Software-Defined Networking Paradigms in Wireless Networks: A Survey

NACHIKETHAS A. JAGADEESAN and BHASKAR KRISHNAMACHARI,
University of Southern California

Software-defined networking (SDN) has generated tremendous interest from both academia and industry. SDN aims at simplifying network management while enabling researchers to experiment with network protocols on deployed networks. This article is a distillation of the state of the art of SDN in the context of wireless networks. We present an overview of the major design trends and highlight key differences between them.

Categories and Subject Descriptors: C.2.1 [**Network Architecture and Design**]: Wireless Communication; C.2.2 [**Computer-Communication Networks**]: Network Protocols; C.2.3 [**Network Operations**]: Network Management

General Terms: Design, Management

Additional Key Words and Phrases: Wireless, SDN, OpenFlow

ACM Reference Format:
Nachikethas A. Jagadeesan and Bhaskar Krishnamachari. 2014. Software-defined networking paradigms in wireless networks: A survey. ACM Comput. Surv. 47, 2, Article 27 (November 2014), 11 pages.
DOI: http://dx.doi.org/10.1145/2655690

1. INTRODUCTION

In this article, we review the published research literature on the application of software-defined networking (SDN) ideas in wireless networks. We begin with a brief overview of the history of SDN and what the term entails.

Software-defined networking is a broad term that is applied to a variety of approaches to network design. Traditionally, packet-switched networks have consisted of nodes running distributed protocols to route packets. There is little, if any, control over the forwarding tables of a switch. The *control* of the path of a packet is given to individual routers making decisions according to some distributed algorithm. SDN is a network view that argues for a separation of routing intelligence from the device itself. In software, this would enable the manipulation of the forwarding tables of these "dumb" switches by a possibly centralized controller. This separation of the control and data planes allows one to experiment with network protocols in an easier manner than what is possible today. The vision is to have a common set of hardware, such as switches, that form a base over which various network functionalities are implemented in software. A rough analogy is how the x86 instruction set provides a common architecture for

This work was supported by the NSF grant AST-1248017.
Authors' address: Nachikethas A. Jagadeesan and Bhaskar Krishnamachari, Department of Electrical Engineering, University of Southern California, Los Angeles, CA-90089; email: {nanantha,bkrishna}@usc.edu.

DOI: http://dx.doi.org/10.1145/2655690

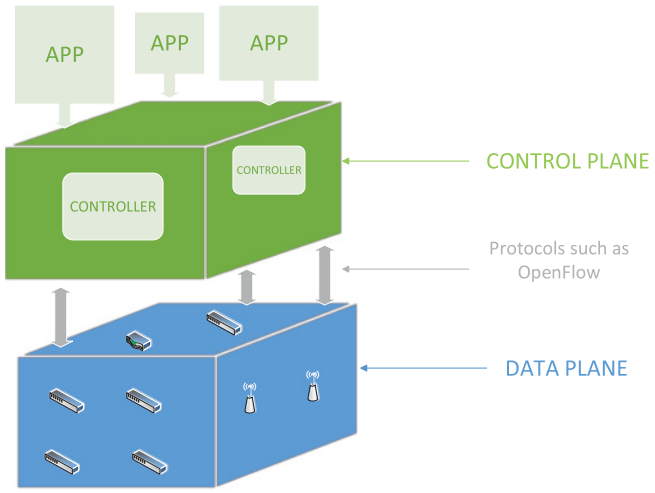

Fig. 1. Basic SDN architecture.

personal computers over which many software implementations, providing different functionalities, are run. Figure 1 provides an illustration of the basic SDN architecture.

The earliest efforts in SDN, such as Ethane [Casado et al. 2007], RCP [Caesar et al. 2005], and 4D [Greenberg et al. 2005], arose as a protest against the ossificationof networks. Researchers were being increasingly aware of the need to simplify the functions baked into network infrastructure and wished for the ability to extend network devices with desired functionality as and when needed. Ethane introduced a flow-based policy language to a network comprised of a simplified data plane and a centralized control plane. It demonstrated the feasibility of operating a centrally managed network. OpenFlow [McKeown et al. 2008] built on this work by defining an open protocol that defines communication between the network controller and a network device such as a switch.

Some of the ideas that characterize present-day SDN predate the coining of the term [TR10 2009]. SDN originally referred to the OpenFlow project at Stanford. Since then, the term has evolved to encompass any network architecture possessing the following two properties [Feamster et al. 2013]. First, the decisions on how to handle packet data are separated from the operations that carry out those decisions. This property is commonly known as *control and data plane separation*. The control and data planes interface with each other using a well-defined API such as OpenFlow. Second, the control plane allows the operation of a possibly disparate set of devices from a single vantage point. For the realization of this property, it is crucial that the devices in the network expose their capabilities through a common API.

Although the idea of moving to a logically centralized network and separating control and data traffic is central to SDN, these concepts were present earlier in the world of telephony. The Signaling System No. 7 (SS7) protocol used in public-switched telephone networks (PSTNs) is characterized by its separation of voice and the signaling required to provide call services. However, PSTNs are circuit-switched networks. Telephones are connected, through subscriber lines, to signaling switching points (SSPs), which are connected to each other using voice trunks. The signaling required for exchanging messages between SSPs and other telephone network elements is carried over a separate network of signaling links. SDN applies these ideas to packet-switched networks, adding features such as flow-based routing.

Table I. Focus Areas of Selected Papers

Project	Focus Network	OpenFlow Compatibility	PHY Layer Programmability	Research Thrust
Yap et al. [2010]	Campus WiFi	Yes	No	Network Slicing
Yiakoumis et al. [2011]	Home Networks	Yes	No	Network Slicing
Suresh et al. [2012]	Enterprise WLAN	Yes	No	Mobility Management
Jin et al. [2013]	Core Cellular	No	No	Network Management
Gudipati et al. [2013]	Cellular RAN	No	Yes	Scalability and Resource Management
Dely et al. [2011]	Mesh Networks	Yes	No	Feasibility study
Luo et al. [2012]	WSN	Yes	No	Network Management
Chen and Krishnamachari [2011]	WSN	No	No	Throughput Maximization
Gnawali et al. [2006]	WSN	No	No	Simplified Application Development

The idea of exposing the resources at individual nodes in a network in an effort to make it easier to introduce new network services is a goal that current SDN efforts share with the research on active networks. Active networking may be credited with pioneering the notion of introducing programmable functions to the network to lower the barrier to network innovation [Feamster et al. 2013]. Active networking focused on improving the data plane functionality of the network. Certain variants of active networking, specifically the capsule model, mandated that new data plane functionality be installed on network devices through code carried through data packets. Projects such as PlanetLab [Chun et al. 2003] feature the separation of traffic to different execution environments on the basis of packet headers. SDN efforts differed from active networking by focusing on problems of immediate import to network administrators. Moreover, a lot more attention has been given to increasing the programmability of the control plane. Together with commercial successes such as Nicira's Network Virtualization Platform [Nicira 2012], these efforts have led to significant industry attention being given to SDN.

The application of these concepts in the context of wireless networks poses many challenges. Consider a WLAN. Each access point (AP) has to make decisions on its modulation format, power, and channel based on SINR estimates. In this case, a fully centralized network architecture imposes strict upper bounds on the latency between the controller and the AP. The control decision should reach the AP before the channel state information, from which the decision was derived, has become obsolete. Roughly, the latency should be on the order of the coherence time of the channel. Thus, in wireless networks, it is not always clear as to which point in the design space one should operate.

In the following sections, we review the major design attempts at bringing SDN concepts to wireless networks. The designs are classified according to target networks, for example, WLAN or cellular. A brief overview of the similarities and differences between these works is shown in Table I. These have been expanded on in more detail in the following sections.

2. WLAN

Much of the research in wireless SDN so far has focused on IEEE 802.11 networks. Perhaps this has to do with the fact that network devices are not as closely tied to

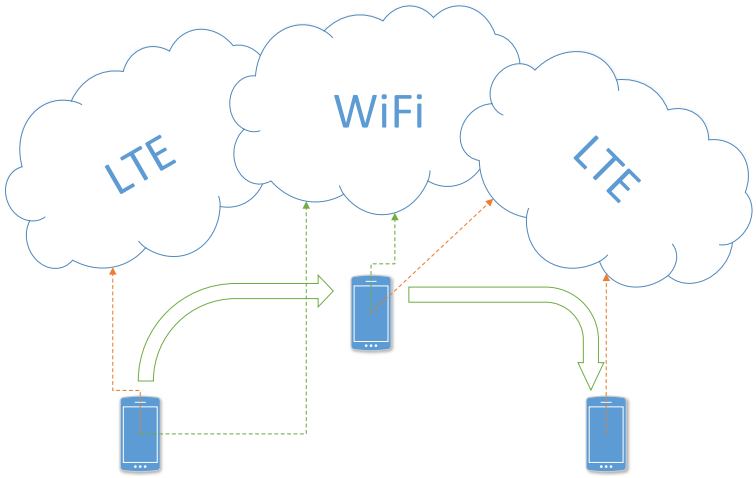

Fig. 2. Mobility across technologies.

the architecture as it is in the case of cellular networks. However, with the ubiquity of high-definition video content, there is considerable interest on the part of cellular companies in offloading data traffic to WiFi networks whenever possible. Thus, SDN efforts in WiFi and cellular are not entirely isolated from each other. There has been some work also on IEEE 802.16 (WiMax) networks, including a demonstration of the handover between WiFi and WiMAX on Openflow Wireless [Yap et al. 2010]. There is an ongoing effort to apply SDN in wireless backhaul network focusing on IEEE 802.16 [IEEE 2013], but it is recent and only beginning to be studied.

In the following subsections, we discuss the major efforts in wireless SDN targeted at WLANs.

2.1. OpenFlow Wireless

In campus WiFi networks, network administration is mostly centralized. Commercial products from various companies [Aruba 2013; Meraki 2011, 2013; Ericsson 2012] stand as a testimony to this fact. OpenFlow Wireless [Yap et al. 2009, 2010] aims to be an open alternative to the proprietary solutions currently being offered. Yap et al. [2010] envisage a future where the end user is free from worrying about the details as to which wireless network she is getting service from. It offers a compelling vision, where a user roams freely between cellular and WiFi networks taking advantage of seamless handovers (see Figure 2 for an illustration). There are obvious advantages for the consumer, such as enhanced coverage and an order-of-magnitude increase in capacity. Network operators benefit as well. The ubiquity of high-definition video content places enormous pressure on the cellular data infrastructure, which would gladly welcome the offloading of traffic to WiFi. Leaving aside the economic and regulatory challenges for the moment, the realization of this vision would require decoupling of service providers and network owners. This decoupling already exists in the present day. H20 Wireless, Tesco Mobile and LycaMobile are mobile virtual network operators (MVNOs) offering services by leasing network resources from AT&T, O2, and Telstra, respectively. An open research problem in this domain is the design of a mobility manager capable of servicing each customer.

An important feature of SDN-enabled WLANs is virtualization. The ability to slice the network, based on users, subnets, or traffic, allows many benefits. Note that flows of different slices are isolated from each other. The isolation provided by slicing enables

one to run experiments safely without affecting production traffic. This enables one to test out new features safely and push updates only upon being convinced of its stability. Another happy consequence is the delegation of management responsibilities for different parts of a network. In OpenFlow-enabled networks, this virtualization is achieved using FlowVisor [Sherwood et al. 2010] to delegate the control of different slices to different controllers. The FlowVisor is essentially a proxy that forwards OpenFlow messages from different slices to the appropriate controllers. Although a FlowVisor achieves slicing of flows, it does not provide facilities for configuration of the network. Independent configuration per slice is achieved using a SNMPVisor.

Home networks form a specific type of WLAN. In a home network, one of the primary objectives is to enable seamless sharing of data between various devices in the home while restricting access to other devices. One of the primary bottlenecks in realizing this goal is the need for network configuration by the end user. More often than not, home users are unwilling to undertake this task. In the same manner mentioned earlier, Yiakoumis et al. [2011] proposed slicing of the home network to overcome some of these problems. Specifically, separating the control and data plane allows outsourcing of network configuration and management. While the idea of allowing service providers or a separate network management company to configure the home network immediately suggests itself, it is not hard to imagine application providers taking an interest as well. Streaming applications like YouTube or Netflix can benefit greatly from a customized network slice. Similar to the MVNOs mentioned earlier, slicing also enables sharing of network infrastructure among different service providers, thereby lowering costs.

Another approach to ease the burden of network configuration on home users is to virtualize the AP by providing each user with their own personalized virtual access point [Yiakoumis et al. 2014]. Using a virtual access point to ensure uniform connectivity across different physical APs is an idea that we shall encounter again. The motivating idea for this approach is that it allows the tracking of a user across physical APs, thereby allowing user's policies, such as prioritizing certain classes of traffic, to be enforced across the network.

However, it is not immediately clear how much of network management and control can be centralized in view of the fact that wireless channel conditions are variable. Given the latency between the controller and the home network, the controller's view of the network state differs from that of the AP. Consequently, the choice of taking a control decision centrally must be made after careful evaluation of the network parameters that the control decisions depend on.

2.2. Software-Defined Radios

The developments stated so far have focused on the network layer and above. OpenRadio [Bansal et al. 2012] aims to provide programmability of the PHY and MAC layers by attempting to define a software abstraction layer that hides the hardware details from the programmer. Wireless protocols are decomposed to *processing* blocks and *decision* logic. Processing blocks operate on and manipulate the analog waveform, while the decision logic charts a path traversing different processing blocks at various times. While it is unclear if all wireless protocols lend themselves to such a simplification, progress has been made in the case of WiFi. Key advantages claimed by OpenRadio include modular programmability and the ability for real-time implementation on commodity digital signal processors.

OpenRadio is not unique in offering programmable PHY and MAC layers. Other efforts in this direction include WARP [Khattab et al. 2008]. See Table II for a listing of active projects. CloudMAC [Dely et al. 2012], on the other hand, is a network architecture aimed at having a programmable MAC layer without resorting to software radios. To this end, CloudMAC uses virtual access points (VAPs). Each physical access

Table II. Software-Defined Radio (SDR) Projects

Project name	Website
WARP [Khattab et al. 2008]	http://warp.rice.edu/trac/wiki/Radio Board
GNRRadio [Blossom 2004]	http://gnuradio.org/redmine/
AirBlue [Ng et al. 2010]	http://asim.csail.mit.edu/redmine/projects/airblue
Sora [Tan et al. 2011]	http://research.microsoft.com/en-us/projects/sora/
OpenRadio [Bansal et al. 2012]	http://snsg.stanford.edu/projects/openradio/
SDC [Shishkin et al. 2011]	http://wireless.ece.drexel.edu/research/sdct.php

point, now termed as *wireless termination point (WTP)*, is "dumb" in that they do not generate their own MAC frames. WTPs are used only to send and receive MAC frames from an OpenFlow switch that they are connected to, in addition to the end user. The OpenFlow switch in turn is connected to a controller and to virtual machines that manage the VAPs. Thus, all MAC frames generated by the user will have to travel to the VAPs, which could potentially lie somewhere deep in the network. This decreases delay performance considerably. Indeed, their implementation shows a three fold increase in round trip time (RTT).

Ultimately, these efforts point to a future where we are able to modify the entire wireless stack to suit our requirements. Along with this ability comes the question of designing an infrastructure that enables its optimal utilization. For example, the traditional centralized controller architecture of SDN might prove too slow in reacting to PHY or MAC layer incidents. Thus, it is not clear what the right demarcation of control should be. There is a trade-off in delay as one moves more and more intelligence to the center of the network, which might not be acceptable as we move lower in the stack. The optimal operating point depends on the specific network requirements.

2.3. Odin

Odin [Suresh et al. 2012] is an SDN framework that proposes to simplify the implementation of high-level enterprise WLAN services, such as authentication, authorization and accounting (AAA), by introducing light virtual access points (LVAPs). This approach is similar to the VAPs used in CloudMAC. Usually, the AP to which a user is connected to may change in accordance with local decisions made by the user. Thus, the last hop connecting the user to the WLAN infrastructure is not stable. Using the abstraction of LVAP, Odin gives programmers a virtual unchanging link connecting users to APs. To achieve this, each user is assigned a unique BSSID, giving the illusion that it has its own AP. This virtual user-specific AP is called an LVAP. Each physical AP will host multiple LVAPs, one corresponding to each client.

Odin is realized on top of an OpenFlow-enabled network. Architecturally, Odin consists of a single master, which is an application running on top of an OpenFlow controller, and multiple agents running on APs. The control channel is a TCP connection between the agent and the master. The advantages of moving to a centralized architecture, such as easy load balancing and mitigating the hidden terminal problem, are preserved. Moreover, it is easier to implement mobility managers because the BSSID at the user does not change during a handover.

Aeroflux [Schulz-Zander et al. 2014] builds on the aforementioned framework and proposes the division of the control plane into two tiers. The lower tier, managed by near-sighted controllers (NSCs), is responsible for events that do not require global state data or those events that occur very frequently. Events such as network monitoring or load balancing, which are global in nature, are handled by a global controller (GC).

3. CELLULAR

There have been two main attempts at utilizing SDN concepts to improve cellular networks. SoftRAN [Gudipati et al. 2013] focuses on improving the design of the radio access network (RAN). RAN is the part of the cellular network architecture that is burdened with providing wide area connectivity to mobile devices. Owing to the scarcity of spectrum resources, cellular providers are interested in minimizing spectrum usage while maintaining connectivity for users. SoftRan argues for improved management of radio resources to achieve this objective. CellSDN [Li et al. 2012] and SoftCell [Jin et al. 2013] constitute the major efforts toward using SDN concepts to improve the core cellular network. The following sections discuss the design principles of these works.

3.1. Core Network

Current cellular networks consist of base stations (BS) connected to server gateways (S-GW), which in turn are connected to packet gateways (P-GW). The S-GW acts as a mobility manager, maintaining state information for each user to ensure uninterrupted connectivity while the user travels across base stations. The P-GW is responsible for policy enforcement and accounting. Note that all traffic to the Internet must go through the P-GW. This inherently centralized architecture of cellular networks imposes certain restrictions. Traffic between users on the same network is needlessly routed through the P-GW, which, in addition to making the device more expensive, is slower. CellSDN [Li et al. 2012] is to distribute the processing load over the switches and base stations while retaining centralized control over them using a controller. Among the various benefits of this approach is the fact that a global view of power allocations across subcarriers at each base station enables us to make better power allocation decisions than what one could have hoped for using a distributed algorithm, as is done at present.

Implementing the architecture sketched earlier [Li et al. 2012], however, would require extensions to both switches and base stations. Each of these devices should now run cell agents that enable remote control of resources. Moreover, as was mentioned in the introduction, latency considerations would dictate which functionality can be handed over to the controller. Overall, the goal is to have a network operating system running over the cellular infrastructure, there enabling various network management requirements to be written as application modules. One can also imagine slicing network resources using FlowVisor to enable virtualization.

SoftCell [Jin et al. 2013] extends CellSDN by designing an architecture that supports fine-grained policies for mobile devices while still allowing the use of commodity switches and servers. This is achieved by employing fine-grained packet classification at the access switches in the base stations. In fact, all packet classification is done at the access edge. The gateway switches perform only basic packet forwarding with the help of policy identifiers embedded directly in the IP packet headers. Support for such policies face the challenge of having to work with small switch tables. This obstacle is circumvented by aggregating traffic based on policy, location, and user equipment ID.

3.2. Radio Access Network

One way of dealing with the increasing mobile data traffic and the limited availability of spectrum resources is to bring the base station closer to the mobile client. This approach would mandate a smaller cell size, leading to a denser deployment. Gudipati et al. [2013] argue that a denser deployment calls for an increase in coordination between neighboring base stations to improve interference management and load balancing. To this end, SoftRAN abstracts out all radio resources of a geographical area in a three-dimensional grid of base station index, time, and frequency slots. A *geographical*

area is defined as a macrocell. The physical base stations are then viewed as individual radio elements that are controlled by a logically centralized controller. The radio elements and the controller communicate with each other using a suitable API. In their architecture design, the authors note that one of their biggest challenges has been to accommodate the latency between the controller and the radio elements. Thus, in view of the varying channel conditions, the controller is effectively working with a possibly outdated view of the network state. Consequently, not all the control plane functionality is given over to the controller. All control decisions that affect the neighboring radio elements are taken at the controller. The control decisions that depend on parameters that are known to vary frequently are taken locally at the radio elements. For example, handovers and power allocation fall under the purview of the controller, while the downlink resource block allocation is handled by the radio elements. The network state is maintained at the controller in the form of a database, known as RAN information base (RIB). The RIB maintains information required by the control modules that decide on the allocation of various resources such as frequency, in addition to the power allocation at the radio elements. The feasibility of this architecture was shown through analysis. A demonstration of the performance properties of the proposed architecture in hardware would be of great interest.

4. MULTIHOP WIRELESS NETWORKS

As will be discussed shortly, there have been multiple research efforts in multihop networks that are in the spirit of SDN while differing in terminology. The following subsections provide an overview of these works.

4.1. Mesh Networks

Dely et al. [2011] provide a feasibility study of using OpenFlow in wireless mesh networks. One of the primary challenges that mesh networks have to face, in contrast to WLAN and cellular networks, is frequent changes in topology. Nodes may be mobile, and nodes arrive and leave more frequently. The proposed network architecture is as follows. All nodes are OpenFlow-enabled mesh routers complying with the IEEE 802.11 standard. A single-channel single radio setup is assumed. Each wireless interface is used for both control and data traffic. Separation between the two is achieved using different SSIDs. A NOX controller is used for managing the forwarding tables of individual nodes and handling node mobility.

Experiments conducted on the KAUMesh testbed show that on increasing the complexity of a rule, throughput suffers. A simple rule matches the port number of an incoming packet, whereas a complex rule might use any part of the header or even the payload. In the implementation mentioned earlier, a complex rule consisted of matching the MAC and IP addresses for each packet. This was shown to lead to a throughput degradation of up to 15%. The control traffic was shown to be small, albeit linearly increasing with the number of rules to be installed as might be expected. It is not quite clear at what point the control traffic begins to pose a significant overhead.

4.2. Wireless Sensor Networks

A wireless sensor network (WSN) is another multihop wireless scenario where an SDN approach has been claimed to be beneficial [Luo et al. 2012]. WSNs are subject to a unique set of constraints. The nodes are typically low powered, small and are often deployed without any particular attention to the topology they form. In any case, a change in topology post deployment is quite common due to node failures or physical displacement of nodes. Nodes may also contain application specific components. Furthermore, nodes are assumed to be autonomous and be adaptive to their environment. The high variability of WSNs and its application-specific nature make these networks

difficult to manage. Luo et al. [2012] propose a SDN-based architecture that proposes to centralize network management and enable running different applications on a single WSN.

This proposed architecture for WSNs is conceptually quite similar to the various wireless SDN efforts described previously. The control plane is decoupled from the data plane that runs on the sensor nodes. A centralized controller uses a customized version of OpenFlow to interact with the nodes. The nodes are modified to enable this centralized control of their flow tables. Various applications may be run on top of the controller. The primary contribution of the article lies in the specifics of extending OpenFlow to support WSN specific use cases, such as flow creation using sensor attributes. However, in contrast with the approaches discussed thus far, control and data traffic share the same network. As might be expected, this increases the average latency of the control channel. Another problem that remains unaddressed is the reduced reliability of the control channel. Compared to the data network, the control network is typically subject to better standards of reliability. Particularly in WSNs, as node and link failures are much more common, the design choice of sharing the network remains dubious. Other difficulties that need to be worked around include the need to minimize control overhead as communication is inherently costly.

There have been other efforts in centralizing control in WSNs that are in the same spirit as SDN, but the terminology pertaining to SDN has not been used. MCC [Chen and Krishnamachari 2011] is multichannel time-scheduled protocol aimed toward real-time data collection in WSNs. MCC design features centralized channel allocation and time scheduling to combat co-channel interference, and routes from sources to the sink are centrally computed using a capacitated minimal spanning (CMS) tree heuristic. Thus, there is a centralized control plane that obtains information about the network topology to configure the transmission and routing decisions for all nodes to use for the distributed data forwarding plane. This makes the protocol design of MCC architecturally close to SDN.

Tenet [Gnawali et al. 2006] is another architecture design that decouples control from the sensor motes. Gnawali et al. [2006] argue for a tiered architecture for WSNs. The lower tier consists of resource constrained motes, whereas the upper tier contains fewer but more capable nodes called *masters*. In the words of the authors, the Tenet design principle may be stated as:

> Multi-node data fusion functionality and multi-node application logic should be implemented only in the master tier. The cost and complexity of implementing this functionality in a fully distributed fashion on motes outweighs the performance benefits of doing so.

In fact, Tenet represents an extreme design point that severely constraints even the type of communication allowed on the motes. Thus, the motes only provide a limited set of generic functions and applications, written in software in the master tier, combine this functionality to achieve network objectives. These properties are characteristic of SDN.

5. CONCLUSIONS

SDN opens many axes in the network design space. See Table I for a comparison of selected projects. The most important of these is the possibility of centralizing much of the intelligence in a network. For wireless networks, this move comes with obvious advantages. For example, the hidden terminal problem ceases to be an issue if transmission decisions are made centrally, based on a view of the entire network. Virtualization of network resources is now possible, enabling sharing of resources across vendors, thereby reducing cost. However, the increased channel variability and the latency sensitive nature of key network parameters, such as power allocation in a

channel, increases the complexity of the design space. In fact, while research thus far has focused on enabling choice in different design axes, the larger question of how to make use of this new-found freedom optimally for different scenarios remains unanswered.

REFERENCES

Aruba. 2013. Aruba Networks. Retrieved from http://www.arubanetworks.com.

Manu Bansal, Jeffrey Mehlman, Sachin Katti, and Philip Levis. 2012. OpenRadio: A programmable wireless dataplane. In *Proceedings of the 1st Workshop on Hot topics in Software Defined Networks (HotSDN'12)*. ACM, New York, NY, 109–114. DOI:http://dx.doi.org/10.1145/2342441.2342464

Eric Blossom. 2004. GNU radio: Tools for exploring the radio frequency spectrum. *Linux J.* 2004, 122 (June 2004), 4.

Matthew Caesar, Donald Caldwell, Nick Feamster, Jennifer Rexford, Aman Shaikh, and Jacobus van der Merwe. 2005. Design and implementation of a routing control platform. In *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation - Volume 2 (NSDI'05)*. USENIX Association, Berkeley, CA, 1528.

Martin Casado, Michael J. Freedman, Justin Pettit, Jianying Luo, Nick McKeown, and Scott Shenker. 2007. Ethane: Taking control of the enterprise. *SIGCOMM Comput. Commun. Rev.* 37, 4 (Aug. 2007), 1–12. DOI:http://dx.doi.org/10.1145/1282427.1282382

Ying Chen and Bhaskar Krishnamachari. 2011. *MCC: A High-Throughput Multi-Channel Data Collection Protocol for Wireless Sensor Networks*. Technical Report, USC Computer Engineering, Los Angeles, CA.

Brent Chun, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak, and Mic Bowman. 2003. PlanetLab: An overlay testbed for broad-coverage services. *SIGCOMM Comput. Commun. Rev.* 33, 3 (July 2003), 3–12. DOI:http://dx.doi.org/10.1145/956993.956995

P. Dely, A. Kassler, and N. Bayer. 2011. OpenFlow for wireless mesh networks. In *Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN'11)*. 1–6. DOI:http://dx.doi.org/10.1109/ICCCN.2011.6006100

P. Dely, J. Vestin, A. Kassler, N. Bayer, H. Einsiedler, and C. Peylo. 2012. CloudMAC - An openflow based architecture for 802.11 MAC layer processing in the cloud. In *Proceedings of the 2012 IEEE Globecom Workshops (GC Wkshps'12)*. 186–191. DOI:http://dx.doi.org/10.1109/GLOCOMW.2012.6477567

Ericsson. 2012. The Cloud and Software Defined Networking. Retrieved Jan 16, 2014 from http://www.ericsson.com/res/investors/docs/2012/ericsson-cloud-and-sdn.pdf.

Nick Feamster, Jennifer Rexford, and Ellen Zegura. 2013. The road to SDN. *Queue* 11, 12 (Dec. 2013), 20 pages. DOI:http://dx.doi.org/10.1145/2559899.2560327

Omprakash Gnawali, Ki-Young Jang, Jeongyeup Paek, Marcos Vieira, Ramesh Govindan, Ben Greenstein, August Joki, Deborah Estrin, and Eddie Kohler. 2006. The tenet architecture for tiered sensor networks. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (SenSys'06)*. ACM, New York, NY, 153–166. DOI:http://dx.doi.org/10.1145/1182807.1182823

Albert Greenberg, Gisli Hjalmtysson, David A. Maltz, Andy Myers, Jennifer Rexford, Geoffrey Xie, Hong Yan, Jibin Zhan, and Hui Zhang. 2005. A clean slate 4D approach to network control and management. *SIGCOMM Comput. Commun. Rev.* 35, 5 (Oct. 2005), 41–54. DOI:http://dx.doi.org/10.1145/1096536.1096541

Aditya Gudipati, Daniel Perry, Li Erran Li, and Sachin Katti. 2013. SoftRAN: Software defined radio access network. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN'13)*. ACM, New York, NY, 25–30. DOI:http://dx.doi.org/10.1145/2491185.2491207

IEEE. 2013. IEEE 802 Tutorial - Wireless SDN in Access and Backhaul. Retrieved Jul 19, 2014 from https://mentor.ieee.org/802-ec/dcn/13/ec-13-0055.pdf.

Xin Jin, Li Erran Li, Laurent Vanbever, and Jennifer Rexford. 2013. SoftCell: Scalable and flexible cellular core network architecture. In *Proceedings of the 9th ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT'13)*. ACM, New York, NY, 163–174. DOI:http://dx.doi.org/10.1145/2535372.2535377

Ahmed Khattab, Joseph Camp, Chris Hunter, Patrick Murphy, Ashutosh Sabharwal, and Edward W. Knightly. 2008. WARP: A flexible platform for clean-slate wireless medium access protocol design. *SIGMOBILE Mob. Comput. Commun. Rev.* 12, 1 (Jan. 2008), 56–58. DOI:http://dx.doi.org/10.1145/1374512.1374532

L. E. Li, Z. M. Mao, and J. Rexford. 2012. Toward software-defined cellular networks. In *Proceedings of the 2012 Eurpean Workshop on Software Defined Networking (EWSDN'12)*. 7–12. DOI:http://dx.doi.org/10.1109/EWSDN.2012.28

Tie Luo, Hwee-Pink Tan, and T. Q. S. Quek. 2012. Sensor OpenFlow: Enabling software-defined wireless sensor networks. *Communications Letters, IEEE* 16, 11 (2012), 1896–1899. DOI:http://dx.doi.org/10.1109/LCOMM.2012.092812.121712

Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. 2008. OpenFlow: Enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.* 38, 2 (March 2008), 69–74. DOI:http://dx.doi.org/10.1145/1355734.1355746

Meraki. 2011. Meraki Whitepaper: Meraki Hosted Architecture. Retrieved Jan 21, 2014 from https://meraki.cisco.com/lib/pdf/meraki_whitepaper_architecture.pdf.

Meraki. 2013. Cloud Managed Wireless by Meraki. Retrieved from http://www.meraki.com/products/wireless.

Man Cheuk Ng, Kermin Elliott Fleming, Mythili Vutukuru, Samuel Gross, Arvind, and Hari Balakrishnan. 2010. Airblue: A system for cross-layer wireless protocol development. In *Proceedings of the 6th ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS'10)*. ACM, New York, NY, Article 4, 11 pages. DOI:http://dx.doi.org/10.1145/1872007.1872013

Nicira. 2012. It'sTime to Virtualize the Network. Retrieved Jan 16, 2014 from http://www.netfos.com.tw/PDF/Nicira/ItisTimeToVirtualizetheNetworkWhite Paper.pdf.

Julius Schulz-Zander, Nadi Sarrar, and Stefan Schmid. 2014. AeroFlux: A near-sighted controller architecture for software-defined wireless networks. USENIX Association, Santa Clara, CA. Retrieved from https://www.usenix.org/conference/ons2014/technical-sessions/presentation /shulz-zander.

Rob Sherwood, Michael Chan, Adam Covington, Glen Gibb, Mario Flajslik, Nikhil Handigol, Te-Yuan Huang, Peyman Kazemian, Masayoshi Kobayashi, Jad Naous, Srinivasan Seetharaman, David Underhill, Tatsuya Yabe, Kok-Kiong Yap, Yiannis Yiakoumis, Hongyi Zeng, Guido Appenzeller, Ramesh Johari, Nick McKeown, and Guru Parulkar. 2010. Carving research slices out of your production networks with OpenFlow. *SIGCOMM Comput. Commun. Rev.* 40, 1 (Jan. 2010), 129–130. DOI:http://dx.doi.org/10.1145/1672308.1672333

B. Shishkin, D. Pfeil, D. Nguyen, K. Wanuga, J. Chacko, J. Johnson, Nagarajan Kandasamy, T. P. Kurzweg, and K. R. Dandekar. 2011. SDC testbed: Software defined communications testbed for wireless radio and optical networking. In *Proceedings of the 2011 International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt'11)*. 300–306. DOI:http://dx.doi.org/10.1109/WIOPT.2011.5930031

Lalith Suresh, Julius Schulz-Zander, Ruben Merz, Anja Feldmann, and Teresa Vazao. 2012. Towards programmable enterprise WLANS with Odin. In *Proceedings of the 1st Workshop on Hot Topics in Software Defined Networks (HotSDN'12)*. ACM, New York, NY, 115–120. DOI:http://dx.doi.org/10.1145/2342441.2342465

Kun Tan, He Liu, Jiansong Zhang, Yongguang Zhang, Ji Fang, and Geoffrey M. Voelker. 2011. Sora: High-performance software radio using general-purpose multi-core processors. *Commun. ACM* 54, 1 (Jan. 2011), 99107. DOI:http://dx.doi.org/10.1145/1866739.1866760

TR10. 2009. Software Defined Networking. Retrieved from http://www.technologyreview.com/article/412194/tr10-software-defined-networking/.

Kok-Kiong Yap, Masayoshi Kobayashi, Rob Sherwood, Te-Yuan Huang, Michael Chan, Nikhil Handigol, and Nick McKeown. 2010. OpenRoads: Empowering research in mobile networks. *SIGCOMM Comput. Commun. Rev.* 40, 1 (Jan. 2010), 125–126. DOI:http://dx.doi.org/10.1145/1672308.1672331

Kok-Kiong Yap, Masayoshi Kobayashi, David Underhill, Srinivasan Seetharaman, Peyman Kazemian, and Nick McKeown. 2009. The stanford openroads deployment. In *Proceedings of the 4th ACM International Workshop on Experimental Evaluation and Characterization (WINTECH'09)*. ACM, New York, NY, 59–66. DOI:http://dx.doi.org/10.1145/1614293.1614304

Kok-Kiong Yap, Rob Sherwood, Masayoshi Kobayashi, Te-Yuan Huang, Michael Chan, Nikhil Handigol, Nick McKeown, and Guru Parulkar. 2010. Blueprint for introducing innovation into wireless mobile networks. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (VISA'10)*. ACM, New York, NY, 25–32. DOI:http://dx.doi.org/10.1145/1851399.1851404

Yiannis Yiakoumis, Manu Bansal, Sachin Katti, and Nick McKeown. 2014. SDN for Dense WiFi Networks. USENIX Association, Santa Clara, CA.

Yiannis Yiakoumis, Kok-Kiong Yap, Sachin Katti, Guru Parulkar, and Nick McKeown. 2011. Slicing home networks. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Home Networks (HomeNets'11)*. ACM, New York, NY, 1–6. DOI:http://dx.doi.org/10.1145/2018567.2018569

Received June 2013; revised July 2014; accepted July 2014