of our architecture we developed the ODMN prototype, which demonstrates the core benefits of our MobileFlow approach.

translates the GTP signaling/context into flow rules and sends the rules via the MFC to each of the MFFEs involved in handling the particular flow. In other words, via the Smf interface we can control packet forwarding behavior, such as GTP tunnel encapsulation/decapsulation, quality of service (QoS) metering, forwarding, and so on, without having this type of control plane functionality in the boxes deployed in the field. The most challenging aspect here is how to split the functionality of an eNB, which is more complex than splitting the control and data planes in an S/P-GW. The interface between eNB-U and eNB-C consists of the forwarding control and wireless bearer control. The wireless bearer control is relevant to the radio handoffs required in mobility management processes, and at this stage of development this design aspect is part of our ongoing work. With respect to the prototype implementation we describe in the following section, the reader should keep in mind that the current code addresses only the split of S/PGW.

Looking forward, one can experiment on a small scale and then widely deploy novel mobility management solutions. In an SDMN this is done by upgrading or changing altogether the respective MobileFlow applications without replacing the deployed MFFEs. For example, Fig. 4b indicates that we can deploy a flat mobility management scheme which does not depend on S/PGW-like anchor nodes. This opens up opportunities for introducing simpler handover and gateway change processes, which today require lengthy standardization efforts. Earlier work on OpenPipes [6] and OpenRoads [8] explored a range of possibilities for optimized traffic steering through different service enablers such as video optimization, we discussed earlier in this article and illustrated explicitly by the different paths for the red and blue flows in Fig. 4b. Adopting a design based on locator/identifier split, including LISP, HIP, and others discussed in [15], can easily be accommodated. An SDMN aims to provide the same level of flexibility in handling mobility of users, flows, and content, but with carrier-grade performance. We are currently actively following the developments in the IETF Distributed Mobility Management (DMM) working group, and we plan to showcase in the future how SDMN can implement the novel solutions standardized there.

Implementation

We validated the SDMN architecture through research prototyping using COTS x86 based general-purpose servers and OpenFlow-based components. To showcase the flexibility of our architecture we developed the On-Demand Mobile Network (ODMN) prototype, which demonstrates the core benefits of our MobileFlow approach. In particular, we show that the MobileFlow stratum enables the software-based definition of different mobile networks (3G, 4G, FMC, and others) with different specifications (radio coverage, subscription numbers, PDP/bearer numbers, maximum bandwidth, etc.) according to administrator demand.

The ODMN is effectively a virtual network management system in which network resources include control and forwarding plane elements. We implement control plane network functions as mobile network applications (e.g., SGW-C, PGW-C, MME, and PCRF) and run them on general-purpose servers. Forwarding plane resources consist of radio access nodes and MFFEs. MFFEs may be implemented solely in software or using specific hardware platforms. By supporting virtualization for both control- and forwarding-plane resources, it is easy to create multiple virtual mobile networks with different types and specifications over the same hardware environment, as illustrated in Fig. 5. The same MFFE can be reused by different types of virtual networks. Each virtual mobile network can evolve and be upgraded solely by replacing the virtual machine running the corresponding MobileFlow application on the control plane servers — MFFEs remain unchanged.

Our testbed environment includes five servers in the control plane, as illustrated in the bottom left part of Fig. 6. Servers marked C1–C4 run multiple KVM virtual machines with different MFCs and applications (MME, SGW-C, PGW-C, GGSN-C, SGSN, etc.). A fifth server acts as the ODMN management server handling the creation, modification, and termination of virtual mobile networks. For example, Fig. 6 illustrates that we created three different networks. In the forwarding plane there are three radio access nodes (eNB1–2, NodeB) and three servers (F1–F3). eNB1 is a real Huawei eNodeB, while NodeB, radio network controller (RNC), and eNB2 are emulated using the Huawei Network Traffic Emulator (NTE). We developed each MFFE by extending the Open vSwitch code (available from www.openvswitch.org) to support GTP tunneling. All control plane and forwarding plane servers are interconnected by commercial off-the-shelf (COTS) LAN switches.

We also developed the ODMN testbed operator graphical user interface (Fig. 6, top left), which we refer to as the ODMN dashboard. This dashboard provides an overview of the physical resource topology and, more important, is used for all virtual resource allocations described next. The dashboard GUI also displays, over each network element icon, the virtual resource instances. Finally, the right side of Fig. 6 portrays different snapshots from the virtual networks we instantiate for the demonstration presented in this article.

Without loss of generality, we use ODMN to deploy three different mobile networks concurrently on the research testbed. The first network (marked in yellow in Fig. 6) is a standard 3G network composed of an NTE emulated RNC, a virtualized serving GPRS support node (SGSN), and a virtualized gateway GPRS support node (GGSN). As the color code indicates in the dashboard (Fig. 6, top left), the SGSN corresponds to an SGSN virtual machine (VM) on C1; the GGSN is composed of an MFFE VM on F3 and a corresponding GGSN-C VM on C4. The second virtual network (shown in red in Fig. 6) is a 4G network including the real eNodeB (eNB1), a virtual MME (C3), and a virtualized S/PGW. The virtual S/PGW comprises an MFFE VM on F1 and the corresponding S/PGW-C VM on C3. The third virtual network (in green) is

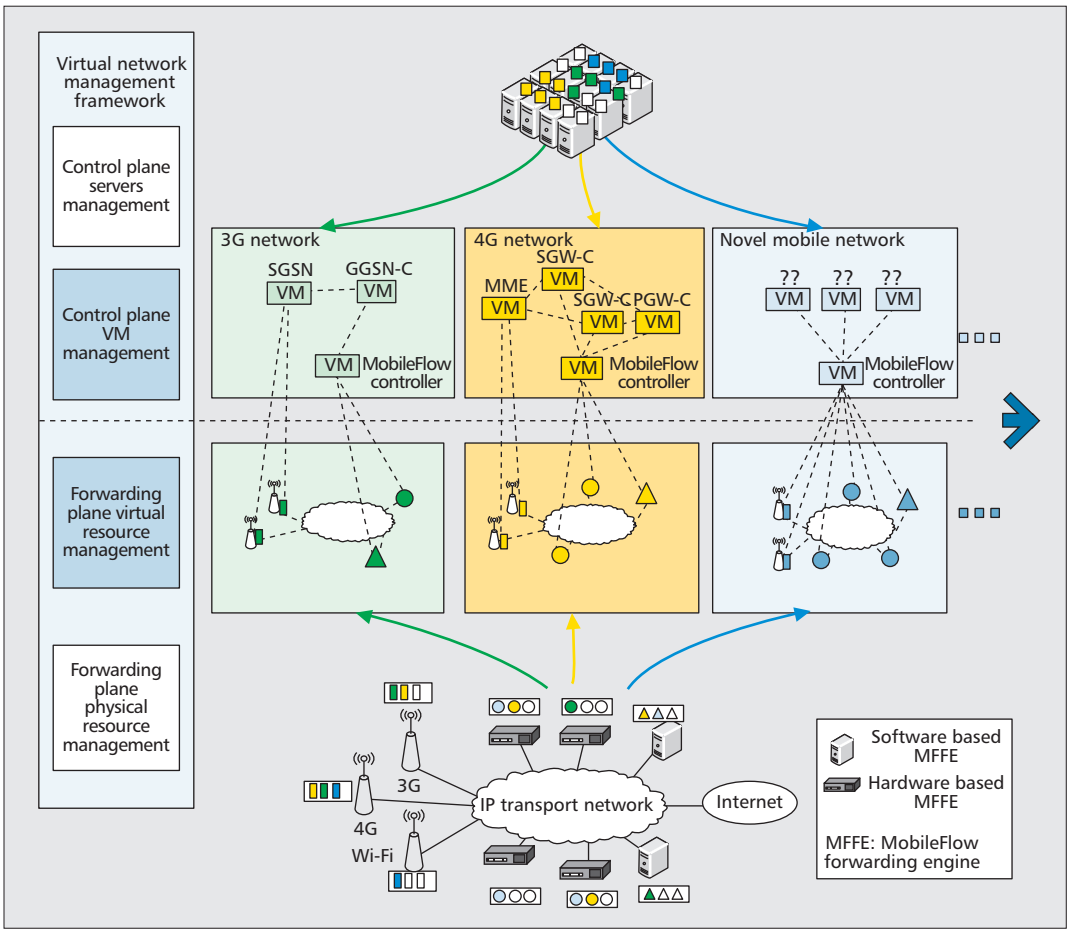

Figure 5. *On-demand mobile network.*

This essential functional test verified that the basic 3GPP attachment and bearer establishment procedures are correctly implemented using SDMN. However, the prototype does not fully support charging and policing; enhancing this aspect is part of our ongoing implementation work.

another 4G network including an NTE emulated eNodeB (eNB2), a virtual MME (C1), a virtual SGW, and a virtual PGW. The virtual SGW comprises an MFFE VM (F2) and the corresponding SGW-C VM on C2. The virtual PGW is composed of an MFFE VM on F3 and the corresponding PGW-C VM on C2.

In ODMN, a COTS LTE Android smart phone can successfully attach to the virtual 4G mobile network ("red network") via a real-world eNodeB (eNB1) through the standard 3GPP attachment procedure. Since the virtual S/PGW (corresponding to MFFE VM on F1) connects through the laboratory LAN to the Internet, the user of the LTE smartphone can surf the web through the ODMN-based virtual mobile network. This essential functional test verified that the basic 3GPP attachment and bearer establishment procedures are correctly implemented using an SDMN. However, the prototype does not fully support charging and policing; enhancing this aspect is part of our ongoing implementation work.

For the virtual 3G network ("yellow network" in Fig. 6) and virtual 4G network ("green network"), we verified their on-demand performance capability for the maximum bandwidth and number of PDP/bearer contexts. We specify the maximum bandwidth to 100 and 200 Mb/s for the virtual 3G and 4G networks, respectively. We specify the maximum number of PDP/bearer contexts to 200 for both virtual networks. We employed NTE to emulate 400 PDP/bearers attached to the two networks with 400 Mb/s maximum bandwidth, respectively, and verified that the ODMN can successfully match the maximum bandwidth and number of PDP/bearers as per each test specification. As illustrated above, virtualization is explicitly supported, while multitenancy is taken to a new level as mobile networks with different types and specifications can be deployed on demand.

Operator Benefits

By explicitly separating the transport stratum from the mobile network one, we introduce a new degree of freedom in our system, which is not present in an OpenFlow-based network. First, the two strata can be developed and deployed independently. This means that a mobile operator can continue to benefit from the reliability of current EPC equipment while rolling out a commodity OpenFlow-based transport network to reduce costs. Simultaneously, the operator can start to deploy MFFEs, which can interoperate with legacy equipment (e.g., via GTP/PMIP tunnels), and experiment with new services built on the software-defined part of the mobile network. Without the MobileFlow stratum, an operator would effectively have to discard all current equipment, including specialized middleboxes, and reintroduce the same functionality over the transport network applications space. This latter approach has many risks due to the

lack of carrier-grade mature OpenFlow products. Instead, our architecture enables operators to choose the best of both worlds and migrate to a fully software-defined network on their own initiative and pace. Furthermore, the architecture is forward-looking, so the operator can also replace the transport stratum without losing its investment in the MobileFlow stratum. This is of particular interest, for example, when considering fixed mobile convergence (FMC).

As our research prototype proof-of-concept results indicate, SDMN enables operators to use infrastructure resources to orchestrate on demand the creation of various mobile network pipes based on different mobile architectures (3G, 4G, FMC, etc.) using the appropriate MobileFlow applications. While doing so, the operator can manage traffic steering for value-added services, enabling new possibilities for improving the end-user quality of experience as well as optimizing the cost of handling lesser-value traffic flows. An example of the latter includes offloading selected traffic to the closest egress at the transport network level without having to traverse the mobile core, as illustrated in Fig. 2. Since the MFC maintains a complete view of the infrastructure network topology, MobileFlow applications can be used to orchastrate various service chains, thus delivering a personalized user experience.

Last but certainly not least, SDMN enables speedy and smooth mobile network evolution and differentiation. As the example in Fig. 5 illustrates, an operator can start using SDMN with a 3G network in place and evolve it into a 4G network by allocating more resources as the rollout proceeds. Changing important mechanisms at the core is also decoupled from evolution at the edge. Today this would require a technology swap, which is costly, time-consuming and may introduce competitive deployment races that can hurt operator profits. Through a phased approach that can follow each operator's own pace, future SDMN-based networks can be deployed and operated. Moreover, within the same domain, a carrier can allocate a dynamic

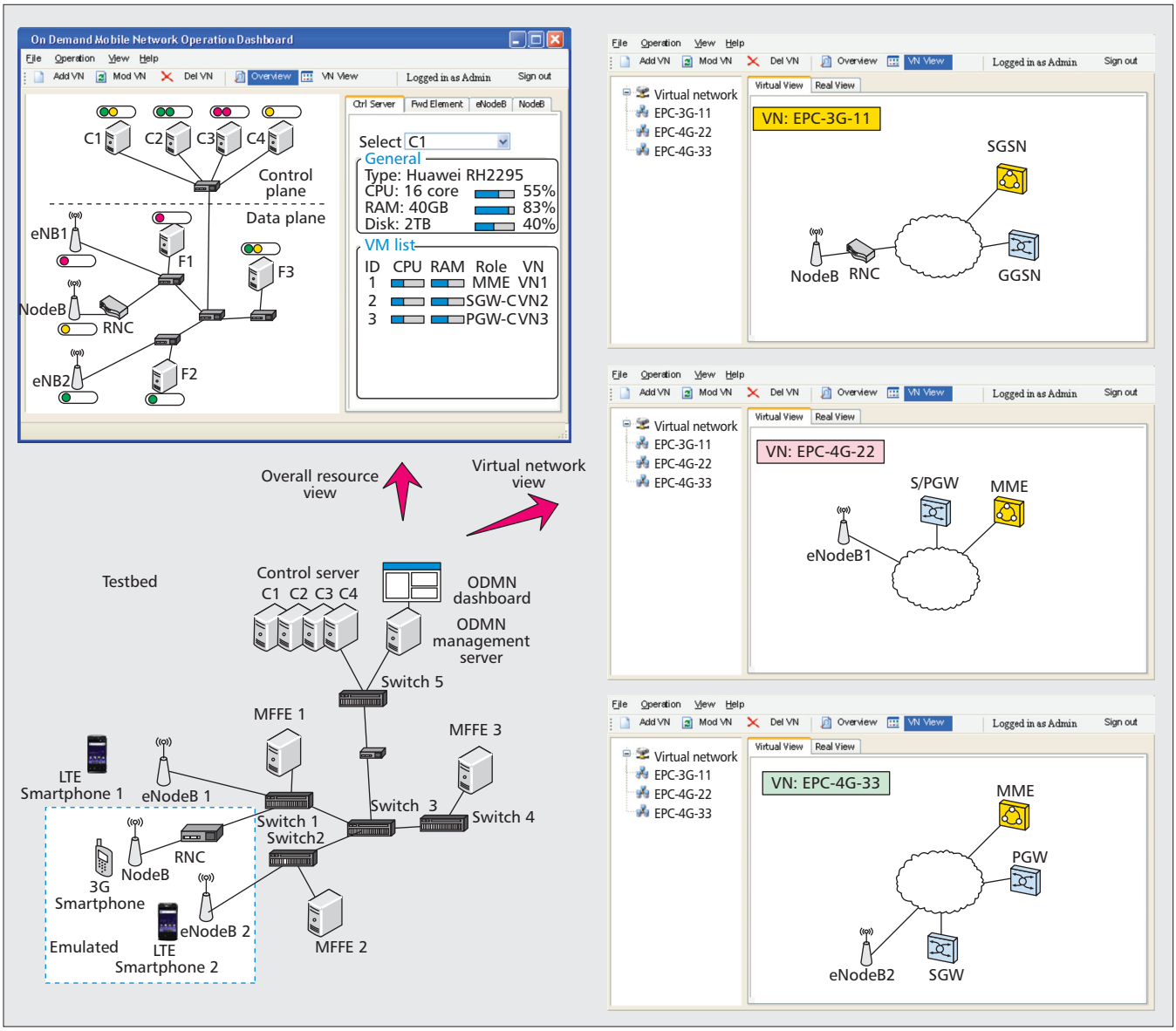

Figure 6. *ODMN prototype testbed and dashboard user interface.*

mix of control and user plane resources from the virtualized network infrastructure to address different pain points as they emerge (e.g., signaling storms or spikes in content distribution during popular events) on much shorter timescales than today. Within this context, operators may procure (or develop) different sets of MobileFlow applications based on the subscriber groups they address and thus further differentiate themselves from their competition.

Conclusion

We introduce SDMN, an architecture for future carrier networks that builds on the decoupling of data and control in the mobile network user plane and a new MobileFlow stratum, which can significantly increase the operator innovation potential. The SDMN open interfaces and APIs foster service innovation, increasing the capability of the operator to roll out new network features while reducing time to market for new services. Our proof-of-concept prototype demonstrates that by employing the SDMN architecture a carrier can configure on-demand and in detail the network architecture, radio coverage, gateway location, PDP/bearers number, maximum throughput, and so on. Moreover, as we have seen, an SDMN operator can orchestrate new service chains and evolve its network by updating or replacing MobileFlow applications.

Our ODMN testbed implementation validates the programmability and flexibility of the SDMN approach. That said, many challenges lie ahead as we focus on improving performance, ease of resource configuration and management, scalability, and code maturity. ODMN provides the foundation for further research and experimentation in this area. We expect that SDMNs can catalyze operator innovations in a range of areas, from speedy introduction of new services to improved monitoring and management of network resources and services, to controlling CAPEX/OPEX, to personalized handling of subscriber traffic to yield more revenue and reduce churn. With SDMN, operators can differentiate their offerings at an unprecedented scale, while protecting their current investment in traditional cellular equipment.

Acknowledgment

We are grateful to the anonymous reviewers for their constructive comments and suggestions, which helped us to hone the core message of this article. We thank Dr. Jiang Li and Hua Huang for their comments and suggestions, which helped improve the overall quality of this work. The views expressed herein are solely those of the authors and do not necessarily represent the views of Huawei Technologies.

References

[1] A. Greenhalgh *et al.*, "Flow Processing and the Rise of Commodity Network Hardware." *SIGCOMM CCR*, vol. 39, no. 2, ACM, 2009, pp. 20–26.
[2] N. McKeown *et al.*, "OpenFlow: Enabling Innovation in Campus Networks," *SIGCOMM CCR*, vol. 38, no. 2. ACM, 2008, pp. 69–74.
[3] M. R. Nascimento *et al.*, "Virtual Routers as A Service: the RouteFlow Approach Leveraging Software-Defined Networks," *Proc. CFI*, Seoul, Korea. ACM, 2011.
[4] C. J. Sher Decusatis *et al.*, "Communication within Clouds: Open Standards and Proprietary Protocols for Data Center Networking," *IEEE Commun. Mag.*, vol. 50, no. 9, Sept. 2012.
[5] M. Bari *et al.*, "Data Center Network Virtualization: A Survey," *IEEE Commun. Surveys & Tutorials*, vol. PP, no. 99 (early access article).
[6] G. Gibb *et al.*, "OpenPipes: Prototyping high-speed networking systems," *Proc. SIGCOMM (Demo)*, Barcelona, Spain, 2009.
[7] T. Koponen *et al.*, "Onix: A Distributed Control Platform for Large-Scale Production Networks," *Proc. OSDI*, Vancouver, BC, Canada, 2010.
[8] K.-K. Yap *et al.*, "Blueprint for Introducing Innovation into Wireless Mobile Networks," *Proc. SIGCOMM VISA Wksp.*, New Delhi, India, 2010.
[9] E. Li *et al.*., "Toward Software-Defined Cellular Networks," *Proc. EWSDN*, Darmstadt, Germany, 2012.
[10] J. Kempf *et al.*, "Moving the Mobile Evolved Packet Core to the Cloud," *Proc. WiMob*, Barcelona, Spain, 2012.
[11] I. Ali *et al.*, "Network-based Mobility Management in the Evolved 3GPP Core Network," *IEEE Commun. Mag.*, vol. 47, no. 2, Feb. 2009.
[12] H. Ekström, "QoS Control in the 3GPP Evolved Packet System," *IEEE Commun. Mag.*, vol. 47, no. 2, Feb. 2009.
[13] J.-J. Pastor Balbás *et al.*, "Policy and Charging Control in the Evolved Packet System," *IEEE Commun. Mag.*, vol. 47, no. 2, Feb. 2009.
[14] K. Pentikousis *et al.*, "Design Considerations for Mobility Management in Future Infrastructure Networks," *Proc. ITU TELECOM WORLD Technical Symp.*, Geneva, Switzerland, 2011.
[15] B. Sousa *et al.*, "Multihoming Management for Future Networks," *ACM/Springer Mobile Networks and Applications*, vol. 16, no. 4, Aug. 2011, pp. 505–17.

Biographies

Kostas Pentikousis (k.pentikousis@huawei.com) is a senior research engineer at Huawei Technologies, Berlin, Germany. He earned his Bachelor's degree in informatics from Aristotle University of Thessaloniki, and his Master's and doctoral degrees in computer science from Stony Brook University. At Huawei he worked on 3GPP EPC research topics beyond Rel. 12, carrier-grade SDN, and NFV.

Yan Wang (jason.wangyan@huawei.com) is a senior research engineer at Huawei Technologies, Shanghai, China. He earned his doctoral degree in communication and information systems from the State Key Laboratory on Advanced Optical Communication Systems and Networks, Shanghai Jiao Tong University (SJTU), China. At Huawei he worked as the leader of the Software Defined Mobile Network (SDMN) group. His research interests include 3GPP packet core research topics, carrier-grade SDN, and NFV.

Weihua Hu (huweihua@huawei.com) is a senior research engineer at Huawei Technologies in Shanghai, China. He earned his Bachelor's and Master's degrees in control theory and control project from SJTU. At Huawei he worked on 3GPP packet core research topics, carrier-grade SDN, and NFV.

Our ODMN testbed implementation validates the programmability and flexibility of the SDMN approach. That said, many challenges lie ahead as we focus on improving performance, ease of resource configuration and management, scalability, and code maturity.