This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

Wireless Software Defined Networking: a Survey and Taxonomy

Israat Tanzeena Haque, *Senior Member, IEEE* and Nael Abu-Ghazaleh, *Senior Member, IEEE*

Abstract—One of the primary architectural principles behind the Internet is the use of distributed protocols, which facilitates fault tolerance and distributed management. Unfortunately, having nodes (i.e., switches and routers) perform control decisions independently makes it difficult to control the network or even understand or debug its overall emergent behavior. As a result, networks are often inefficient, unstable and fragile. This Internet architecture also poses a significant, often insurmountable, challenge to the deployment of new protocols and evolution of existing ones. Software Defined Networking (SDN) is a recent networking architecture with promising properties relative to these weaknesses in traditional networks. SDN decouples the control plane, which makes the network forwarding decisions, from the data plane, which mainly forwards the data. This decoupling enables more centralized control where coordinated decisions directly guide the network to desired operating conditions. Moreover, decoupling the control enables graceful evolution of protocols, and the deployment of new protocols without having to replace the data plane switches. In this paper, we review recent work that leverages SDN in wireless network settings, where they are not currently widely adopted or well understood. More specifically, we evaluate the use of SDN in four classes of popular wireless networks: cellular, sensor, mesh and home networks. We classify the different advantages that can be obtained by using SDN across this range of networks, and hope that this classification identifies unexplored opportunities for using SDN to improve the operation and performance of wireless networks.

1 Introduction

One of the principles behind the design of the Internet is fully distributed operation where routers and switches control both the network traffic routing as well as packet forwarding [1]. While distributed operation improves the resilience of the network to failures, in practice, it results in networks that are difficult to control. The vertical integration of routing decisions (the control plane) and forwarding decisions (the data plane) complicate network control, and policy implementation. It also makes evolution of protocols difficult, leading to inefficient and fragile networks.

Software Defined Networking (SDN) [2], [3], [4] recently emerged to address these shortcomings by decoupling the control plane from the data plane. For example, software defined and OpenFlow [4] enabled Google's WAN, B4 [5], deployment reveals that poor WAN link utilization (30%-40%) can be enhanced up to 70% (with some links reaching 100% utilization) using SDN to improve dynamic routing and to enable traffic engineering. It is also reported [6] that the North American SDN market was $1.4 Billion in 2014, and is estimated to grow to $4.6 Billion by 2019.

In particular, switching elements in SDN carry out data plane functionality such as packet forwarding, but are controlled and configured by controllers that do not reside on the switches themselves. The controllers serve as logically central control points that collect network state information and make coordinated decisions on how to configure the network resources. SDN embodies a separation of concerns that leads to a cleaner, more efficient and evolvable design. As a result, SDN simplifies the design of switches and routers since they become dedicated only to the data plane. At the same time, SDN controllers directly and intentionally control the network behavior, rather than indirectly attempting to guide emergent behavior of the distributed network resources.

SDN has been investigated primarily in the context of wired provider networks and data centers. However, they are also starting to be explored in other contexts including wireless networks; this paper provides a survey and taxonomy of the use of SDN in different types of wireless networks. For each type of network, we overview, classify and contrast, proposed SDN designs. We also discuss

Israat Tanzeena Haque is partially supported by a Natural Sciences and Engineering Research Council (NSERC) post doctoral fellowship.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

open challenges and opportunities.

We start with Wireless Cellular Networks (WCNs), the class of wireless networks where a significant number of SDN solutions have been proposed. A primary concern in WCNs is efficient resource allocation and interference mitigation to enable scalable deployments. WCN providers often have to integrate multiple technologies like WiFi, WiMAX, and LTE within the same network and support seamless mobility across them. Congestion control and load balancing are further concerns to support increasing traffic demand in WCN. We overview proposed SDN designs that tackle these challenges through a number of techniques including decoupled control logic, virtualization, task distribution, hardware abstraction and data plane programmability. We first outline and contrast the existing work, and then review open research challenges.

Next we consider the use of SDN in Wireless Sensor Networks (WSNs) where the primary challenge revolves around managing the limited resources of the sensors as they collaborate on gathering information about the phenomena being monitored. This limited resources exacerbate scalability concerns with large networks, as the amount of control traffic necessary to maintain an up-to-date view of the network state increases. Sensor networks also require in-network processing for summarization and aggregation. Further challenges include supporting multiple applications and enabling multiple networks to coexist.

The third class of networks we review is Wireless Mesh Networks (WMNs). The complex interference between different mesh nodes can significantly harm the performance and fairness in mesh settings. It is important to control the networks to provide separation between destructively interfering nodes. Typical routing and MAC protocols are often unable to resolve such interference using distributed solutions. SDN's decoupled architecture with centralized control helps to realize these goals in mesh networks. We believe that task distribution also applies for mesh network settings to improve their scalability.

Finally, we consider Wireless Home Networks (WHNs). Typical workloads for such networks include video streaming, interactive games, and large-size downloads. These applications are challenging because they require high bandwidth and have real-time constraints. In addition, home deployments are mostly unplanned and interference between neighboring homes is a significant issue, especially in urban settings. Another concern is Quality of Experience (QoE) aware Adaptive Bit Rate (ABR) streaming. The presence of a centralized controller and the use of SDN principles allow the network to mitigate the interference and congestion using global network view and dynamic resource allocation. ABR streaming can better identify congestion and reconfigure routing path using an SDN-based centralized controller. This also helps to dynamically assign delivery nodes and improve the overall QoE. Furthermore network virtualization or slicing can optimize the resource utilization and home network deployment cost.

Together, we believe that this growing body of work highlights the promise, but also the challenges, in using SDN in wireless network contexts.

The remainder of the paper is organized as follows. Section 2 provides background on SDN, OpenFlow, and network slicing. This section also reviews related work in wired and wireless SDN. Software defined cellular network architectures are discussed in Section 3. Similarly, Section 4 discusses the use of SDN in sensor networks. In section 5, SDN architectures that target mesh networks are reviewed. Similarly, Section 6 introduces SDN works targeting home networks. After the detailed discussion of these four classes of networks, Section 7 presents software defined architectures in other wireless networks such as WAN, ad hoc, vehicular, Device-to-Device (D2D), social, and smart grids communications networks. Section 8 summarizes the main themes and lessons we gained from the existing work, while Section 9 outlines future research directions.

2 Background

In this section, we overview the general architecture of Software Defined Networks, and briefly introduce OpenFlow and network virtualization. We then discuss the role that SDN could play in the wireless networks that we consider. Finally, we present an overview of other recent SDN literature reviews in wireless and wired networks and explain how our work differs from them. In Table 1 we list a set of acronyms used throughout the paper.

2.1 Basic Operation and OpenFlow

SDN differs from the conventional IP networking in that packet forwarding is based on *flows* rather than packets. The flow abstraction is independent of, and can accommodate, various network hardware technologies. The logically centralized controller,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

TABLE 1
A List of acronyms used in this work.

Acronyms	Elaboration
ABR	Adaptive Bit Rate
CDN	Content Distribution Network
CN	Core Network
CSPF	Constrained Shortest Path First
DASH	Dynamic Adaptive Streaming over HTTP
D2D	Device-to-Device
EFTM	Embedded Flow Table Manager
HWMP	Hybrid Wireless Mesh Protocol
IED	Intelligent Electronic Device
LVAP	Virtual Access Point
MAP	Mesh Access Point
MCS	Monitoring and Control Server
MPLS	Multi-Protocol Label Switching
MSN	Mobile Social Network
NFV	Network Function Virtualization
NOS	Network Operating System
OLSR	Open Link State Routing
PANE	PArticipatory NEtworking
P-GW	Packet data network Gateway
QoE	Quality of Experience
QoSen	Quality-of-Sensing
RAN	Radio Access Network
SDN	Software Defined Networking
SDR	Software Defined Radio
SDWN	Software Defined Wireless Networking
SDN-WISE	SDN-WIreless SEnsor network
S-GW	Serving Gateway
SoftRAN	Software defined RAN
SSID	Service Set ID
TCAM	Ternary Content Addressable Memory
V2I	Vehicle to Infrastructure
V2V	Vehicle to Vehicle
WCN	Wireless Cellular Network
WHN	Wireless Home Network
WMN	Wireless Mesh Network
WSN	Wireless Sensor Network

also called the *Network Operating System (NOS)*, performs the network management and control given a global network view. Network applications may be written on top of the NOS, making it possible to support network programmability and to achieve flexible network management, reconfiguration, and protocol evolution. For instance, adding a new network functionality requires a new application on top of the NOS avoiding the need to install new and expensive hardware to support the new functionality as required in conventional networks.

The overall SDN-based architecture is depicted in Figure 1. SDN standards define the Application Programming Interfaces (APIs) among the network applications, the control plane and the data plane. More specifically, the *northbound* API is deployed between the network applications and the controller, whereas the *southbound* API carries control rules from the controller to the data plane entities.

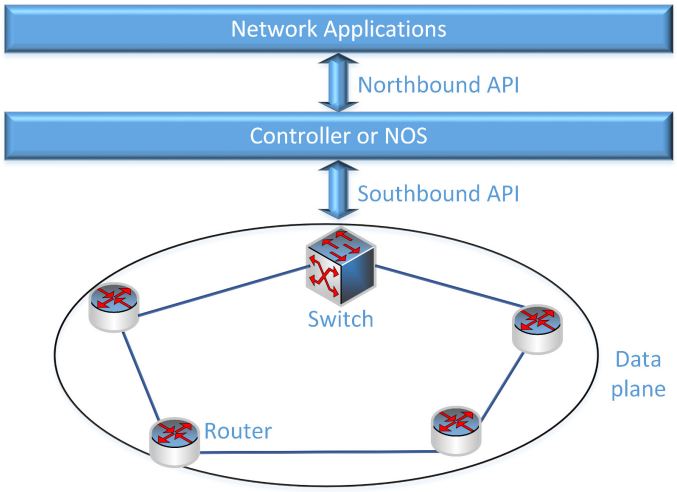

Fig. 1. A Software Defined Networking architecture.

OpenFlow [3], [4] is an example of a standard interface between the control and data plane. It is an open protocol that allows programming the packet forwarding tables (*flow tables*) in OpenFlow compatible switches to manage the network resources. An OpenFlow switch consists of a flow table (a set of rules with associated actions), a secure channel to connect to an associated controller, and the OpenFlow protocol to set up and update its flow table by the controller.

OpenFlow switches do not have packet forward decision-making functionalities; rather, these responsibilities are taken over by the controller. These switches carry out data plane tasks such as forwarding or dropping packets based on the policy specified in the flow table. They can coexist with OpenFlow enabled Ethernet switches and routers. When an OpenFlow-enabled switch receives a packet, it is matched against the stored flow table to perform actions like forward (to a specific port, the controller, or normal processing pipeline), drop, or modify. If no match is found a control packet is sent to the controller for appropriate actions.

The controller can install flow entries reactively after receiving a request, proactively before receiving any requests, or a combination of both. The controller gathers network state and topology information from the data plane elements to make an appropriate decision. The out-of-band secured control channel between the two planes are used for this control packets exchange. However, the performance of OpenFlow based architecture depends

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

on the flow setup time [7], [8]. OpenFlow based controllers include NOX [9], POX [10], Beacon [11], Floodlight [12], and OpenDaylight [13].

2.2 SDN for Wireless Networks–What is new?

SDN has been investigated in the context of provider and enterprise networks, as well as data centers; they are getting increasingly deployed in these spaces. In contrast, significantly less work has considered using SDN in wireless network settings. Wireless networks introduce a number of new challenges to an SDN framework; we overview these challenges in this subsection.

Wireless medium: The wireless network has unique characteristics such as the shared error-prone communication medium, spatial reuse, complex propagation and interference artifacts due to effects such as the hidden and exposed terminal problems [14], [15], and user mobility. As a result, wireless networks are characterized by limited available bandwidth, frequent and unpredictable changes in link quality, and network topology. Efficient resource allocation and interference management require having an expanded, and even global view of the network state. Gathering network status information is challenging given the dynamic nature of this information.

Expanded data-plane functionality: the shared medium in wireless network is often managed by a flexible data plane offering functionality not present in wired networks. In particular, the radio parameters can be configured to manage interference, to optimize access, or to improve energy efficiency. In some wireless networks, in-network processing is also important. In SDN context where a controller manages the data plane, new interfaces have to be exposed to enable effective control of these data plane mechanisms. Beyond data plane reconfiguration, Software Defined Radios (SDRs) [16] allow complete reprogramming of the data plane, creating interesting questions about how to integrate them with SDN.

Supporting network slicing: Another challenge in the software defined wireless network is the implementation of network slicing. Given the complex nature of interference within a channel, it is better to have individual channels (i.e., separate radios) for each slice as it is tricky to share a single channel among slices [7], [17], [18]. Network slicing can effectively support heterogeneous networks since they already use separate radios (e.g., offering seamless connectivity among WiFi and 3G).

Highly dynamic structure: In the wireless environment the network conditions change often, which may take some time to be noticed by the centralized controller. In addition some changes (e.g., uplink frequency in cellular networks or link failure in sensor networks) are local means they do not require any global coordination. Also, the flow setup time will be affected by frequent changes; this is an issue even in the more stable wired networks [7].

In the remaining sections of this paper, we overview more specifically the challenges in each of the four classes of networks we consider, and how SDN can be brought to bear the above challenges. Where possible, we classify the different ways that SDN can assist with respect to each environment. We chose these four classes of networks because they are not only important, but also architecturally different. By overviewing solutions across such different classes of wireless networks, we hope to observe cross cutting themes and conclusions. We hope that these can then inform designs in other classes of wireless networks such as ad hoc , vehicular, and D2D communication networks. We briefly discussed other networks in Section 7.

2.3 Network Virtualization

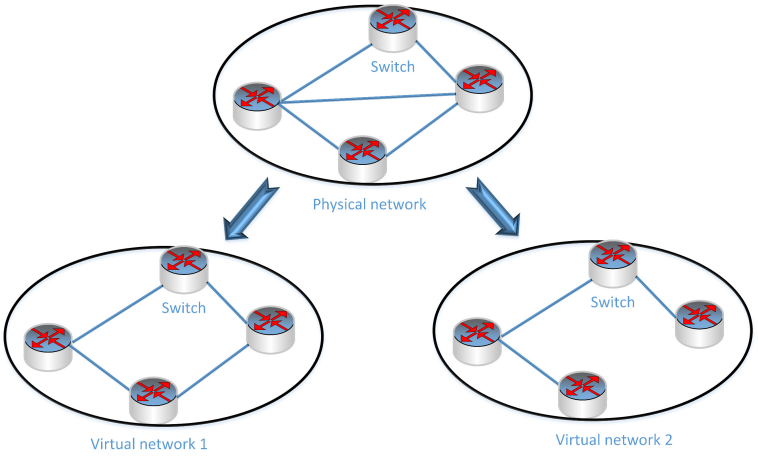

Fig. 2. An example of network virtualization.

Network slicing allows the coexistence of a set of virtual networks within a single underlying physical network. The virtual networks are composed of a set of virtual nodes and links [19]. Thus, the infrastructure provider can manage the underlying physical network for the service providers who own a virtual network or a slice to provide end-to-end network service. These virtual networks or slices managed by individual service provider are

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

isolated from each other while effectively sharing and managing the underlying resources; thus, slicing represents a flexible and effective mechanism that can support a number of resource allocation policies. An example of network slicing is depicted in Figure 2.

SDN can directly express and support slicing. For example, FlowVisor [20] supports network slicing, where each slice is controlled by a dedicated controller. It is implemented as a layer between the controller and the data plane elements to allow multiple controllers coexist on a single shared data plane, while each controlling a slice of the network. OpenVirteX is a similar slicing technique [21]. In contrast CoVisor [22], which is also implemented between the control and data plane, does not restrict each controller to a dedicated slice, but rather allows multiple controllers to process the same shared traffic. The network configuration requirements provided by the administrator are used as the input to CoVisor. This input is used to determine the network policies, create the virtual network for each controller, and define the traffic access policy for the controllers. Based on this principle the standard OpenFlow input from individual controllers is translated into appropriate rule updates for the switches. This design principle allows CoVisor to support multiple control applications written in different programming languages such that different controllers coexist in a single network. For instance, CoVisor accepts a monitoring application written in Python on a Ryu [23] controller and a routing application written in Java on Floodlight [12] in the same network; such functionality cannot be supported in FlowVisor and OpenVirteX [22].

Note that although network slicing is sometimes called network virtualization, it is a different concept from *Network Function Virtualization (NFV)* [24], [25]. NFV decouples network functions (or services) from hardware for better network service management. Candidate functions for NFV include load balancers and firewalls. In particular, network functions are implemented through software virtualization that runs on industry standard servers, storage elements, and switches so that on demand function instantiation is possible without installing new hardware elements [24]. For instance, baseband processing can be virtualized and decoupled from the base stations in cellular networks [26].

2.4 Related Work

A number of general models and surveys have appeared recently for SDN. We present these and motivate our paper relative to them in this section. We organize the presentation into two parts: the first focuses on general SDN, while the second summarizes work related to wireless SDN.

2.4.1 General Models and Surveys

A comprehensive survey on SDN [27] depicts SDN as a layered architecture with the lower layer consisting of infrastructure and southbound API. On top of this layer, there are a network hypervisor, an OS, and the northbound API. Finally, the top layers are language based virtualization, programming language, and network applications. Each of these layers are described in detail along with future research directions, which include efficient hardware design, scalable and efficient controller deployment, SDN migration and security.

Several surveys explore data, control, and application layer architectures of SDN [28], [29], [30], [31]. In particular, several [28], [31] focus on OpenFlow based designs. Hu et al. present a brief discussion on OpenFlow based sensor and mesh network architectures [31].

SDN can bring flexibility to Internet traffic engineering. Akyildiz *et al.* [32] first review the traffic engineering challenges faced by ATM, IP, and MPLS based networks and present a SDN-based attempt to resolve them. They consider challenges including flow management, fault tolerance, topology updates, and traffic analysis. The article reviews existing SDN-based traffic engineering tools from both the academia and industry.

Kobayashi *et al.* [7] describe a four phase deployments of SDN through which a deployed architecture progresses to maturity. The phases are: (1) a small prototype deployment with OpenFlow enabled switches; (2) enhancing the deployment with network slicing; (3) large scale deployment [33]; and (4) production level deployment. A set of interesting findings and insights are learned from this phased deployment including the impact of flow setup time on the performance of SDN; hardware compatibility to support both the legacy and OpenFlow traffic; proper data and control plane isolation among slices; managing overlapping flowspace; impact of the controller placement; and loop avoidance without considering Spanning Tree Protocol (STP).

2.4.2 SDN for Wireless Networks

Most related to our work, two survey articles [31], [34] present a brief discussion on software defined wireless mesh and sensor networks. However, a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

more detailed survey on wireless software defined network can be found in [35], where in addition to OpenFlow based sensor and mesh networks, software defined cellular networks are considered. In contrast, our paper reviews a significantly larger set of projects in detail in each area, and includes areas such as home networks that were not considered in the previous surveys. In addition, we attempt to classify the different projects within each area, as well as glean common crosscutting themes and challenges.

There is a symbiotic relationship between SDN and Software Defined Radios (SDRs). SDRs allow the MAC and PHY layer to be defined by the user, allowing flexible definition of these layers within the radio parameters. Macedo *et al.* [36] argue that the software defined radio, network virtualization, and software defined networks all can be considered as a complimentary solution to each other to form a fully flexible and evolvable programmable network. They overview existing efforts on programmable networks describing their features, weaknesses, and strengths. For instance, a programmable wireless data plane may allow programming of software defined radio architectures (*modal* and *reconfigurable* SDR), platforms, and applications. Jagadeesan and Krishnamachari also reflect on the role of SDRs within wireless SDN environments [35].

Network programmability is also discussed by Hakiria et al. [34] who include software defined radios, security, cloud-based networks and wireless and mobile networks in their review. Programability is the main goal by Reza *et al.* [37] who compare proprietary and open source implementation strategies of SDN in terms of their features, advantages, and disadvantages. Nunes *et al.* [38] present programable networks in the context of SDN. They present OpenFlow and ForCES [39] as the current programable SDN-based architectures. ForCES is different from OpenFlow in deploying the separated control and data plane entities. In OpenFlow the control plane functionality is completely removed from the data plane devices and placed in a separate hardware, whereas, ForCES allows the separated control and data plane coexist in the same network device.

A general architecture for wireless SDN (called *Software Defined Wireless Virtual Network (SDWVN)*) has been recently proposed [40]. SDWVN is composed of *physical network layer* (L1), *controller or virtualization function layer* (L2), and *virtual layer* (L3) to accommodate a heterogeneous multi-technology overlapped network with different service requirements. In the context of SDN, we can say that Open vSwitches operated by an infrastructure provider are in L1, FlowVisors operated by a network operator are in L2, and NOX controllers operated by a service provider are in L3. In addition to summarizing the existing efforts fall under the above architecture, SDWVN also lists a set of unsolved issues like efficient resource sharing among the virtual networks, joint power and bandwidth optimization, and effective interaction between physical and virtual networks.

3 Software Defined Cellular Networking

In this section, we first outline the general design challenges that cellular networks face, and how SDN may be brought to bear on addressing several of them. We then review existing SDN-based efforts organized by how they address these challenges. We conclude the section with open challenges and opportunities in the SDN-based cellular network design.

3.1 Cellular Network Background and Design Challenges

In Wireless Cellular Networks (WCNs), phones as well as smart devices connect to the Internet through base stations mounted on cellular towers. The towers host base stations of two types: Serving Gateway (S-GW) and Packet data network Gateway (P-GW). The gateways act both as control and data plane entities. The data plane functions include access control and traffic monitoring while the control plane tasks include connection establishment, mobility management, routing, radio resource assignment, Quality of Service (QoS) and billing [41]. However, the tight coupling between the control and data planes introduce *resource management* and *scalability* challenges in WCN, in addition to making the hardware costly. For example, in the current design all the data traffic flows through P-GW (even that destined for the same network), consuming scarce wireless resources.

WCNs consist of two primary components, Radio Access Network (RAN) and Core Network (CN). RAN consists of a few nearby cellular towers that provide access to user devices. It provides connectivity between these towers, and eventually connects to the Core Network. In addition, it performs other essential management and resource allocation functionality such as inter-cell Radio Resource

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

Management (RRM), radio Resource Block (RB) control and scheduling, and handoff management. CN provides inter-RAN connectivity and supports services such as mobility, connection establishment and maintenance, and QoS support.

As WCN proliferation has increased, user demand places tremendous pressure on the limited wireless bandwidth; the global mobile traffic is increasing almost linearly [42]. Thus, WCNs must have *effective resource management*. WCNs use a number of techniques to optimize resource allocation to improve the capacity of the network and the user experience. In particular, they use cell splitting, power control and sophisticated channel allocation strategies to support increasing traffic demand [43], [44]. These techniques could benefit substantially from a global network view and coordination to optimize resource allocation and to better manage the interference between nearby cells. Beyond optimizing the use of limited bandwidth, another resource-related challenge in WCNs is meeting user-desired *Quality of Service* for workloads including voice calls and streaming multimedia content.

Supporting *heterogeneous protocols and services* and their *continuous evolution* is a practical and important challenge in WCNs that require malleable architectural designs. The current generation of WCN architectures are not designed with such a philosophy, making it difficult to accommodate heterogeneous technologies such as WiMAX, LTE, and WiFi. Users can connect to one such network at a time without considering capacity and load of the serving network, potentially leading to poor user experience and ineffective utilization of resources. Similarly, the deployment of a new protocol or the evolution of an existing one requires significant amount of hardware and software investments, manual effort, and deployment delays. The same is true about services as WCN providers continue to expand their business model to include the application and services markets.

3.2 SDN-Based Solutions

SDN offers promise in addressing all of the above challenges. Decoupling the control from the data plane removes the burden from the P-GWs to manage the traffic improving scalability and network manageability, while reducing hardware cost. With decoupled control and data planes, a logically centralized controller can perform global resource allocation and interference management, enabling more accurate decisions, and improving performance and stability. SDN-based WCNs can naturally integrate multiple technologies such as WiFi and WiMax through hardware abstraction. In particular, decoupling the control plane allows us to support multiple concurrent control plane technologies within the same network. SDN programmability also facilitates service deployment and extensibility to new services and technologies.

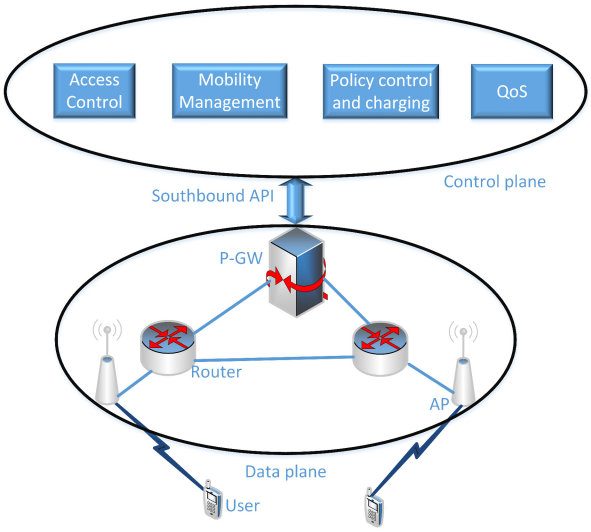

Fig. 3. A general framework for software defined cellular network.

A software defined wireless cellular networking architecture consists of data plane elements and a set of application modules (see Figure 3) to offer QoS, load balancing, mobility management, connection setup, as well as efficient billing system through real-time traffic monitoring. The SDN model also supports efficient network virtualization that can be used for service isolation to achieve the above service goals. Several SDN-based designs for WCN have been proposed in the past few years.

In the remainder of this section, we discuss the application of SDN to solve WCN challenges in more detail. We organize the discussion under different challenge categories. We describe the proposed solutions under the primary challenge they attempt to solve, keeping in mind that *some designs target multiple challenges*. The section concludes with a discussion and comparison of different designs, and an outline of open problems and opportunities.

3.3 Resource Allocation and Scalability

WCNs use cell-splitting to increase capacity in areas with increasing users and traffic demands.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

In particular, cells are split into smaller cells that are each managed by an access point. Each of the smaller cells transmits at lower power enabling the reuse of the available spectrum in a more dense pattern, increasing spatial reuse and ultimately the bandwidth available to customers. However, cell splitting increases interference across cells as the limited spectrum is shared among increasingly smaller and possibly overlapping cells. This interference eventually limits the capacity. In addition, the larger number of cells and their smaller size complicate mobility management and load balancing [45].

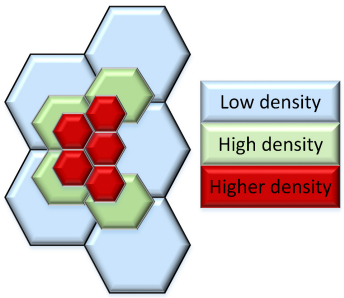

Fig. 4. Cell splitting.

A number of efforts have proposed using SDN with the goal of more effective allocation of resources. We have categorized them into three main groups: efforts targeting RAN (e.g., [45]), efforts that focusing on the CN (e.g., [46]), and comprehensive efforts [41], [47] that consider resource allocation across the RAN and the core network concurrently. In the following we outline these architectures with their primary features.

3.3.1 Resource allocation in the Radio Access Network: Programmable RAN

Efforts in this category target effective resource allocation at the level of the RAN, the edge component of a cellular network that provides wide area access to mobile devices. *SoftRAN* [45] uses a logically centralized radio access control plane to abstract a set of base-stations as a single virtual base-station. Radio resources are abstracted in the space, time, and frequency dimensions, leading to a 3D grid of resources that in principle can be allocated centrally to reach near-optimal resource allocation. However, this design has to tackle the communication delay between the data and control plane even when decisions have to be made quickly. The delay is handled through *task distribution*, where individual data plane elements manage the local control while the logically centralized control plane tackles the global network control aspects.

To illustrate the above principle, operations such as handover management and transmission power allocation are done at the controller as these decisions at a base station need coordination with the neighboring stations. On the other hand, downlink frequency allocation is delegated to the base station as this assignment does not require coordination among neighboring cells. Indeed, such frequency allocation parameters among the cell clients get updated frequently, making it difficult to tolerate the delays of making these decisions centrally. However, the uplink frequency allocation is done at the controller to avoid interference among neighboring cells.

SoftRAN improves handover, interference managed resource assignment (time and frequency at each base station), and effective transmit power allocation for a resource block at a base station for a dense WCN. Different modules of the controller access an information database (RIB) consisting of interference maps, flow records, and operator preferences and use this information to make radio resource management decisions.

SoftMobile [48] offers an alternative task distribution approach that is functional rather than structural. More specifically, they identify three major sub-problems: (1) how to distribute state information; (2) how to configure cells in a coherent way; and (3) how to handle real-time inter-cell operations. They decompose the overall problem into coordinated solutions of these three subproblems managed by three separate controllers.

3.3.2 Programmable Core Network

Resource allocation is also needed at the cellular core network where the RAN connects to the Internet boundary to carry most of the traffic. The traditional cellular network centralizes control of this aspect of the network at the PG-W gateways leading to delay and congestion. In addition, the architecture leads to major failure and scalability limitations, requiring complex and expensive P-GW units in order to meet fault-tolerance and scalability goals. SoftCell [41] removes the complex functionality from the P-GW to the access switches at the base stations. The controller has a global network view and can route the traffic through middle-boxes installed in the switches. However, this design approach requires high state (to support wide varieties of packets) and bandwidth (to support the Internet traffic) requirements which can possibly limit the network scalability. To address this concern, SoftCell introduces a multi-dimensional packet aggregation algorithm to reduce the forwarding table size in the data plane entities. To reduce the bandwidth requirement packet classification is done at the access switches using a local controller and software switches such as vSwitch [49].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

3.3.3 Programmable RAN and Core Network

In subsequent work, *cellular SDN* [41], resource allocation is attempted concurrently in the RAN and CN. Again, SDN principles provide scalability and congestion management issues by decoupling the control and data plane, in this case consisting of switches and base stations, as shown in Figure 5, as well as judicious task distribution. As part of the controller's NOS, a set of application modules that require global network view are managed by the controller. These modules include radio resource management, mobility management, Subscriber Information Base (SIB) tracking, policy and charging rule function, and infrastructure routing. In addition efficient resource sharing is achieved through network virtualization using FlowVisor [20] (a proxy between the controller and the data plane).

The controller delegates simple tasks to the data plane devices through agents to reduce its control burden as well as to improve the response time of requesting events. For example, local agents can perform simple traffic monitoring and react by changing queue priority; such operations do not benefit significantly from controller coordination based on global network view, but require fast response time. Data plane elements can also perform flexible traffic classification to help scheduling, routing, and intrusion detection. Packet compression can also be done at these elements to improve the performance of low-bandwidth links.

Promoting QoE via managed Device to Device Communication: An extension of the above architecture takes the control task one step further: in addition to controlling the decoupled data plane (core and RAN), they consider installing an agent at the user device for a better QoE [50]. The architecture enables D2D communication between end users along with a centralized controller for a better resource allocation, which promotes better QoE (ease/difficulty of the Internet access from an end user perspective). The architecture is similar to Softcell except for the D2D communication among the end users for a better QoE.

Programmable RAN and core network for 5G: SoftAir [47] is a SDN-based wireless network architecture for 5G. The data plane of SoftAir consists of Software-Defined Radio Access Network (SD-RAN) BSs and SDN capable CN switches. The control plane has two components: network management tools and customized applications for service providers and network operators (see Figure 5). The programmable core network supports efficient

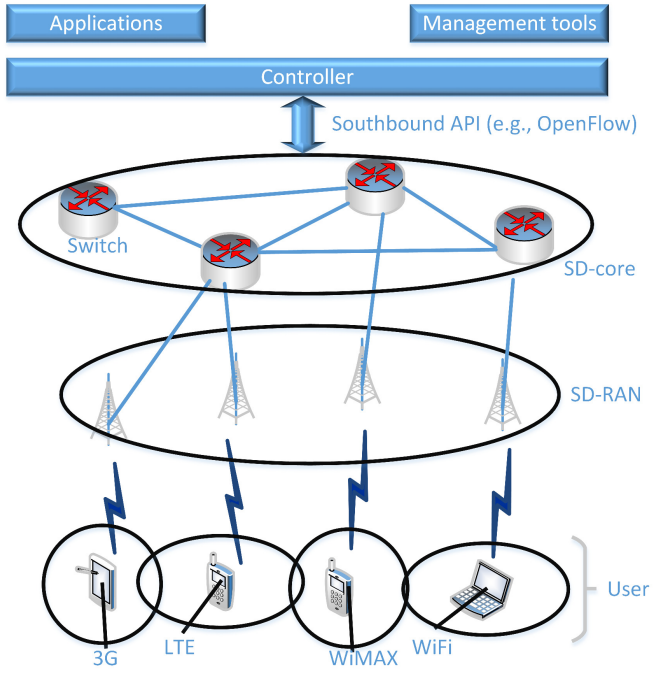

Fig. 5. An example SDN-based architecture for cellular networks.

network virtualization and traffic classification. In addition, decoupling the control task from the programmable RAN enables radio technology evolution at reduced cost.

The architecture uses three-way virtualization of the network, the physical wireless medium, and the switches. Network virtualization ensures effective and non-conflicting allocation of network resources (e.g., spectrum, power) among the service providers and network operators. Wireless virtualization allows low-level wireless resource sharing via control over scheduling. Finally, switch virtualization supports low-level bandwidth slicing using FlowVisor [20].

The traffic classification in SoftAir is decomposed into local classification, performed by the base stations, and global classification, performed by the controller. This efficient traffic engineering along with a virtual Software Defined-Base Station (SD-BS) (formed out of a set of radio resources) ensures effective radio resource and energy utilization.

Mobility management in SoftAir is done in two-steps: location management and QoS guaranteed handoff-rerouting. Low-cost location management can be achieved through unified multi radio technology access. Thus, operations such as cell association and paging are abstracted such that they can be done without inter-networking among technologies. QoS aware handoff is performed exploiting

the centralized controller's global network view to dynamically configure the rerouting path enabling seamless inter-BS mobility. The mobility pattern of the users along with the optimal controller placement determines the optimal in-band control traffic path from the data plane entities towards the controller such that the overall system delay is minimized.

MyNET [51] is a similar architecture to SoftAir designed for 5G network. The most interesting difference in the architecture is that MyNet implements hierarchical deployment of controllers, each responsible for part of the network, to reduce the control traffic flow and improve responsiveness.

Yazici *et al.* propose a hierarchical architecture called CMaaS [26], which consists of a four layer controller hierarchy. The bottom *UE controller* manages the available radio access technology selection for a user constrained by the local network status and higher level controllers' policies and mandates. The next controller above UE is the *BS controller* that controls the time-sensitive radio resource management and scheduling with a local network view. Above the BS controller, the *RAN controller* manages a set of Basestations with a regional view. Finally, the *Network controller* has a global network view and manages services such as QoS, routing, mobility management. Based on this global view, it passes instructions to the lower controllers.

In CMaaS, control is distributed such that a lower layer controller's function is constrained by the upper layer decisions. At the same time, the upper layers acquire network state from underlying controllers to collect a global view and make control decisions. Similar RAN programability is also suggested in SoftRAN, but instead of hierarchical controller decomposition the controller delegates time-sensitive tasks to a local controller (task distribution).

3.4 Protocol Heterogeneity and Evolution

Another group of works in WCNs use SDN to support protocol evolution and heterogeneity. In this respect, the decoupling of the control plane allows the controller to support different protocols on the same data plane.

3.4.1 Protocol Evolution

Motivated by the difficulty of re-engineering networks to keep up with protocol evolution, *OpenRadio* [52] proposes to use SDN to build cellular core networks that more effectively support such evolution. The main idea behind OpenRadio is to systematically support different protocols (3G, 4G, or WiFi), while optimizing operation across all of them. By using modular abstractions, OpenRadio is able to evolve protocol elements by upgrading the control plane operation, often by upgrading software, without replacing the controller hardware, or the data plane elements. OpenRadio further provides a programmable interface between the processing and decision components of the wireless protocols. This approach helps WCN managing inter-cell interference as well as provides support for QoS and incremental protocol evolution and standardization. To reach this goal OpenRadio has to meet the computation and response demand of the WCN protocols; the authors verify that they achieve this goal through an initial prototype implementation.

3.4.2 Supporting Multiple Co-located Technologies

OpenRoads [17], [53] uses SDN as a bridge between technologies like LTE, WiFi, or WiMAX to support a seamless user mobility. Supporting multiple co-located technology can increase the capacity and coverage of the network taking advantage of the available technologies. OpenRoads supports this functionality through network virtualization. In particular, three levels of virtualization are used: (1) OpenFlow separates control from the data plane; (2) FlowVisor is used to slice the network in different dimensions to support concurrent resource sharing; and (3) SNMPVisor is used to configure the infrastructure.

The controller consists of a network OS, NOX [9], and executes modules such as routing, mobility management, and billing; any required application module can be added to this framework. Network slicing is used to support multiple services coexisting in the same network. Each slice is managed by a separate controller. Slicing is accomplished through FlowVisor, which allocates a slice to a designated controller. A hierarchical slicing is also possible to manage a large network. Similar to FlowVisor, SNMPVisor slices the configuration of the data path, by controlling the operation of functionality such as the transmission power, channel allocation and interference control. Recall that SoftAir [47] also performs a similar three-way virtualization and hardware abstraction to support multi-technology.

OpenRAN [54] also relies on the virtualization to support heterogeneous technologies. Operator service-level virtualization is performed at the application modules to separate protocol specific

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

TABLE 2
A comparison of the software defined cellular networks

ARCHITECTURE	FEATURES	DATA PLANE	VIRTUALIZATION	OPENFLOW COMPATIBILITY
SoftRAN [45]	Resource allocation through task distribution.	RAN	Virtual BS	No
SoftMobile [48]	Resource allocation through control abstraction.	RAN	Yes	No
SoftCell [46]	Scalability	Core network	vSwitch [49]	No
Cell SDN [41]	Scalability through task distribution.	RAN and core	FlowVisor [20]	Yes
D2D-LTE-A [50]	QoE	RAN and core	No	No
SoftAir [47]	Scalable 5G architecture.	RAN and core	3-level virtualization	Yes
MyNET [51]	Data analytics, content delivery, and D2D.	RAN and core	SONAC	No
CMaaS [26]	Hierarchical decomposition of controllers.	RAN	Yes	Yes
OpenRadio [52]	Protocol evolution.	Core network	No	No
OpenRoads [53]	Multi-technology and evolution.	RAN and core	3-level virtualization	Yes
OpenRAN [54]	Multi-technology support.	RAN	4-level virtualization	No
SDWN [55]	QoE through virtualization.	RAN	3-level virtualization	Yes
CROWD [56]	Capacity and energy.	RAN	No	Yes

flows. There is also computing and storage virtualization through cloud computing to manage these resources. OpenRAN adds an extra computing and storage management virtualization level through the cloud, which is a feature not present in OpenRoads.

The authors of *SDWN* [55] argue that SDN can enhance network performance by dynamic and efficient virtualization that takes into account the current network states and reacts accordingly. While SDWN is fundamentally similar to OpenRoads and OpenRAN, one interesting difference is that SDWN attempts to improve QoS and QoE for users through dynamic traffic configuration and RAN programmability.

CROWD [56] supports efficient interference and mobility management in dense cellular networks through hierarchical controller deployment. The logically centralized global controller CRC (CROWD Regional Controller) performs long-term optimization using the aggregated network data captured through the local controller CLC (CROWD Local Controller). For instance, interference among BSs is taken care at the CRC using a multi-tier scheduling. CRC also uses the gathered network state information to dynamically manage the cell association to guarantee better user experience and energy optimization. In addition CROWD supports seamless mobility among technologies like LTE and 3G through the local and global coordination among CLCs and CRC.

3.5 Learned Lessons and Open Challenges

Table 2 summarizes the SDN-based solutions for cellular networks. We can identify a range of objectives like efficient resource allocation and management, scalability, QoE, supporting various types of networks, D2D communication, or energy optimization. The solutions target RAN, core, or both of them as their data plane components while designing a software defined architecture with a decoupled controller. Some solutions also consider existing OpenFlow architecture as their predecessor. The network slicing or virtualization is the technique adopted in most of the solutions to achieve their objectives. As each solution has its unique features, one can be a complementing solution to others. For example, both the cellular SDN and SoftRAN consider the task distribution but with different objectives (scalability and resource allocation), which could be brought together by combining these two architectures. However, we see further research challenges under this network, which are briefly outlined below.

Among the four network types we present in this work, the SDN-based cellular network is the most mature one in terms of solving the design challenges in the classical WCN. Moreover, because these are commercial networks, there are barriers to experimentation and data collection. Nevertheless, there remain opportunities to leverage SDN, for example to enhance fault tolerance in the presence of controller failure and improving robustness. In fact we argue that though centralized control attempts are made in every stages of cellular network architecture ranging from RAN to end-users, a generalized architecture for SDN-based cellular network that clearly outlines design approach of various components is needed. Such an architecture would allow specification of task distribution (local, vs. global, vs. hierarchical). Furthermore a

benchmark scenario for evaluation and quantitative comparison of different SDN based solutions would allow researchers to effectively evaluate innovative ideas in this space.

QoE and cellular network service optimization are not well investigated other than a few initial attempts, introduced as an additional feature in more comprehensive architectures. Future cellular network are moving from an emphasis on infrastructure to one on services and applications. However, most of the existing solutions focus on the infrastructure optimization rather than considering service level performance. Finally, we believe that network and service performance can be further enhanced through a self adapting architecture, where technique like temporary cell splitting or turning off idle BS can be considered to support busy-hour traffic demand or energy optimization.

4 SDN in Wireless Sensor Networks

Wireless Sensor Networks (WSNs) [57], [58] differ substantially from WCNs with respect to network and node architecture, traffic characteristics, scale, and design goals. In this section, we first outline the primary challenges in a WSN, and how SDN can be used to address them. We follow by overviewing existing efforts in this space. In the end we list a set of open issues in the software defined sensor networks.

4.1 Sensor Network Background and Design Challenges

Wireless sensor networks are typically composed of a set of resource-constrained sensors that are deployed to measure a phenomena of interest. Sensor networks can dramatically improve our ability to monitor the world at unprecedented precision and scale. They have applications domains such as target tracking [59], habitat monitoring [60], and many others. They also often form the sensing component in the emerging domain of cyber physical systems [61]. *Energy efficiency* is crucial for most WSN deployments because the sensors are battery operated. WSNs are fundamentally application oriented networks; in-network processing for aggregation and summarization is a common pattern unique to this class of networks. In light of the emphasis on energy efficiency, distributed protocols often cannot successfully reach effective network operating points.

Coverage and topology management is another design challenge in WSN: how to have a connected topology covering the target/survey area. Limited energy, environmental challenges, and radio range necessitate a multihop topology for most deployments. Wireless communication over low-power radios, sometimes in difficult physical settings can lead to frequent packet-transmission errors and link disconnections. Often redundant sensors or mobile sinks are deployed to improve the coverage and topology, but further complicating the protocols.

Often, sensor networks are *application specific*, with vertical integration of the software stack, and sometimes even the hardware, such that they are specialized to the application. In some scenarios, sensing infrastructure can be tasked by multiple applications that co-exist within the network. In such settings, efficient multi-application support is needed. Also, sensor nodes can be installed with multiple sensing components to sense modalities such as ultrasonic, photoelectric, or temperature. This is a common situation supported by sensor nodes software stacks, including TinyOS [62]; however, effective abstractions to support a multi-modality multi-application environments are needed.

4.2 SDN-Based Solutions

WSNs architecture relies on one or more centralized base station/sink to task the sensor network and to gather the data. As such, it naturally maps to the SDN model with the controller being centralized at the sinks. With the control centralized, the sensors can simply become data plane elements forwarding and processing data along the way. Sensors are freed from network control tasks such as routing and topology management, simplifying their architecture and improving their energy efficiency. On the other hand, armed with a global network view the controller can offer efficient resource allocation and management through centrally controlled topology control, scheduling, routing, and network coverage and connectivity planning.

Using network slicing and hardware abstraction, multi-application sensor networks with different hardware can be accommodated under a single physical network architecture. Data plane programmability is another benefit of SDN, allowing the support of diverse applications and environments. Protocol evolution can be supported directly, by upgrading the control plane implementation at the sinks. An efficient multi-application sensor network can be realized supporting dynamic applications and integrating sensor of different hardware and sensing capabilities.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

While the overall SDN organization fits WSNs naturally, sensor networks differ in a number of ways from conventional networks for which SDN were designed. The traffic in sensor networks follows a gather (or reverse multicast) pattern with in-network processing (data aggregation [63]) at intermediate nodes. Moreover, sensor nodes are embedded devices that are unlikely to be imbued with a separate physical interface for control. Thus, SDN based wireless sensor network design must consider that both the control and data traffic will flow through the same network topology, which is dynamic, resource poor and unreliable. Additionally, sensor networks are often queried and addressed using data centric approaches, rather than by IP addresses. Thus, supporting WSN requires extending SDN to accommodate the above requirements. In addition we need to design software enabled sensors supporting new architectures .

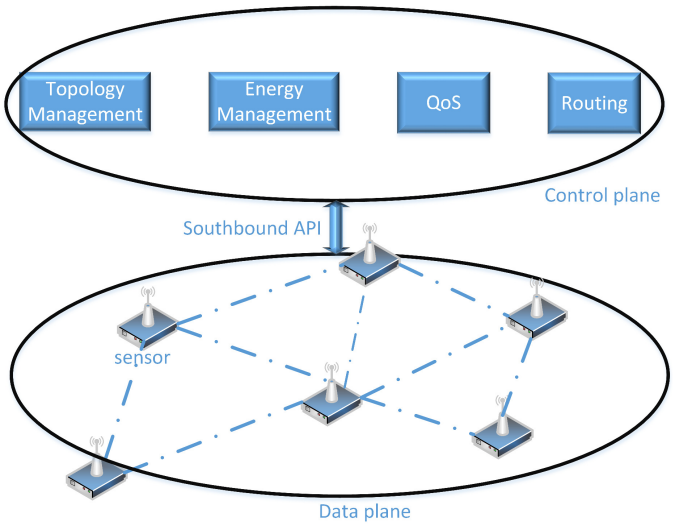

Fig. 6. A software defined wireless sensor network architecture.

A SDN-based WSN framework with a centralized controller is shown in Figure 6. Different network functions, such as routing and topology control, can be supported in this architecture as a control module residing on the logically centralized controller (typically on the sinks). Applying SDN to operate WSNs remains a new topic with only a few exploratory studies; we believe that there are significant open opportunities. We first summarize work in this area, and then discuss remaining challenges and opportunities.

4.3 Extending SDN for WSNs

Due to the differences between WSNs and conventional wireless networks, SDN must be extended with additional functionality to be able to support WSNs effectively. A number of position papers discussed such extensions in principle including SDWN [64], and Smart [65]. Sensor OpenFlow [66] offers a concrete design of an SDN extension to support WSNs [66]. It adds new classes of forwarding rules to OpenFlow to fit the requirements of WSN. It also supports energy optimization through efficient duty-cycle control [67]. Sensor OpenFlow forwarding rules support data aggregation [63]. Additional goals of Sensor OpenFlow include support for protocol evolution and multi-application operation.

The flow in typical SDN is address centric, which is not the case in WSN, where we gather data with specific interest. Sensor OpenFlow supports creating the flow tables by using *IP* alternatives for low power devices (such as *uIP*, *uIPv6* [68] or *Blip* [69]). In this case, an OpenFlow control channel between the control and data plane can use exiting *TCP/IP* support. Alternatively, Sensor OpenFlow supports addressing by appending a new matching function in OpenFlow that works on compact addresses and attributes; this support enables user-customized transport protocols.

Another design challenge in Sensor OpenFlow is that the same network is used for both the control and data traffic, whereas, in a typical SDN scenario a dedicated control channel is used by the controller to communicate with the data plane. Usually, when the controller receives a regular sensor request for a new flow table entry in response to a new incoming packet, the controller sends back the response with an expire time associated with that entry. As the topology of WSN changes frequently and usually a single or a small number of sinks handle such control traffic, a burst of requesting packets may flow between the sensors to the sink. To limit the request rate in response to changes in topology, the sensor is not allowed to request a flow entry for the same destination address until it gets back the response or a timeout occurs.

An extension of Sensor OpenFlow, called SDN-WISE [70], [71] leverages SDN to simplify policy implementation within a reconfigurable, and vendor independent WSN. Regular sensors participate in local control tasks without interaction with the global controller, which makes SDN-WISE a stateful OpenFlow based solution. Multiple controllers are supported through slicing where one controller serves as a proxy between the data plane and rest of the controllers. Thus a packet may follow different flow rules for different controllers as per the application requirements.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

Within SDN-WISE, the SDN-enabled sensors and controllers are extended with the required data structures to manage the states (active or not), accepted neighbor IDs (whose packets will be accepted), and flow table. The control messages from sensors to controller rely on local routing protocol (called topology discovery (TD)) learned by the sensors. Also, this learned local topology is periodically sent to the central controller to have a global network view. The centralized controller sends periodic message to let the sensors know about the best next hop node towards the controllers.

Pfammatter *et al.* design a software enabled sensor [72] mainly for wideband spectrum monitoring. The architecture consists of a controller, collector, and sensor. The controller delegates sensing tasks to the sensors as per the requirement of applications. In addition, it maintains the network state information to make necessary changes in the assigned sensing tasks. The collector preprocesses the gathered data from the sensors before passing them to the controller. The main application of this architecture is to monitor spectrum use. However it is possible to deploy these sensors for other sensing task and integrate them with the software enabled sensor network system designs.

4.4 Resource Allocation and Management

A second class of SDN proposals for WSNs focuses primarily on using SDN to control resource allocation. For example, *SDWN* [64] targets efficient duty cycling (turning off the radio while not in service), data-aggregation, and flexible routing rules for cross layer optimization through a decoupled architecture. Their SDN enabled architecture for the WSN installs the controller at the sink. In regular sensors, the protocol stack appends a forwarding layer on top of the physical and MAC layers, which consists of the flow tables. This layer also supports the data aggregation for the aggregation layer on top of it. The NOS is layered on top of these but under the application layer and manages all the local actions instructed by the controller. The sink is similar to a regular sensor with an embedded system that serves as the controller. The layers include an adaptation layer (for message formatting), a virtualization layer (slices the network in terms of the topology, which is also formed by the same layer), a controller (creates flow table rules based on the current topological knowledge), and an application layer.

Using the above architecture SDWN defines simple data aggregation and flow table rules. However the controller and general sensors need to exchange messages to create the network topology at the controller and to learn the path information from sensors to the sink. It is unclear whether integrating SDN in WSN as per this architecture results in minimizing energy use: the design still requires message passing between the control and data plane that may not be worth for the resource constrained sensors.

Smart [65] is a similar solution that proposes a controller architecture for better WSN management. The controller resides on the sink and comprises a five layer stack. The lower three layers are the physical, MAC, and NOS layers. The next layer up is called the middleware where the controller sits. The authors claim that through this centralized architecture, tasks like routing, QoS, mobility management, and localization can be managed by the controller leading an energy efficient solution as the regular sensors will be free from the communication and processing burdens.

The architecture in Smart uses a localization service that resides on the application layer for location-based routing. The solution has the same limitation of not addressing the overhead due to the control message flowing through the routing network to the controller.

Zeng *et al.* propose a SDN-based architecture for multi-tasking WSNs [73], [74]. Multi-tasking allows a sensor network to support multiple applications that share the same network; each sensor has multiple programs corresponding to these applications. Multi-tasking can efficiently be realized using SDN based architecture. Optimization across different applications (including energy optimization) can be accomplished through the globally controlled scheduling and Quality-of-Sensing (QoSen). In particular, the architecture defines the multi-application objectives as a Mixed Integer Linear Program (MILP) problem of minimum energy sensor activation problem with the constraints on coverage, storage, and scheduling. In addition, an online algorithm is designed to take into account the applications and sensors dynamics: newly required sensing application as well as departing or arriving sensors can be accommodated in the designed system. The simulation results reveal that the solution obtains almost the same energy use as a global optimization but with less rescheduling time and control overhead.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

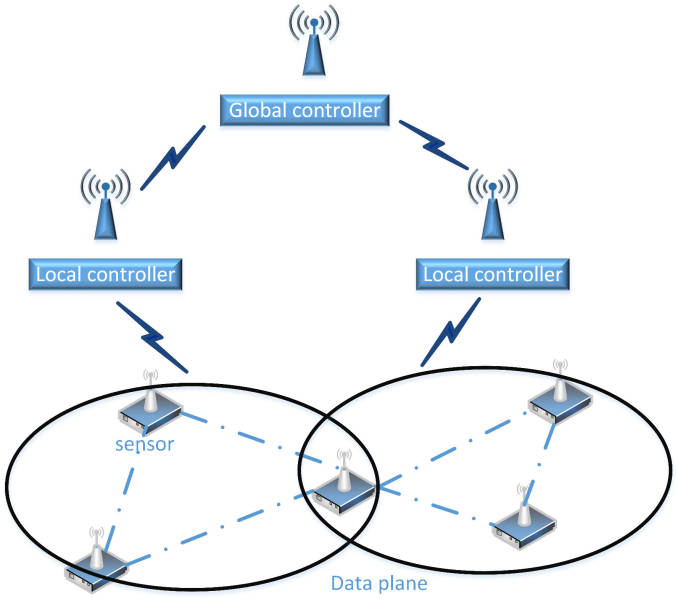

Fig. 7. An example of hierarchical architecture.

4.5 Hierarchical Scalable Architectures

As a sensor network scales, it is inefficient to support it using a single base station/controller from the perspective of not only data gathering but also for centralized control. Therefore, some projects propose using multiple controllers to enhance the performance of SDN enabled WSNs. Multiple controllers can be used either to provide scalability and reduce the need to communicate control messages centrally, or to specialize the network to different ongoing semantic contexts.

Spotled [75] uses hierarchical controllers to reduce the communication overhead relative to a centralized controller architecture. In particular, local controllers manage a part of the network and inform the global controller about the topological and other state changes (see Figure 7).

Flow Sensor [76] proposes a hierarchical controller organization to cluster the sensors according to their gathered data type or context. Sensors that are from the same context form a cluster even if they are physically distant. Each such cluster has its own controller or cluster head that performs the local processing for the cluster. The local controllers form a logical controller for the entire network. Thus, this architecture focuses on managing sensed data through hierarchical control and the clusters in Figure 7 will represent sensors sensing context-aware data that may not physically reside in the same locality.

4.6 Learned Lessons and Open Challenges

We summarize the discussed software defined sensor network design in Table 3. Sensor OpenFlow is the first attempt to bring SDN in WSN with customized rules and architectures. In particular the design includes flow setup and rules management, in-band control traffic management, and in-network processing. This is the only architecture that offers options for both in-band and out-of-band control traffic management. SDN-WISE performs measurement with a real-life deployment. However, it does not provide any measurement on duty cycle and data aggregation. Both SDWN and Smart propose a high-level layered architecture for efficient resource allocation and management. SDWN proposes a simple data aggregation approach that is not present in Smart.

Spotled is a hierarchical solution that decomposes the network into a set of clusters each associated with a local logical controller, which are finally managed by a global logical controller. This decomposition allows better in-band control message management that will lead to a scalable network. Scalability is also considered in Flow Sensor (in addition to reliability and reachability). Flow sensor uses content based clustering rather than location-based clustering.

The work in this area is in a nascent stage and generally lacks evaluation using representative deployments or even detailed simulation. A possible exception is SDN-WISE, which is the first attempt of deploying a testbed for performance measurement. There is a need for deployments, models and benchmarks to allow deep evaluation of the utility of using SDN in WSNs.

We also believe that there is a number of remaining opportunities and open challenges. One of the design objectives of Sensor OpenFlow [66] is supporting multiple applications sharing a single physical sensor network. Although a design goal, it is not clearly addressed in Sensor OpenFlow. SDN principles can be leveraged to slice the network among the applications. These slices can be centrally coordinated by the controller. The proposed designs did not consider a number of other core capabilities for WSNs. These include support for *mobility*, *topology management*, scheduling control and data traffic, *security*, and *robustness* to sensor and controller failure.

Scalability is an important design issue in WSNs and researchers proposed a plethora of solutions along this line. SDN based WSN also needs to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

TABLE 3
A comparison of the software defined sensor networks

Architecture	Features	Control channel	Data Aggregation	OpenFlow compatibility
Sensor OpenFlow [66]	Multi applications support.	in-band and out-of-band	Yes	Yes
SDN-WISE [70]	Duty cycle and data aggregation.	in-band	Yes	Yes
Multi-task SDSN [74]	Dynamic multi-tasking.	in-band	No	No
Software Sensor [72]	Software enabled sensors.	N/A	N/A	N/A
SDWN [64]	Data aggregation and routing.	in-band	Yes	No
Smart [65]	Resource allocation and management.	in-band	No	No
Spotled [75]	In-band control traffic management.	in-band	No	No
Flow Sensor [76]	Content based logical clustering.	in-band	No	No

account for scalability. This issue is raised in Smart, where the authors argue that a decoupled architecture can naturally promote scalability in SDN-based WSNs. However, we argue with such claim as the communication overhead will increase with the increasing number of sensors in the network with a single controller. A hierarchical architecture of multiple controllers may help reducing such overhead as pointed in Spotled, without adequate evaluation of such proposal. In such a design, an open problem is how to choose the number and placement of controllers, and the decomposition of control among them.

In addition to supporting multi-applications, SDN-based design can also help *protocol evolution* through data plane programmability. This issue is yet to be explored. Finally, none of the above work offers a complete system design for a SDN-based WSN with sufficient evaluation. We believe that there is significant need for further exploration of SDN in this space.

5 Software Defined Wireless Mesh Network

In this third class of networks, we follow a similar presentation where we first outline the primary challenges faced by this class of networks, and discuss how SDN can address them. We follow by reviewing existing works in this space. Like WSNs, there are only a few efforts, leaving a number of opportunities for exploiting SDN to more effectively build, optimize and manage mesh networks.

5.1 Mesh Network Background and Design Challenges

In Wireless Mesh Network (WMN), a set of wireless routers form a backbone to offer the Internet access to clients that connect to it [77]. Although it is possible to have a single channel mesh network, most commonly mesh networks use multiple interfaces/channels to improve performance and enable separation [78], [79]. Typically, some of the mesh routers have interfaces that connect to the Internet (gateways); traffic to and from mobile clients that are connected to routers without the Internet access must be forwarded within the mesh backbone to reach a router with the Internet access.

Since a limited number of routers act as gateways, congestion can arise. Thus, efficient *resource allocation and management* is crucial to maximize the fairness and capacity: *congestion control*, *load balancing and traffic engineering*, and *mobility management* are important. It is important to identify the congested gateways and routers and distribute the traffic to alternate gateways and routers that have remaining capacity.

5.2 Mitigating Challenges using SDN

SDN, with centralized control based on a global network view, can be used to efficiently tackle the above design challenges. In particular, the centralized controller (as shown in Figure 8) can offer better resource allocation and management to avoid congestion and to distribute the load among the routers. The global network view and direct control of the network operations can also help to better support mobility management and energy efficient operation.

Reliance on a single controller to manage the entire network can compromise network reliability in SDN: if the network gets partitioned or the controller fails, the entire network will suffer. Thus, any SDN-based solutions, irrespective of the network type, must consider the network *fault tolerance*.

5.3 Efficient Resource Allocation and Mobility Management

The first group of projects uses SDN to implement more effective resource allocation and mobil-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

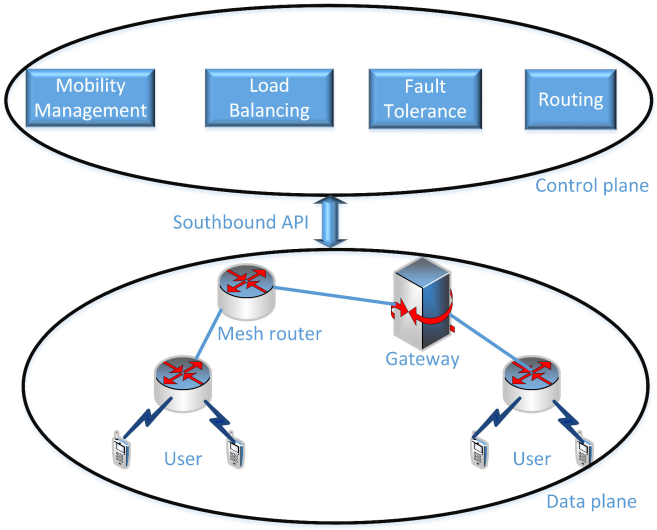

Fig. 8. A software defined wireless mesh network architecture.

ity management in mesh networks. *Mesh Flow* [80], a representative of this class, has two goals: (1) flexible routing to improve the performance and load balancing within a mesh network; and (2) flexible and efficient client mobility. Mesh Flow implements an OpenFlow based mesh network architecture that consists of OpenFlow enabled mesh routers. These routers have multiple physical interfaces to connect to an access network, other routers, or the Internet. Each physical interface is decomposed into two virtual interfaces to support isolated control and data traffic. Each virtual interface is assigned an unique Service Set ID (SSID).

As the network topology changes often a large amount of control traffic flows through the network to update the topology. OLSR [81] routing is used for this control traffic to reduce the communication delay between the controller and data plane elements. Thus, the control interface is connected to the IP routing daemon OLSRD, whereas the data interface is connected to OpenFlow controller. Each router also has a local monitoring agent to keep updated topological information that can be accessed by the controller. An end station uses standard IEEE 802.11 to connect to a Mesh Access Point (MAP), it also consists of a local agent that is accessed by the controller to control and manage the network association and handover.

The primary advantage of Mesh Flow results from having data traffic be controlled by the centralized controller, through OpenFlow, to optimize resource allocation. In contrast, typical routing protocols such as OLSR, ignore link quality, and favour shortest path routing, resulting in poor performance, congestion and hot-spotting.

Mesh Flow also targets optimizing handoffs. In particular, the controller consists of a Monitoring and Control Server (MCS) and a NOX. MCS queries the local agent that resides in the routers and end stations to form an updated topology. In addition, it maintains information for association and handover for the end users. The NOX creates the flow table for routing and manages mobility with the help of MCS. The performance of the proposed architecture is evaluated in KAUMesh testbed [82] to show that SDN-based mesh network can offer efficient routing, load balancing, and handover through efficient resource allocation. However the current implementation of mesh Flow manually triggers this handover and needs an algorithm to automatically identify an end station and potential MAP for mobility management.

Gap is an extension of mesh Flow [83] which supports efficient resource allocation through a flexible routing scheme. In Gap, a mesh router consists of multiple virtual interfaces each associated with an unique SSID to support OpenFlow data, 802.11s control traffic, and ad hoc connectivity to other routers. In particular, two physical interfaces of a mesh router are decomposed into four virtual ones (rather than into two virtual interfaces in mesh Flow), using OpenWRT [84], [85] to support OpenFlow and 802.11s. One of the virtual interfaces serves as a control channel to forward traffic using the Hybrid Wireless Mesh Protocol (HWMP) [86]. Two more interfaces are used to connect to the access networks and the fourth one goes to the backbone.

Similar to Mesh Flow, the Gap controller controls routing as by setting the flow rules in a way that leads to effective resource allocation. These rules are added to OpenFlow to make the proposed architecture suitable for mesh environment.

Yang *et al.* [87] propose a SDN-based network architecture for load balancing in WMNs. In particular, load balancing is accomplished using a network management tool that gathers the network status to track the current traffic condition. Based on this information, the controller can reconfigure the routes dynamically to avoid the congested routes towards the Internet gateways. In the case of gateway congestion, alternate gateways may be chosen for some flows. Load balancing requires in-band control message exchange between mesh routers and the controller.

Huang *et al.* [88] consider the problem of self-interference between control and data traffic in a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

SDN-based mesh network. In a frequently changing wireless network, the volume of the control traffic is high to maintain an up-to-date network topology. This traffic also has a spatial variation as the links near the gateway may expect more data than control traffic, while the links near the controller may expect more control than the data traffic [88]. To effectively manage this spatiotemporal traffic variation, an OpenFlow based mesh architecture is proposed in [88]. The controller gathers network status for dynamic routing, scheduling, and spectrum allocation based on the current control and data traffic volume. The mesh routers host a local control module to gather the topology information to be exchanged with the controller. They also have a radio spectrum tuning module that is operated as per the controller's instruction. This radio frequency is tuned using Software Defined Radio (SDR) to avoid the requirement of multiple interfaces. The communication between the data and control plane is performed through an extended OpenFlow for mesh networks.

The spectrum allocation and scheduling problem in [88] is defined as a weighted throughput maximization problem, where control traffic is assigned higher weight compared to the data. Three algorithms are designed: (1) Fixed-band non-sharing spectrum (FB-NS): a fixed band is allocated for each link in such a way that the assigned spectrum may go unused even if other traffic may require further spectrum. For instance, control traffic cannot use the data traffic spectrum even if there is currently no data to send; (2) Non-fixed band algorithm (NFB-NS): spectral resources are not partitioned but rather allocated in such a way that the control traffic gets the higher priority. The above two approaches do not share the assigned spectrum between the data and control in the sense that data traffic uses only the data-spectrum and control traffic uses the control-spectrum; and (3) To further share spectrum in the NFB-S, data traffic can utilize the unused spectrum after control traffic is sent.

5.4 Supporting Fault Tolerance

One of the issues with SDN is that the use of centralized control may compromise network reliability since failures or partitioning can leave nodes without access to a control plane. Supporting fault tolerance requires adding mechanisms to detect loss of connectivity to the controller, and activating an alternative mechanism for implementing the control

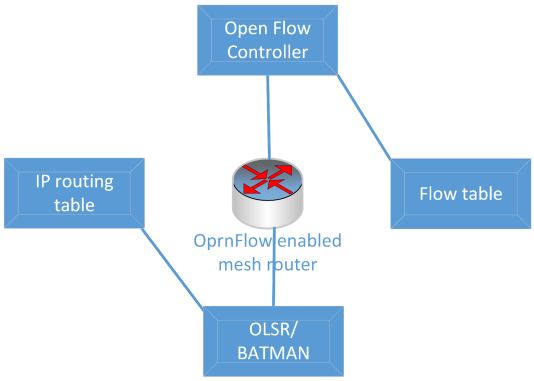

Fig. 9. A mesh router with IP routing and flow tables.

plane (for example, an alternative controller, or a switchover to distributed control).

The *wmSDN* project [89] extends mesh Flow [80] to support uninterrupted operation under controller failure, by supporting failover to a backup distributed control mechanism. wmSDN uses a single SSID for the control and data traffic instead of using two SSIDs as used in mesh Flow. The architecture consists of a set of Mesh APs (MAPs), one of which is connected to the centralized controller using a wired or wireless connection. Usually a MAP has two optional wired interfaces (for access networks and the Internet), one or more wireless interfaces to connect other routers, and a virtual switch, such as vSwitch, for OpenFlow.

The control traffic rules are managed by a local controller from inside a MAP using OLSR and an entity called OLSR-to-OpenFlow (O2O), where the control and data traffic use different subnets. OLSR is also used to manipulate the switch-table in the case of controller failure due to network partition or malfunctioning. In addition to the data traffic rules, the flow table contains rules for the OLSR control traffic. The data traffic rules is manipulated by the central controller that accesses the connected MAP to get the topology. In the case of controller failure O2O removes all the control rules from the table and inserts the OLSR routing rules (see Figure 9).

An extension of mwSDN is proposed by Salsano *et al.* [90] to further improve the fault tolerance. The architecture consists of multiple controllers, which synchronously form a single logical controller. A MAP selects a single master controller based on a local master section entity. A local control agent called Embedded Flow Table Manager (EFTM) is embedded at each MAP. It locally configures the flow table and chooses an appropriate controller for a MAP. An OLSR daemon in a MAP uses wireless

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

TABLE 4
A comparison of the software defined mesh networks

ARCHITECTURE	FEATURES	CONTROL TRAFFIC ROUTING	LOCAL CONTROLLER	OPENFLOW COMPATIBILITY
Mesh Flow [80]	Load balancing and mobility management.	OLSR [81]	Yes	Yes
Gap [83]	Routing.	HWMP	No	Yes
wmSDN [89]	Reliability under controller failure.	OLSR	Yes	Yes
Multi-controller mesh [90]	Deploy multiple controllers.	OLSR	Yes	Yes
OpenFlow load [87]	Load balancing.	BATMAN [91]	Yes	Yes
Traffic orchestration [88]	Spectrum allocation and scheduling.	No protocol.	Yes	Yes
Flow Energy [92]	Energy optimization.	Not applicable.	No	No

interfaces to exchange control messages among the MAPs to build the flow tables. Once such tables are generated, flow rules are used to forward packets.

In the case of controller failure, EFTM removes the rules from the flow table and initiates regular OLSR routing. However, all active controllers need to be synchronized such that they have the same topological view. This global topological view is accomplished through OLSR. Also, each MAP needs to be associated with a single master controller. EFTM initiates new controller selection after a MAP detects any topology changes. The newly connected controller may purge the old rules from previous controller and insert new ones for itself. This selection strategy ensures single controller association and avoids the need for controllers synchronization. In the case of a single network without any partition a single controller is chosen by the entire network.

5.5 Energy Efficient Operation

Amokrane *et al.* [92] explore using SDN to improve the energy efficiency of flow based routing in WMN. The problem is formulated as an Integer Linear Program (ILP) problem with the objective of minimizing the network operation and reconfiguration costs with constraints on bandwidth and delay. The problem is NP-hard; therefore, they use an ant colony based meta heuristic to converge on a solution and deploy it using central control. The authors envision integrating the framework within OpenFlow, but do not actually carry out this integration.

5.6 Learned Lessons and Open Challenges

Table 4 summarizes the existing efforts employing SDN in WMN context. Overall, we see that most of the architectures (the first five from Table 4) target resource allocation: flexible routing, load balancing, mobility management, and fault tolerance. Structurally, the proposals have many similarities, and it is conceivable to unify them in an architecture that combines their individual advantages. Somewhat different objectives (optimal scheduling and energy-efficient operation) are considered in [88] and [92], respectively.

Another feature exists in most of the SDN-based mesh architectures is the task distribution through local controller. We have also seen similar trend in multihop sensor networks. In the following we outline further extensions of the above discussed architectures.

The limited existing work leaves a number of open opportunities to exploit SDNs. Virtualization is an effective way of accommodating multiple overlapped networks under a single physical network. Virtualization can also help supporting seamless mobility among multiple heterogeneous technologies. Data plane programmability for better protocol evolution is another missing component from the above architectures. While SDRs were considered in one instance [87] to control allocation of data and control traffic, SDRs can be leveraged as part of a virtualized architecture for a better programmable architecture.

The reliability and scalability issues are considered in a couple of the above designs, however both seem at their primary stage without having much evaluation evidence. Complete system design with a set of performance metrics as a benchmark architecture is also missing in software defined mesh networks. We also see scope for self adaptation architecture for better congestion and energy management. Finally, the use of SDNs for other dimensions of operation including application and service support, and security remain open areas of research in WMNs.

6 SDN-BASED HOME NETWORKS

The final network class we consider is Wireless Home Networks (WHNs). WHNs are an extremely popular class of wireless networks: In 2010, there

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

were an estimated 149 million home networks, a number that is expected to grow to a billion by 2030 [93]. In this section, we first review the design challenges faced by this type of network, and how SDN could help. We follow by discussing existing SDN efforts in this space. Finally we present the open challenges in SDN-based home network design.

6.1 Design Challenges in WHN

Homes are increasingly connected using wireless networks. A home network environment consists of devices such as computers, phones, tablets, TVs, and gaming consoles that communicate using commodity wireless technologies. Home networks differ from traditional enterprise wireless LANs in a number of ways. In terms of applications, multimedia rich entertainment applications that stream video and audio are common. Some of these applications have real-time constraints, including gaming, and video-teleconferencing applications [94]. These applications require high bandwidth, low latency or loss that may not happen due to multiple concurrent activities. The continued emergence and proliferation of home automation systems introduce additional traffic with stringent Quality of Service (QoS) and Quality of Experience (QoE) requirements.

Home networks also differ from enterprise WLANs in that their deployment is not coordinated. Each home has its own Internet connectivity through one or more wireless routers whose deployment is not planned or optimized. In urban settings, this leads to high interference between nearby homes. Thus, efficient *interference management and resource allocation* is critical in WHNs to support QoE for users. This QoE can also be improved by gathering the preferences from users using, for example, API deployed in home networks.

6.2 SDN-based Solutions

SDN-based design can bring significant benefits to home networks. In current solutions devices within a home compete in a distributed way for bandwidth, leading to destructive interference and failure to meet user expectations for service. At another level, interference between nearby homes is also managed in a distributed way that can lead to unfairness and poor performance. SDN allows more effective and fair use of the available bandwidth through a centralized control plane that can coordinate resource allocation across homes and across applications within a home.

Network slicing is a promising approach to allow splitting a physical home network into multiple independent controllable slices to provide isolation for different services (or homes) and to provide a better user experience. The management of slices may be assigned to a third party, who can remotely manage the interference, QoS, security, across multiple nearby homes. Alternatively, application providers like Netflix can be allowed to manage their own slice to offer a high quality video streaming [95].

A SDN-based architecture may also gather user preferences to customize and improve user experience. Like other networks, SDN can also be used to optimize resource allocation and content delivery by exploiting a global network view and central control of the data plane. A possible SDN-based home network architecture is shown in Figure 10.

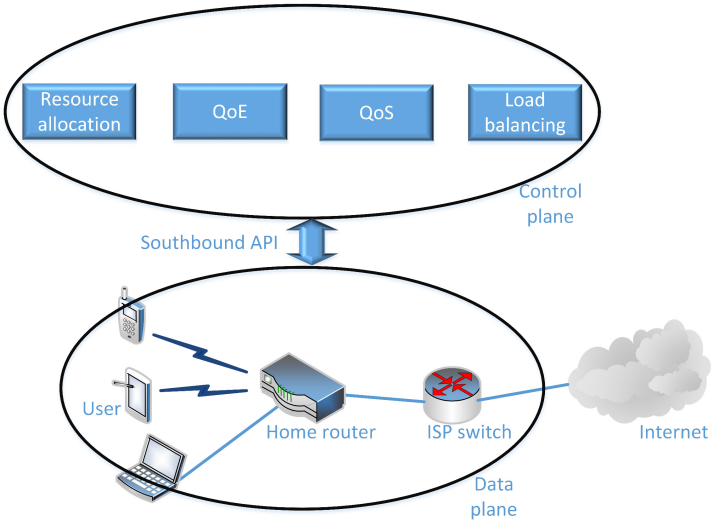

Fig. 10. A software defined wireless home network architecture.

6.3 Resource Management through Slicing

In this group of proposals, network slicing is proposed to improve resource management in WHNs. Yiakoumis *et al.* [95] propose a home network slicing architecture (*Home Slice*), where each slice is defined by its resource requirements (bandwidth, CPU, and forwarding table entries), traffic, and packet forwarding control logic. This slicing layer is implemented between the data plane (which consists of OpenFlow enabled gateways and routers) and the control plane (which consists of one or more

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

controllers for slices). The implemented prototype uses FlowVisor [20] as the slicing layer and two NOX controllers as network management and video controllers, respectively. An SNMP manager from the control plane configures the home Access Points (APs). The design suffers from the communication delay between the home routers and the controller, which resides in the provider's network. HomeVisor [96] is an extension of Home Slice by implementing a user interface to gather user preferences for a better QoS and network management.

In subsequent work, Wang *et al.* [97] propose a slicing framework that consists of a pair of OpenFlow enabled switches, one for high-speed and the other one for the low-speed devices. These switches are then connected to the controller through the home gateway. Slicing is done based on the device's requirement in terms of bandwidth, latency, or use frequency with the goal of offering fair and efficient resource sharing among similar class home devices. Three slicing strategies are proposed, namely: application-, location-, and bandwidth- centred slicing. The slicing in the first strategy is based on the user preference or experience. The location-centered slicing groups nearby devices. Finally, the bandwidth-centered strategy considers the bandwidth requirements of the devices. Simulation results using Mininet [98] show that the application based strategy is best in terms of delay, whereas the location based one is more stable in terms of throughput. The architecture also differs from Home Slice in terms of the controller placement as it allows the controller to be installed inside the home network.

6.4 Managing Dense Home Networks

In dense urban areas, it is common to have a large number of APs within transmission range of each other. These APs from neighboring homes suffer from interference and congestion that lead to poor service and unfairness. The next group of works focus on solutions to this problem.

The problem can be mitigated through coordinating neighboring APs, allowing users to configure their service while smartly managing the infrastructure. Yiakoumis *et al.* [99] propose an architecture that virtualizes the APs. Users configure and customize their own virtual APs that remain unchanged throughout the entire coverage. The virtual APs are mapped to physical APs. In dense deployments, some of the physical APs may be turned off to control interference and congestion. A central controller configures and manages the network to ensure QoE for the users. Some practical details are not specified completely, including addressing the issue of how APs from neighboring homes that may be supported by different providers could coordinate.

BeHop [100] is an extension of the above architecture for better resource allocation and user association. It consists of a central controller, a set of APs forming the data plane, and a network monitoring and evaluation data collector. Each BeHop AP is an OpenFlow switch that has a virtual AP (VAP) and a client table to track users as well as the network state information. Each AP also has an API for channel and power allocation. The collector's role is to collect state information from various network components and combine them to generate a higher level statistical model for management and evaluation purposes. The controller can access this model through an interface of the collector. Thus, the design could be viewed as a task distribution approach between the collector and the controller to remove the data gathering and monitoring burden from the controller to offer better responsiveness.

The architecture is implemented using a POX OpenFlow controller [10]. The data plane entities are commodity dual radio WiFi APs that run OpenWRT. Each AP runs Open vSwitch with SDN WiFi extension. A use case study shows that 5GHz band is a better choice in terms of minimizing interference and packet delivery time. However, it requires higher density compared to 2GHz band, which has longer propagation distance. Thus, increasing the AP density will improve the performance of the 5GHz band, whereas this will introduce further interference to 2GHz. This problem could be better handled through a centralized controller where the controller may even take care of selecting the best band for user operation (2GHz vs 5GHz).

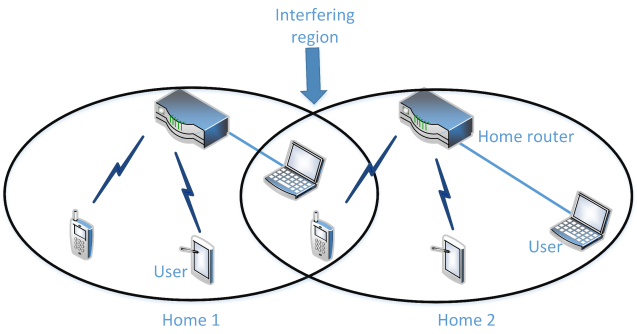

Fig. 11. Interference among home networks.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

None of the above architectures specifically mentioned how to manage the interference from APs served by different providers (Figure 11). COAP [101] presents a design to address this problem using central coordination to manage channel allocation and interference without relying on end users or service providers. The cloud-based controller (implemented on Floodlight [12]) gathers network statistics to have a global network view to dynamically manage the COAP-enabled APs which are served possibly by different providers. The communication between the controller and the APs is executed through a wireless extended OpenFlow. However, controlling such multi-provider APs requires purchasing the COAP controller service to manage the entire building network, as well as significant technical expertise to implement the local configuration. There is a need for an automated solution even in the case of multi-provider APs, so that the controller could be placed and controlled from a remote location by a third party service provider.

6.5 User Controlled Resource Management

User experience can be enhanced if their intent is known and accommodated by their ISP. Thus, Yiakoumis *et al.* propose a SDN-based home network architecture [102] having the users define *which* traffic gets *what* type of service, and *when* this happens. At the same time, the ISP figures out *how* and *where* in the network, provisioning is implemented. The first part of this principle is designing a user agent that takes the user choices and translates them to low-level network directives. Once the required directives are received by the ISP, it does validity check (requesting user is allowed to pose such request) and admits the request if network capacity is available. The ISP then uses network slicing to better configure the network. A prototype of the proposed solution is implemented to test the performance of Skype and video streaming demonstrating the feasibility and effectiveness of the proposed solution.

Ferguson *et al.* propose *PArticipatory NEtworking* (PANE): a similar principle that accumulates user choices [103]. In this solution, the user-facing API is layered on top of the controller to pass the acquired information to the controller. The design targets resolving conflicting user requests while allowing concurrent user controls for better fairness. Concurrent conflict-free user activities is also targeted by Kumar *et al.* [94]. The architecture consists of an OpenFlow enabled switch where the home gateway is connected, residing as part of the ISP network that also hosts the controller. The user interaction GUI is installed in one of the home network devices. This GUI allows a user to set parameters for existing devices and applications. The parameters are then passed to the controller that dynamically slices network link capacity by creating flow rules in the switch.

The existing Open vSwitch architecture used for home networks does not support per-flow QoS. FlowQoS [104] overcomes this limitation by proposing a SDN-based architecture, where a front end GUI at the user end captures their preferences and passes that information to the controller. The controller has two components: (1) traffic classifier, which classifies each flow to identify corresponding application such that the controller can install appropriate flow rules at the switches; and (2) rate controller which then enforces the preferred rate to that ongoing flow as per user choice. The architecture is implemented on OpenFlow using a POX [10] controller. Evaluations show that FlowQoS improves the performance of competing home applications like video streaming and VoIP. In a similar vein, P2PQoS [105] targets QoS for P2P video streaming service through a SDN enabled architecture. The proposed architecture consists of a controller for the content or the ISP provider. The video content can either be directed to the clients or can be passed as a P2P multicast stream. Simulation studies using Mininet [98] demonstrate that the SDN-based solution enhances QoS for P2P multicast streaming.

6.6 Cloud-based control

Gharakheili *et al.* propose fine-grained management of home-user experience using a cloud-supported SDN-based home network design [106]. The cloud based front-end API is exposed to users for their preferences while SDN-based back end resides in the network of the ISP for a fine-grained control and management of the last mile connection from the home gateway to the ISP's network. A cloud supported design is also proposed in [107], where the controller resides in the cloud to offer services like home-device auto recognition and connection management without users involvement like in [106]. The design also supports service-priority based bandwidth allocation. Thus it can be a complimentary solution for the architecture in [106] to avoid the need for users participation and also to offer priority based resource allocation. A similar

user interface based design is also presented in [108] that allows users to control their network as per their service requirements without using any cloud support.

6.7 Optimizing Content Delivery to Homes

Adaptive video streaming protocols such as Dynamic Adaptive Streaming over HTTP (DASH) [113] adjust the resolution of an ongoing streaming according to the current network conditions. However, the video and the Internet service providers as well as the clients have no network wide view or knowledge that may help to detect and avoid congestion on the assigned routing path from the delivery node of the Content Distribution Network (CDN) to the client. Clients usually connect to an assigned delivery node and do not switch to another one in the case of congestion or malfunction.

Within this context, Hyunwoo *et al.* [110] propose a QoE aware SDN-based video streaming protocol operating from the service providers perspective. The architecture consists of a controller that assigns the best delivery node to a requesting client according to the network conditions. The routing path from the delivery node to the client is determined using Constrained Shortest Path First (CSPF) algorithm over Multi-Protocol Label Switching (MPLS). The controller then continuously monitors the network (links' bandwidth, packet loss rate, and jitter) for congestion. It also monitors the delivery QoE metrics (video start-up delay, buffering rate, and play-out buffer) from the client. When performance problems are detected, the controller dynamically reassigns a new delivery node to the client. The proposed solution is implemented using Junos Space [114], a SDN-based network management solution for Juniper Networks, to show its effectiveness.

With adaptive streaming it is not easy to measure the QoE from the content or ISP providers side as the bandwidth utilization always looks high regardless of the video quality. This is because the bitrate of an ongoing stream is always adjusted to respond to the available bandwidth. Indeed, users always adjust their bitrate according to their experience instead of coordinating with other users, resulting in frequent rate switches and unfairness. Ramakrishnan and Zhu [111] as well as Hyunwoo *et al.* [110] propose taking user QoE input with respect to device, content, and subscription type. The controller collects this input along with the network status to dynamically configure and allocate the network resources from a centralized OpenFlow based controller to guarantee QoE fairness. Note that the first solution [111] packs multiple streams within a given bandwidth to ensure QoE fairness, whereas the second proposal [110] dynamically changes the delivery node to guarantee QoE.

Rukert *et al.* [109] propose supporting multiple concurrent user sessions through a single OpenFlow enabled home gateway by flexible traffic management between this gateway and the provider's server. In particular, the architecture concentrates on the last mile connection between the gateway and the provider, and replaces it with a set of software defined devices to distribute some of the control burden from the provider to other components of this last mile network. Thus, scalability in terms of multiple concurrent users sessions is accomplished.

6.8 Learned Lessons and Open Challenges

A comparison of various software enabled home network architectures is presented in Table 5. The different projects may be classified into four groups. The first one focuses on network virtualization or slicing for traffic and resource isolation to achieve QoS and flexible network manageability. The following group includes Dense Home [99], BeHop [100], and COAP [101] and targets managing dense urban home network settings. In the third category, the proposed home network architectures involve users in expressing their preferences to improve the overall QoE. Two of these architectures (FlowQoS [104] and P2PQoS [105]) support per-flow QoS while gathering user preferences. Some solutions use a cloud-based service to improve controller utilization for better QoE. The final class of architectures use SDN to optimize last-mile and/or provider network video streaming.

Only one of the above architectures ([109]) discusses the scalability in terms of supporting multiple concurrent sessions through a single home gateway. A scalable and robust software enabled home network is needed to be designed. Task distribution and hierarchical designs are also missing in the home network design, which can bring further flexibility and responsiveness in the performance. Multi home interference mitigation also needs further investigation to have an easy to use and flexible architecture that can operate even when networks are served by different providers.

With the explosion of WHNs and services and markets that rely on them, we expect commercial

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

TABLE 5
A comparison of the software defined home networks

ARCHITECTURE	FEATURES	VIRTUALIZATION	USER INVOLVEMENT	OPENFLOW COMPATIBILITY
Home Slice [95]	Resource and traffic isolation for QoS and manageability.	Yes	No	Yes
HomeVisor [96]	Slicing and user involvement for a better QoS.	Yes	Yes	Yes
Optimal Slice [97]	Identified the best slicing strategy.	Yes	No	Yes
Traffic management [109]	Traffic management for scalability.	No	No	Yes
Dense Home [99]	User managed vAPs for interference mitigation.	Yes	Yes	Yes
BeHop [100]	Flexible dense network management.	Yes	No	Yes
COAP [101]	Multi-provider AP management.	No	No	Yes
Home user [102]	QoE through user participation and slicing.	Yes	Yes	Yes
PANE [103]	Fairness among services.	No	Yes	Yes
User control [103]	Fairness among services.	No	Yes	Yes
FlowQoS [104]	QoS through user participation.	No	Yes	Yes
P2PQoS [105]	QoS for P2P streaming service.	No	Yes	Yes
Home experience [106]	QoE through cloud-based SDN.	No	Yes	Yes
Home auto-config [107]	Connection and resource management.	No	No	Yes
Understand home [108]	Designing software enabled home router.	No	Yes	Yes
QoE-aware video [110]	QoE-aware video streaming.	No	No	No
Adaptive video [111]	QoE-aware adaptive video streaming.	No	No	No
Fair QoE [112]	Supports QoE fairness.	No	No	Yes

interest to continue to drive innovation in this space. We believe that more collaborative services may continue to be pushed closer to the homes, opening up additional possibilities for SDN.

7 SDN IN OTHER WIRELESS NETWORKS

Thus far, we have presented the use of SDN in detail for four classes of wireless networks. In this section we summarizes wireless SDN based solutions for other wireless network classes.

7.1 Wireless Local Area Networks (WLAN) Architectures

The last hop connection between an AP and user in 802.11 standard is made based on a local user decision without infrastructure control. This may make the mobility and load balancing a tedious task. Several designs use SDN to virtualize this last hop connection to better support mobility and allow infrastructure management. Odin [115] introduces the idea of Lightweight Virtual Access Points (LVAPs) in enterprise WLAN environment to create a continuing connection between APs and users using an unique BS SSID associated with the user. Thus, each user is associated with a LVAP and multiple such LVAPs are hosted by an AP. Multiple agents run on each AP and centralized controller with a global network view manages the mobility (without triggering re-association), load balancing, and interference.

Several extensions of Odin provide additional improvements. Aeroflux [116] allows the local controller to manage events that do not require any global coordination and may change often. The global controller takes care of the events that need global coordination such as load balancing. Thor [117] also extends Odin to support energy efficient mobility management without compromising performance. OpenSDWN [118] is another extension of Odin for both home and enterprise network management. A similar concept of virtual access point for efficient authentication, authorization, mobility and interference management is proposed in CloudMAC [119]. Complex MAC-level functionality is extracted from an AP and brought to a centralized controller.

Zhao *et al.* propose using SDN to mitigate interference in IEEE 802.11 WLANs [120] using centralized control of APs. In particular, OpenFlow enabled APs are managed by a centralized controller in the sense that their downlink (80% of the total traffic) scheduling is realized by the controller. The packets are sent to 802.11 interface of the APs as per the installed flow-rules from where DCF is followed to reach the receiving user. However, the scheduling algorithm is not optimal; rather it uses a heuristic to minimize interference. *meSDN* is a complimentary solution where the controller monitors QoS traffic

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

through an application deployed in the user-devices for a fair resource distribution. In addition, meSDN proposes a TDMA like scheduling on top of 802.11 MAC, called pTDMA that minimizes the contention among slices as well as users within a slice. The buffered downlink traffic transmission from the APs is triggered by the user's uplink transmission for power saving. However, pTDMA also cannot guarantee complete interference-free transmission.

7.2 Mobile Ad hoc Network Architectures

Mendonca *et al.* [121] make the case for using SDN in heterogeneous networks combining infrastructure and ad hoc networks, identifying several benefits. Abolhasan *et al.* [122] propose a hierarchical architecture that is composed of centralized controller, regulars nodes, and a set of relay nodes to connect these nodes to the controller. The main idea is to bring the routing preprocessing (learning the topology with associated weights) to the controller to reduce the load from regular nodes for a scalable solution. Ku *et al.* propose another architecture for ad hoc networks [123] that exploits the centralized control for routing performance improvement.

Dong *et al.* [124] pursue improving network scalability in a mobile ad hoc wireless network through a two-tier Ternary Content Addressable Memory (TCAM) caching strategy for efficient rule management. A hash map is used in the case of a cache miss that triggers the controller to take care of this missing event. The framework is implemented using SDN.

7.3 SDN in Vehicular Networks

Vehicular networks are challenging because of their high mobility. This mobility also challenges the use of SDN because communication with a centralized controller can delay decisions and make estimates of network state stale. Nevertheless, a number of recent proposals have considered using SDN in vehicular network settings. Zheng *et al.* [125] propose augmenting CloudRan to improve the cellular network performance for vehicular networks. Conversely, Baron *et al.* consider using a vehicular network to offload backhaul traffic so that it is carried by vehicles as they move between cellular towers [126]. Cao *et al.* propose a type-based content delivery infrastructure for Vehicle to Vehicle (V2V) and Vehicle to Infrastructure(V2I) content delivery [127]. The proposed infrastructure is implemented using a SDN network. Ku *et al.* [128] also argue that SDN can bring management flexibility and programability in VANET to improve the overall performance.

7.4 SDN and Device-to-Device (D2D) Communications

EnergyFlow [129] is an OpenFlow based architecture that unifies the control of the data plane supporting technologies like WiFi, cellular, and D2D communications under a single centralized controller. The controller learns the network state for a dynamic configuration that targets balancing the performance and the energy use. For instance, the controller may turn off lightly used APs to save energy. Another architecture for D2D communication is proposed in [130], where the controller gathers the service and corresponding resource requirements for a dynamic configuration, like flow-scheduling.

7.5 Social Network Architectures

Content delivery in Mobile Social Networks (MSNs) is based on the social behaviors like social ties, community, and mobility. The data volume in such networks is expected to be large. To deal with this large volume of data Su *et al.* [131] design a software defined MSN where the controller uses a special *social switch* in addition to the social behaviors to generate a high speed secure channel between users and the social switch. However, content delivery will be executed without the social switch if there is no social tie between the end users for a requesting content delivery.

7.6 SDN in Smart Grid

The smart grid usually consists of electrical substations with a large number of various Intelligent Electronic Devices (IEDs). Managing large-scale heterogeneous devices introduces substantial complexity, cost and time. SDN can be used to dynamically configure and manage IEDs as explored by the Software-Defined Energy Communication Network (SDECN) architecture [132]. The architecture separates the control task from the data-plane/hardware and moves it to a controller. The controller configures the devices to carry out complex network traffic monitoring (e.g., circuit breaker closure). The controller also improves the access control policy by exploiting the global network information available to it. SDECN also suggests grid and IED virtualization for effective resource utilization.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

Dorsch *et al.* [133] pursue improving the reliability, robustness, time-critical performance, and communication security in transmission power and distribution grids using SDN. Furthermore, Dong *et al.* [134] envision SDN-based design bringing resilience in smart grids against failures and malicious attacks. Akkaya *et al.* [135] also explore bringing SDN to smart grid communications for a better management and security.

8 Discussion

We have shown through a review and classification of existing literature that SDN can play an important role in improving the performance, reliability and extensibility of four classes of common wireless networks. The four classes differ substantially in their organization, application demands and goals, and device architectures. Despite these differences, decoupling the control plane from the data plane offers substantial advantages for each scenario. We summarize our primary conclusions for each type of network below.

Cellular Networks: For software defined cellular networks we have classified all the architectures into two groups. The first group mainly focuses on the performance enhancement and scalability through efficient resource allocation. This is done by RAN, core, or both RAN and core programmability.

SoftRAN is based on the RAN programmability, which improves the mobility, resource management, and load balancing by removing some control task from the central controller to the data plane elements that has better view of the local network state (like downlink frequency allocation). We categorize this strategy as the "task distribution" between the control and data plane. SoftMobile also provides control task distribution through an abstract layered control plane design, where higher layers perform long term time-insensitive controls and lower layers manage frequent local changes.

SoftCell [46] considers core network components, whereas, its extension cellular SDN [41] focuses on both RAN and core. Both of them remove the control burden from the P-GWs for a scalable solution. In addition cellular SDN exploits the task distribution strategy to further relieve the control burden of the centralized controller. SoftAir, MyNET, and CMaaS proposed architectures for 5G. The next group of architectures are OpenRadio, OpenRoads, OpenRAN, SDWN, and CROWD. These solutions have a general goal of using SDN as a bridge layer between different technologies like WiFi, WiMAX to offer better QoS and QoE. SDN also supports continuous evolution, and even dynamic reconfiguration, of existing standards.

Sensor Networks: In comparison to WCNs, the other three network classes that we consider have only nascent efforts exploring the use of SDN. With respect to sensor networks we have identified three groups of SDN-based design philosophies in existing works. The first one, Sensor OpenFlow, is a SDN-based sensor network architecture to support multi-application environment on a single physical network. The design also targets smooth protocol evolution and multi-vendors compatibility. However, the architecture mainly focuses on customizing OpenFlow for sensor networks. The main objective of multi-application support is a future goal of the proposed design. An extension of Sensor OpenFlow is SDN-WISE [70], [71] that targets optimizing the energy usage through software enabled duty-cycle and data aggregation. In the same group of architectures [74] targets to support multi-tasking and energy optimization through the decoupled centralized control plane.

The next group of architectures include SDWN and its extension Smart [65], whose main goal is to optimize the energy usage through the task distribution between the control and data plane. Another design goal is to support duty-cycle and data aggregation. Both of these solutions provide a preliminary architecture that lacks modules like duty-cycle and data aggregation. The last group of architectures are the hierarchical designs for efficient data management and control-data plane interaction.

Mesh Networks: Mesh networks architectures are classified into three groups, where the first group considers traditional mesh routing protocol to manage the in-band control traffic while OpenFlow takes care of the data. The objectives of these architectures include flexible routing, load balancing and node mobility. Controller failure or network partitioning may be a relatively common occurrence: thus, we review a group of architectures with a goal of improving fault tolerance. Finally, we examine some solutions that use SDN to improve the energy efficiency of the network.

Home Networks: SDN-based home networks have five group of architectures. The first class includes solutions that use network slicing for resource management. Real deployment shows that slicing based home network architectures can be a cost effective and fair solution that will guarantee better QoE.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2016.2571118, IEEE Communications Surveys & Tutorials

The next group targets interference management in dense home network setup, especially in urban area where interference from nearby homes is common. Although proposed architecture achieves interference mitigation in dense urban setup using SDN, it is not clear how interference from different ISP providers could be mitigated. COAP [101] tries to mitigate multi home interference, which needs additional technical knowledge of the manager of an apartment to use their tools.

QoE aware architectures fall in the third group that uses SDN-based solution along with interactive users input to better control the network and support QoE. A group of proposals pursues QoE through a cloud-based design, while another focuses on efficient and QoE sensitive content delivery.

9 Crosscutting Themes and Open Research

We have outlined and discussed the open challenges in each class of networks we have considered. Importantly, we believe that there are themes and lessons that crosscut these different areas. In this section, we overview these common lessons and offer a perspective on the future of SDN use in wireless networks.

Consolidate Design: It is interesting to note that despite the large numbers of solutions, most pursue narrow objectives. We believe that there is room to integrate these solutions to provide an architecture that combines their strengths. For example, data plane programmability along with network virtualization controlled by a centralized controller may simultaneously support efficient and flexible resource allocation, fast and easy protocol evolution, energy optimization, within a robust and scalable solution that could also support multiple co-located applications.

Task Distribution: A common theme in many solutions is task distribution to manage complexity, improve scalability and to localize response. In particular, most of the sensor and mesh network solutions explore task distribution to manage in-band control traffic. Task distribution is also realized in the form of hierarchical control planes in software defined cellular network designs such as SoftMobile and CMaaS. Interestingly, CMaaS proposes decompositions along structural boundaries (different parts of the network), while SoftMobile proposes decomposition along functional boundaries. We believe that future work should more formally study this problem to move from ad hoc solutions and strategies to a deeper understanding of the tradeoffs involved.

Programmable Networks: A programmable wireless network can be comprised of SDN, NFV, and SDR. Each of these components serves a specific design goal and provide programmability to serve it. Combining them together opens the door to a broader wireless system design that can be adapted at multiple levels. The programmability of SDN enables not only protocol evolution, but also heterogeneous protocols and applications. It also supports the deployment of services and application-specific operations.

For instance, layered programmable components of SDN may include user, data, control, and application planes. SDR can then can care of dynamic radio access across different planes or slices. In addition, some network functions can be relegated to a cloud-based server for hardware-independent on-demand function instantiation. There is a need for a general model expressing such integration to enable systemic reasoning about and support of diverse wireless networks.

Use of Virtualization: We also identify virtualization as an integrated component of programmable networks. There is a a need for efficient virtualization strategies that may even incorporate SDR for dynamic radio access among the slices. Providing an interface to enable principled interaction among these slices is another design issue.

Benchmarks and experimentation methodologies: Many of the proposed designs have not been evaluated in sufficient detail to allow a deeper understanding of their properties. The root of the problem is the lack of availability of benchmarks and data. We believe that developing and releasing experimentation tools and simulation models for SDN can substantially improve our understanding of these solutions and accelerate the rate of innovation in this area.

Self-Optimizing Architectures: SDN-based designs provide flexibility of network configuration and management (e.g., efficient resource allocation, fairness, scalability, and QoE). Protocol evolution, multi-applications and multi-tasking are also supported by SDN-based designs. All these functionalities can be enhanced through an autonomic, self-adapting, self-optimizing and self-healing architectures, where different components of the network can be dynamically configured based on the net-

work dynamics. For instance, turning off some BSs in off-peak hours for energy savings in cellular network designs.

10 Conclusions

This article reviews the state of the art in terms of the application of Software Defined Networking (SDN) in Wireless Network settings. SDN is an exciting new technology that is revolutionizing network architecture an management in conventional enterprise networks and data centers. However, it also holds substantial promise in wireless networks where there are a number of challenges that they can help to address. The paper focuses on four classes of important wireless networks (Cellular Networks, Sensor Networks, Mesh Networks, and Home Networks) that differ substantially in their architecture, use models, traffic patterns and goals. For each of these classes of networks, we review the primary challenges they face, review existing works that have applied SDNs to address some of these challenges, and classify them based on their goals and architecture. The paper also briefly reviews applications of SDN in other classes of wireless networks. Furthermore, the paper attempts to examine the different solutions across the different classes of networks to come up with a taxonomy of SDN use in wireless environments. Finally, identify a number of research opportunities that we believe hold the most promise in terms of application of SDN to wireless networks.

References

[1] D. Clark, "The design philosophy of the darpa internet protocols," *ACM SIGCOMM Computer Communication Review*, vol. 18, no. 4, pp. 106–114, 1988.

[2] M. Casado, M. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker, "Ethane: Taking control of the enterprise," *SIGCOMM Computing Communication Review*, vol. 37, no. 4, pp. 1–12, August 2007.

[3] M. Casado, M. Freedman, J. Pettit, L. Jianying, N. Gude, . McKeown, and S. Shenker, "Rethinking enterprise network control," *IEEE/ACM Transactions on Networking*, vol. 17, no. 4, pp. 1270–1283, Aug. 2009.

[4] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Review*, vol. 38, no. 2, pp. 69–74, Apr. 2008.

[5] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Hölzle, S. Stuart, and A. Vahdat, "B4: Experience with a globally-deployed software defined WAN," *SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 3–14, October 2013.

[6] "North america software defined networking market," 2015, available online at http://www.micromarketmonitor.com/market/north-america-software-defined-network-sdn-9065162554.html.

[7] M. Kobayashi, S. Seetharaman, G. Parulkar, G. Appenzeller, J. Little, J. V. Reijendam, P. Weissmann, and N. Mckeown, "Maturing of openflow and software-defined networking through deployments," *Computer Networks*, vol. 61, pp. 151–175, March 2014.

[8] A. Bianco, R. Birke, L. Giraudo, and M. Palacin, "Openflow switching: Data plane performance," in *IEEE International Conference on Communications (ICC)*, June 2010, pp. 1–5.

[9] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. Mckeown, and S. Shenker, "NOX: towards an operating system for networks," *SIGCOMM Computer Communication Review*, vol. 38, no. 3, pp. 105–110, Jul. 2008.

[10] "Pox sdn network controller," 2015, available from http://www.noxrepo.org/pox/about-pox/.

[11] D. Erickson, "The beacon openflow controller," in *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN)*, August 2013, pp. 13–18.

[12] "Floodlight." [Online]. Available: http://www.projectfloodlight.org/floodlight/

[13] "OpenDaylight." [Online]. Available: https://www.opendaylight.org

[14] L. Kleinrock and F. Tobagi, "Packet switching in radio channels: Part i–carrier sense multiple-access modes and their throughput-delay characteristics," *IEEE Transactions on Communications*, vol. 23, no. 12, pp. 1400–1416, 1975.

[15] F. Tobagi and L. Kleinrock, "Packet switching in radio channels: Part ii–the hidden terminal problem in carrier sense multiple-access and the busy-tone solution," *IEEE Transactions on Communcations*, vol. 23, no. 12, pp. 1417–1433, 1975.

[16] W. H. Tuttlebee, *Software defined radio: enabling technologies*. John Wiley & Sons, 2003.

[17] K. Yap, M. Kobayashi, D. Underhill, S. Seetharaman, P. Kazemian, and N. McKeown, "The stanford openroads deployment," in *Proceedings of the 4th ACM International Workshop on Experimental Evaluation and Characterization (WINTECH)*, September 2009, pp. 59–66.

[18] C. Chaudet and Y. Haddad, "Wireless software defined networks: Challenges and opportunities," in *Proceedings of the International Conference on Microwaves, Communications, Antennas and Electronics Systems (COMCAS)*, October 2013, pp. 1–5.

[19] N. Chowdhury and R. Boutaba, "A survey of network virtualization," *Comput. Netw.*, vol. 54, no. 5, pp. 862–876, April 2010.

[20] R. Sherwood, G. Gibb, K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "Can the production network be the testbed?" in *Proceedings of the USENIX Conference on Operating Systems Design and Implementation*, 2010, pp. 1–6.

[21] A. Al-Shabibi, M. D. Leenheer, M. Gerola, A. Koshibe, G. Parulkar, E. Salvadori, and B. Snow, "OpenVirteX: Make your virtual SDNs programmable," in *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking (HotSDN)*, August 2014, pp. 25–30.

[22] X. Jin, J. Gossels, J. Rexford, and D. Walker, "CoVisor: A compositional hypervisor for software-defined networks," in *12th USENIX Symposium on Networked Systems*

Design and Implementation (NSDI 15), Oakland, CA, May 2015, pp. 87–101.

[23] "Ryu OpenFlow controller." [Online]. Available: https://osrg.github.io/ryu/

[24] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. D. Turck, and R. Boutaba, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Communications Magazine*, vol. 53, no. 2, pp. 90–97, February 2015.

[25] ——, "Network function virtualization: State-of-the-art and research challenges," *IEEE Communications Surveys and Tutorials*, September 2015.

[26] V. Yaz?c?, U. Kozat, and M. Oguz-Sunay, "A new control plane for 5g network architecture with a case study on unified handoff, mobility, and routing management," *IEEE Communications Magazine*, vol. 52, no. 11, pp. 76–85, November 2014.

[27] D. Kreutz, F. Ramos, P. Verissimo, C. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software defined networking : A comprehensive survey," *CoRR*, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[28] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Communications Surveys and Tutorials*, vol. 17, no. 1, pp. 27–51, March 2015.

[29] H. Farhady, H. Lee, and A. Nakao, "Software-defined networking: A survey," *Computer Networks*, vol. 81, pp. 79–95, April 2015.

[30] Y. Jarraya, T. Madi, and M. Debbabi, "A survey and a layered taxonomy of software-defined networking," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 4, pp. 1955–1980, November 2014.

[31] F. Hu, Q. Hao, and K. Bao, "A survey on software-defined network and OpenFlow: From concept to implementation," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 4, pp. 2181–2206, November 2014.

[32] I. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, "A roadmap for traffic engineering in SDN-OpenFlow networks," *Computer Networks*, vol. 71, pp. 1–30, October 2014.

[33] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks (HotSDN)*, August 2012, pp. 7–12.

[34] A. Hakiria, A. Gokhale, P. Berthou, D. C. Schmidt, and T. Gayraud, "Software-defined networking: Challenges and research opportunities for future internet," *Computer Networks*, vol. 75, pp. 453–471, December 2014.

[35] A. N. Jagadeesan and B. Krishnamachari, "Software-defined networking paradigms in wireless networks: A survey," *ACM Computing Survey*, vol. 47, no. 2, pp. 27:1–27:11, November 2014.

[36] D. Macedo, D. Guedes, L. Vieira, M. Vieira, and M. Nogueira, "Programmable networks–from software-defined radio to software-defined networking," *IEEE Communications Surveys and Tutorials*, vol. 17, no. 2, pp. 1102–1125, May 2015.

[37] M. Reza, S. Sivakumar, A. Nafarieh, and B. Robertson, "A comparison of software defined network (sdn) implementation strategies," in *2nd International Workshop on Survivable and Robust Optical Networks*, June 2014, pp. 1050–1055.

[38] B. A. A. Nunes, M. Mendonca, X. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 3, pp. 1617–1634, August 2014.

[39] A. Doria, L. Dong, W. Wang, H. M. Khosravi, J. H.Salim, and R. Gopal, "Forwarding and control element separation (ForCES) protocol specification," March 2010.

[40] B. Cao, F. He, Y. Li, C. Wang, and W. Lang, "Software defined virtual wireless network: framework and challenges," *IEEE Networks*, vol. 29, no. 4, pp. 6–12, August 2015.

[41] L. Li, Z. Mao, and J. Rexford, "Toward software-defined cellular networks," in *Proceedings of the 2012 European Workshop on SDN*, Washington, DC, USA, 2012, pp. 7–12.

[42] "Cisco visual networking index: Global mobile data traffic forecast update 2014–2019," February 2015.

[43] J. I. Agbinya, M. C. Aguayo-Torres, and R. Klempous, *4G Wireless Communication Networks: Design Planning and Applications*. River publisher, August 2013.

[44] A. Damnjanovic, J. Montojo, W. Y. Wei, J. Tingfang, L. Tao, M. Vajapeyam, Y. Taesang, S. Osok, and D. Malladi, "A survey on 3gpp heterogeneous networks," *IEEE Wireless Communications*, vol. 18, no. 3, pp. 10–21, June 2011.

[45] A. Gudipati, D. Perry, L. Li, and S. Katti, "SoftRAN: Software defined radio access network," in *Proceedings of the SIGCOMM Workshop on Hot Topics in SDN*, 2013, pp. 25–30.

[46] X. Jin, L. Li, L. Vanbever, and J. Rexford, "SoftCell: Scalable and flexible cellular core network architecture," in *Proceedings of the ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT '13)*, 2013, pp. 163–174.

[47] I. F. Akyildiza, P. Wangb, and S.-C. Lin, "SoftAir: A software defined networking architecture for 5G wireless systems," *Computer Networks*, vol. 85, pp. 1–18, July 2015.

[48] T. Chen, H. Zhang, X. Chen, and O. Tirkkonen, "Softmobile: control evolution for future heterogeneous mobile networks," *IEEE Wireless Communications*, vol. 21, no. 6, pp. 70–78, December 2014.

[49] "Open vswitch," 2013. [Online]. Available: http://openvswitch.org

[50] J. Liu, S. Zhang, N. Kato, H. Ujikawa, and K. Suzuki, "Device-to-device communications for enhancing quality of experience in software defined multi-tier LTE-A networks," *IEEE Networks*, vol. 29, no. 4, pp. 46–52, August 2015.

[51] H. Zhang, S. Vrzic, G. Senarath, N.-D. Do, H. Farmanbar, J. Rao, C. Peng, and H. Zhuang, "5G wireless network: MyNET and SONAC," *IEEE Networks*, vol. 29, no. 4, pp. 14–23, August 2015.

[52] M. Bansal, J. Mehlman, S.Katti, and P. Levis, "OpenRadio: A programmable wireless dataplane," in *Proceedings of the Workshop on Hot Topics in SDN*, 2012, pp. 109–114.

[53] K. Yap, R. Sherwood, M. Kobayashi, T. Huang, M. Chan, N. Handigol, N. McKeown, and G. Parulkar, "Blueprint for introducing innovation into wireless mobile networks," in *Proceedings of the SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures*, 2010, pp. 25–32.

[54] M. Yang, Y. Li, D. Jin, L. Su, S. Ma, and L. Zeng, "OpenRAN: A software-defined RAN architecture via virtualization," *SIGCOMM Comput. Commun. Review*, vol. 43, no. 4, pp. 549–550, Aug. 2013.

[55] C. Bernardos, A. D. L. Oliva, P. Serrano, A. B. L. Contreras, J. Hao, and J. Zniga, "An architecture for software defined wireless networking," *IEEE Wireless Communications*, vol. 21, no. 3, pp. 52–61, Jun. 2014.

[56] H. Ali-Ahmad, C. Cicconetti, A. D. L. Oliva, V. Mancuso, M. R. Sama, P. Seite, and S. Shanmugalingam, "An SDN-based network architecture for extremely dense wireless networks," in *IEEE SDN for Future Networks and Services (SDN4FNS)*, November 2013, pp. 1–7.

[57] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, March 2002.

[58] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman, "A taxonomy of wireless micro-sensor network models," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 6, no. 2, pp. 28–36, 2002.

[59] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda *et al.*, "A line in the sand: a wireless sensor network for target detection, classification, and tracking," *Computer Networks*, vol. 46, no. 5, pp. 605–634, 2004.

[60] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, 2002, pp. 88–97.

[61] E. Lee *et al.*, "Cyber physical systems: Design challenges," in *IEEE International Symposium on Object Oriented Real-Time Distributed Computing (ISORC)*, 2008, pp. 363–369.

[62] P. Levis, S. Madden, J. Polastre, R. Szewczyk, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler, "Tinyos: An operating system for sensor networks," in *in Ambient Intelligence*, 2004.

[63] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of network density on data aggregation in wireless sensor networks," in *IEEE International Conference on Distributed Computing Systems*, July 2002, pp. 457–458.

[64] S. Costanzo, L. Galluccio, G. Morabito, and S. Palazzo, "Software defined wireless networks: Unbridling SDNs," in *Proceedings of the European Workshop on Software Defined Networking*, October 2012, pp. 1–6.

[65] A. D. Gante, M. Aslan, and A. Matrawy, "Smart wireless sensor network management based on software-defined networking," in *27th Biennial Symposium on Communications*, June 2014, pp. 71–75.

[66] T. Luo, H. Tan, and T. Quek, "Sensor OpenFlow: Enabling software-defined wireless sensor networks," *IEEE Communications Letters*, vol. 16, no. 11, pp. 1896 –1899, Nov. 2012.

[67] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient mac protocol for wireless sensor networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, June 2002, pp. 1567–1576.

[68] "The contiki os," 2013. [Online]. Available: http://www.contiki-os.org

[69] "Blip : Berkeley IP," 2011. [Online]. Available: http://smote.cs.berkeley.edu:8000/tracenv/wiki/blip

[70] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "SDN-WISE: Design, prototyping and experimentation of a stateful sdn solution for wireless sensor networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, April 2015, pp. 513–521.

[71] L. Galluccio *et al.*, "Reprogramming wireless sensor networks by using SDN-WISE: a hands-on demo," in *IEEE International Conference on Computer Communications (INFOCOM)*, April 2015, pp. 19–20.

[72] D. Pfammatter, D. Giustiniano, and V. Lenders, "A software-defined sensor architecture for large-scale wide-band spectrum monitoring," in *Proceedings of the 14th International Conference on Information Processing in Sensor Networks*, ser. IPSN '15, 2015, pp. 71–82.

[73] D. Zeng, P. Li, S. Guo, and T. Miyazaki, "Minimum-energy reprogramming with guaranteed quality-of-sensing in software-defined sensor networks," in *IEEE International Conference on Communications*, June 2014, pp. 288–293.

[74] D. Zeng, P. Li, S. Guo, T. Miyazaki, J. Hu, and Y. Xiang, "Energy minimization in multi-task software-defined sensor networks," *IEEE Transactions on Computers*, vol. 64, no. 11, pp. 3128–3139, July 2015.

[75] B. Oliveira and C. Margi, "Spotled: A hierarchical control plane architecture for software defined wireless sensor networks," 2014.

[76] R. Rahmani, H. Rahman, and T. Kanter, "On performance of logical-clustering of flow-sensors," *International Journal of Computer Science Issues*, vol. 10, no. 5, p. 1, Sep. 2013.

[77] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Computer networks*, vol. 47, no. 4, pp. 445–487, 2005.

[78] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *Proc. ACM International Conference on Mobile Computing and Networking (Mobicom)*, 2004, pp. 114–128.

[79] A. Raniwala, K. Gopalan, and T.-c. Chiueh, "Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 8, no. 2, pp. 50–65, 2004.

[80] P. Dely, A. Kassler, and N. Bayer, "Openflow for wireless mesh networks," in *20th International Conference on Computer Communications and Networks (ICCCN)*, Aug. 2011, pp. 1–6.

[81] "The OLSRD : an adhoc wireless mesh routing daemon," 2007. [Online]. Available: http://www.olsr.org/

[82] "The KAUMesh testbed," 20011. [Online]. Available: http://www.kau.se/kaumesh

[83] V. Nascimento, M. Moraes, R. Gomes, B. Pinheiro, A. Abelem, V. Borges, K. Cardoso, and E. Cerqueira, "Filling the gap between software defined networking and wireless mesh networks," in *10th International Conference on Network and Service Management (CNSM)*, Nov. 2014, pp. 451–454.

[84] "The openwrt," 2004. [Online]. Available: https://openwrt.org/

[85] M. Bahr, "An openwrt solution for future wireless homes," in *IEEE International Conference on Multimedia and Expo (ICME)*, July 2010, pp. 1701–1706.

[86] C. E. Palazzi, M. Brunati, and M. Roccetti, "Update on the hybrid wireless mesh protocol of ieee 802.11s," in *IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS)*, October 2007, pp. 1–6.

[87] F. Yang, V. Gondi, J. O. Hallstrom, K. Wang, and G. Eidson, "OpenFlow-based load balancing for wireless mesh infrastructure," in *IEEE 11th Consumer Communications and Networking Conference (CCNC)*, January 2014, pp. 444–449.

[88] H. Huang, P. Li, S. Guo, and W. Zhuang, "Software-defined wireless mesh networks: architecture and traffic orchestration," *IEEE Networks*, vol. 29, no. 4, pp. 24–30, August 2015.

[89] A. Detti, C. Pisa, S. Salsano, and N. Blefari-Melazzi, "Wireless mesh software defined networks (wmsdn)," in *IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, October 2013, pp. 89–95.

[90] S. Salsano, G. Siracusano, A. Detti, C. Pisa, P. Ventre, and N. Blefari-Melazzi, "Controller selection in a wireless mesh SDN under network partitioning and merging scenarios," *CoRR*, vol. abs/1406.2470, 2014.

[91] D. Johnson, N. Ntlatlapa, and C. Aichele, "A simple pragmatic approach to mesh routing using batman," in *2nd IFIP International Symposium on Wireless Communications and Information Technology in Developing Countries*, October 2008.

[92] A. Amokrane, R. Langar, R. Boutabayz, and G. Pujolle, "Online flow-based energy efficient management in wireless mesh networks," in *IEEE Global Communications Conference (GLOBECOM)*, December 2013, pp. 329–335.

[93] P. Filipovic, "The future of home networks - a global perspective," 2008.

[94] H. Kumar, H. Gharakheili, and V. Sivaraman, "User control of quality of experience in home networks using SDN," in *IEEE International Conference on Advanced Networks and Telecommuncations Systems (ANTS)*, December 2013.

[95] Y. Yiakoumis, K. Yap, S. Katti, G. Parulkar, and N. McKeown, "Slicing home networks," in *Proceedings of ACM SIGCOMM Workshop on Home Networks*, 2011, pp. 1–6.

[96] T. Fratczak, M. Broadbent, P. Georgopoulos, and N. Race, "Homevisor: Adapting home network environments," in *2013 Second European Workshop on Software Defined Networks (EWSDN)*, 2013, pp. 32–37.

[97] S. Wang, X. Wu, H. Chen, Y. Wang, and D. Li, "An optimal slicing strategy for sdn based smart home network," in *2014 International Conference on Smart Computing (SMARTCOMP)*, 2014, pp. 118 –122.

[98] "Mininet." [Online]. Available: http://www.mininet.org

[99] Y. Yiakoumis, M. Bansal, S. Katti, and N. McKeown, "SDN for dense WiFi networks," in *USENIX open networking summit*, 2014, pp. 1–2.

[100] Y. Yiakoumis, M. Bansal, G. Covington, J. Reijendam, S. Katti, and N. McKeown, "BeHop: A testbed for dense WiFi networks," *Mobile Computing and Communications Review*, vol. 18, no. 3, pp. 71–80, 2014.

[101] A. Patro and S. Banerjee, "COAP: A software-defined approach for home WLAN management through an open API," in *Proceedings of the 9th ACM Workshop on Mobility in the Evolving Internet Architecture (MobiArch)*, September 2014, pp. 31–36.

[102] Y. Yiakoumis, S. Katti, T. Huang, N. McKeown, K. Yap, and R. Johari, "Putting home users in charge of their network," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, 2012, pp. 1114–1119.

[103] A. Ferguson, A. Guha, C. Liang, R. Fonseca, and S. Krishnamurthi, "Participatory networking: An API for application control of SDNs," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, 2013, pp. 327–338.

[104] M. Seddiki, M. Shahbaz, S. Donovan, S. Grover, M. Park, N. Feamster, and Y. Song, "FlowQoS: QoS for the rest of us," in *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking (HotSDN)*, 2014, pp. 207–208.

[105] I. Trajkovska, P. Aeschlimann, C. Marti, T. Bohnert, and J. Salvachua, "SDN enabled QoSprovision for online streaming services in residential ISP networks," in *IEEE International Conference on Consumer Electronics*, 2014, pp. 33–34.

[106] H. Gharakheili, J. Bass, L. Exton, and V. Sivaraman, "Personalizing the home network experience using cloud-based sdn," in *IEEE 15th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, June 2014, pp. 1–6.

[107] M. Lee, Y. Kim, and Y. Lee, "A home cloud-based home network auto-configuration using sdn," in *IEEE International Conference on Networking, Sensing and Control (ICNSC)*, April 2015, pp. 444–449.

[108] R. Mortier, T. Rodden, T. Lodge, D. McAuley, C. Rotsos, A. Moore, A. Koliousis, and J. Sventek, "Control and understanding: Owning your home network," in *IEEE International Conference on Communication Systems and Networks (COMSNETS)*, January 2012.

[109] J. Rkert, R. Bifulco, M. Rizwan-Ul-Haq, H.-J. Kolbe, and D. Hausheer, "Flexible traffic management in broadband access networks using software defined networking," in *IEEE Network Operations and Management Symposium (NOMS)*, May 2014, pp. 1–8.

[110] N. Hyunwoo, K. Kyung-Hwa, J. Y. Kim, and H. Schulzrinne, "Towards qoe-aware video streaming using sdn," in *2014 IEEE Global Communications Conference (GLOBECOM)*, 2014, pp. 1317–1322.

[111] S. Ramakrishnan and X. Zhu, "An sdn based approach to measuring and optimizing abr video quality of experience," 2014.

[112] P. Georgopoulos, Y. Elkhatib, M. Broadbent, M. Mu, and N. Race, "Towards network-wide qoe fairness using openflow-assisted adaptive video streaming," in *Proceedings of the 2013 ACM SIGCOMM Workshop on Future Human-centric Multimedia Networking*, 2013, pp. 15–20.

[113] T. Stockhammer, "Dynamic adaptive streaming over HTTP –: Standards and design principles," in *Proceedings of the Second Annual ACM Conference on Multimedia Systems (MMSys '11)*, 2011, pp. 133–144.

[114] "Junos space." [Online]. Available: http://www.juniper.net/us/en/products-services/network-management/

[115] L. Suresh, J. Schulz-Zander, R. Merz, A. Feldmann, and T. Vazao, "Towards programmable enterprise WLANS with odin," in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, ser. HotSDN '12, August 2012, pp. 115–120.

[116] J. Schulz-Zander, N. Sarrar, and S. Schmid, "Aeroflux: A near-sighted controller architecture for software-defined wireless networks," in *Open Networking Summit 2014 (ONS 2014)*, 2014.

[117] R. Riggio, C. Sengul, L. Suresh, J. Schulz-Zander, and A. Feldmann, "Thor: Energy programmable WiFi networks," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, April 2013, pp. 21–22.

[118] J. Schulz-Zander, C. Mayer, B. Ciobotaru, S. Schmid, and A. Feldmann, "Opensdwn: Programmatic control over home and enterprise wifi," in *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*, June 2015, pp. 1–12.

[119] J. Vestin, P. Dely, A. Kassler, N. Bayer, H. Einsiedler, and C. Peylo, "CloudMAC: Towards software defined wlans," *SIGMOBILE fMob. Comput. Commun. Rev.*, vol. 16, no. 4, pp. 42–45, February 2013.

[120] D. Zhao, M. Zhu, and M. Xu, "Leveraging SDN and OpenFlow to mitigate interference in enterprise WLAN," *Journal of Networks*, vol. 9, no. 6, pp. 1526–1533, June 2014.

[121] M. Mendonca, K. Obraczka, and T. Turletti, "The case for software-defined networking in heterogeneous networked environments," in *Proceedings of the 2012 ACM conference on CoNEXT student workshop*, 2012, pp. 59–60.

[122] M. Abolhasan, J. Lipman, N. Wei, and B. Hagelstein, "Software-defined wireless networking: centralized, dis-

tributed, or hybrid?" *IEEE Networks*, vol. 29, no. 4, pp. 32–38, August 2015.

[123] I. Ku, Y. Lu, and M. Gerla, "Software-defined mobile cloud: Architecture, services and use cases," in *International Wireless Communications and Mobile Computing Conference (IWCMC)*, August 2014.

[124] M. Dong, H. Li, K. Ota, and J. Xiao, "Rule caching in SDN-enabled mobile access networks," *IEEE Networks*, vol. 29, no. 4, pp. 40–45, August 2015.

[125] K. Zheng, L. Hou, H. Meng, Q. Zheng, N. Lu, and L. Lei, "Soft-defined heterogeneous vehicular network: Architecture and challenges," *arXiv preprint arXiv:1510.06579*, 2015.

[126] B. Baron, P. Spathis, H. Rivano, M. D. De Amorim, Y. Viniotis, and J. Clarke, "Software-defined vehicular backhaul," in *Wireless Days (WD), 2014 IFIP*. IEEE, 2014, pp. 1–6.

[127] Y. Cao, J. Guo, and Y. Wu, "Sdn enabled content distribution in vehicular networks," in *IEEE International Conference on Innovative Computing Technology (INTECH)*, 2014, pp. 164–169.

[128] I. Ku, Y. Lu, M. Gerla, F. Ongaro, R. L. Gomes, and E. Cerqueira, "Towards software-defined vanet: Architecture and services," in *Proceedings of the IEEE Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET)*, 2014, pp. 103–110.

[129] C. Donato, P. Serrano, A. Oliva, A. Banchs, and C. Bernardos, "An OpenFlow architecture for energy-aware traffic engineering in mobile networks," *IEEE Networks*, vol. 29, no. 4, pp. 54–60, August 2015.

[130] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, and N. Venkatasubramanian, "A software defined networking architecture for the Internet-of-Things," in *IEEE Network Operations and Management Symposium (NOMS)*, May 2014, pp. 1–9.

[131] Z. Su, Q. Xu, H. Zhu, and Y. Wang, "A novel design for content delivery over software defined mobile social networks," *IEEE Networks*, vol. 29, no. 4, pp. 62–67, August 2015.

[132] A. Cahn, J. Hoyos, M. Hulse, and E. Keller, "Software-defined energy communication networks: From substation automation to future smart grids," in *Proceedings of the IEEE International Conference on Smart Grid Communications (SmartGridComm)*, October 2013, pp. 558–563.

[133] N. Dorsch, F. Kurtz, H. Georg, C. Hgerling, and C. Wietfeld, "Software-defined networking for smart grid communications: Applications, challenges and advantages," in *Proceedings of the IEEE International Conference on Smart Grid Communications (SmartGridComm)*, November 2014, pp. 422–427.

[134] X. Dong, H. Lin, R. Tan, R. K. Iyer, and Z. Kalbarczyk, "Software-defined networking for smart grid resilience: Opportunities and challenges," in *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security (CPSS)*, 2015, pp. 61–68.

[135] K. Akkaya, A. S. Uluagac, and A. Aydeger, "Software defined networking for wireless local networks in smart grid," in *Proceedings of the IEEE International Local Computer Networks Conference Workshops*, October 2015, pp. 826–831.

Israat Tanzeena Haque received her PhD degree in Computing Science from the University of Alberta. Currently she is working in the department of Computer Sc. and Eng. at the University of California, Riverside as a postdoctoral fellow. Her research interest includes network design and optimization, wireless sensor networks, Internet of Things, software defined networking, and cyber physical system. Dr. Haque serves as a TPC member for IEEE ICC, Globecom, WCNC, LCN, and many more. She is an ACM member and IEEE senior member.

Nael Abu-Ghazaleh is a Professor in the Computer Science and Engineering department and the Electrical and Computer Engineering department at the University of California at Riverside. His research interests are in the areas of secure system design, parallel discrete event simulation, networking and mobile computing. He received his PhD from the University of Cincinnati in 1997.