EURASIP Journal on
Wireless Communications and Networking
a SpringerOpen Journal

REVIEW Open Access

# Building programmable wireless networks: *an architectural survey*

Junaid Qadir[1*], Nadeem Ahmed[1] and Nauman Ahad[1,2]

**Abstract**

In recent times, there is increasing consensus that the traditional Internet architecture needs to be evolved for it to sustain unstinted growth and innovation. A major reason for the perceived architectural ossification is the lack of the ability to program the network as a system. This situation has resulted partly from historical decisions in the original Internet design which emphasized decentralized network operations through colocated data and control planes on each network device. The situation for wireless networks is no different resulting in a lot of complexity and a plethora of largely incompatible wireless technologies. With traditional architectures providing limited support for programmability, there is a broad realization in the wireless community that future programmable wireless networks would require significant architectural innovations. In this paper, we will present an unified overview of the programmability solutions that have been proposed at the device and the network level. In particular, we will discuss software-defined radio (SDR), cognitive radio (CR), programmable MAC processor, and programmable routers as device-level programmability solutions, and software-defined networking (SDN), cognitive wireless networking (CWN), virtualizable wireless networking (VWN) and cloud-based wireless networking (CbWN) as network-level programmability solutions. We provide both a self-contained exposition of these topics as well as a broad survey of the application of these trends in modern wireless networks.

## 1 Introduction

Wireless networks have become increasingly popular due to the inherent convenience of untethered communication. They are deployed ubiquitously in myriad of networking environments ranging from cellular mobile networking, regional or city-wide networking (e.g., through worldwide interoperability for microwave access (WiMAX) technology), and local area or even personal networking environments (e.g., through Wi-Fi and Bluetooth technology, respectively) [1]. With the usage of wireless networks promising to increase in the future, both in demand and application diversity [2], the issue of devising and implementing flexible architectural support becomes all the more important.

While newer wireless technologies have been emerging at a prolific rate, the architecture of wireless networking has largely been static and difficult to evolve. The malaise of architectural 'ossification' is not unique to wireless networking though but applies more generally to networking. Before we can describe the reasons of this ossification, we operationally define the *data plane* to be responsible for forwarding packets at line speed, and the *control plane* for figuring out, and instantiating, the forwarding state that the data plane needs. Various reasons have been offered to explain the Internet's architectural ossification such as: *i*) vertical integration and coupling of the data plane and the control plane at node level; *ii*) lack of abstractions and modularization of the control plane, and finally, resulting from the preceding two reasons; and *iii*) lack of programmability of the network as a whole. These reasons, subtly related to each other, have collectively discouraged networking growth and innovations [3].

To manage the complexity of computer systems, computer scientists have long recognized the potency of the concept of abstraction [4]. It has been argued that the most formidable challenge to the networking industry is posed by the paucity of useful abstractions [5]. With a lack of foundational abstractions, networking reduces to a 'plethora of protocols and tools' without any underlying architectural base [5,6]. There are three main benefits

*Correspondence: junaid.qadir@seecs.edu.pk
[1] School of Electrical Engineering and Computer Sciences (SEECS), National University of Sciences and Technology (NUST), Islamabad 44000, Pakistan
Full list of author information is available at the end of the article

of using *abstractions*: *i*) modularity, which allows managing complex problems scalably through reuse of modules offering common functionality; *ii*) separation of concerns through loose inter-module coupling which is implementation agnostic; and *iii*) innovation, since new developments can focus on the module that needs fixing without 'reinventing the entire wheel' [7]. The Open System Interconnection (OSI) layered model - which is composed of layers representing modular subcomponents that interact through well-defined abstract interfaces - is often considered as a major success story of computer networking. The OSI layering model, however, relates mostly to the Internet's data plane which has evolved to offer various useful abstractions [6]. The Internet's control plane, on the other hand, has developed mostly in an ad hoc fashion and has lacked well-developed abstractions until quite recently [8].

Broadly speaking, *programmable networks* denote networks that can tune itself, or reconfigure itself, through a software-based adaptation - typically managed through an application programmer interface (API). Incorporation of functionality in software allows networks to innovate at the rate of software development cycle which is a lot more agile than the sluggish rate of hardware development [9]. Applications of programmability include rapid provisioning of services [10], flexible resource management [11], efficient resource sharing [12], and support for new architectures such as cloud computing, Internet of things (IoT) [13], etc.

Traditionally, networking devices have offered some ability to configure the system/network but with limited programmability. In particular, the devices mostly offered a limited set of *configuration* options which may be envisioned as 'knobs' that can be tuned to suit the operator. The vision of a programmable device is to allow the operator to *program* in any way desired; i.e., the operator should be free to define new custom tunable knobs as desired to support niche applications or services and not be limited to the configuration options made available by the device's vendor. Programmable devices thus offer far greater flexibility than configurable devices.

Interestingly, programmable networking is not entirely a recent concept. The lack of programmability of networks has long been recognized, and various approaches have been proposed to address this deficiency [14,15]. In a remarkably prescient paper, published by Campbell et al. in 1999 [14], the impending impact of numerous programming trends is anticipated using surprisingly modern terminology. In particular, it was predicted that higher levels of network programmability will result from separation of hardware from software, availability of open network interfaces, virtualization of networking infrastructure, and rapid creation and deployment of network services. These predictions have come to fruition exactly as forecasted in the forms of software-defined networking (SDN), standardized APIs, network virtualization, and cloud computing. Similarly, another insightful paper [16], the early version of which dates to 1996, talks about applying programming language perspective to networks and their protocols and talks about their aim of creating the 'Smalltalk of networking'. These ideas are having a renaissance in the modern era in the context of SDN programming languages [17][a].

To develop programmable wireless networks, it is imperative that we emphasize the development of both *programmable wireless data planes* and *programmable wireless control planes*. In this paper, we will provide a unified holistic overview of programmable wireless networking and will highlight overarching themes and insights. We will see that there are four prominent technological trends that underlie most of the current research focusing on enabling future programmable wireless networking. These technological trends are: *i*) software-defined wireless networks (SWNs), *ii*) cognitive wireless networks (CWNs), *iii*) virtualizable wireless networks (VWNs), and *iv*) cloud-based wireless networks (CbWNs). We will provide a background on these trends and will provide a broad survey of the applications of these trends in the literature.

The *main contribution of this paper* is that we provide a unified overview of the emerging field of programmable wireless networks. We demonstrate that the seemingly disparate fields of active networking, software-defined radios, cognitive radios, software-defined networking, and wireless virtualization, are in fact kindred disciplines. We develop this idea and propose new directions of future programmable wireless networks. Our paper is different from other survey papers [11,18,19] in its focus on *wireless* programmable networks while the previous papers had focused mainly on generic (wired) programmable networks.

The organization of the remaining paper is as follows. In the next section, we describe the basic *device-level* building blocks for building future programmable wireless networks. Thereafter, we discuss the various architectural approaches of building *generic programmable networks* in Section 3. We will thereafter discussing four dominant categories of programmable *wireless* networks, i.e., SWNs, CWNs, VWNs, and CbWNs, and highlight works belonging to each category in Sections 4.1, 4.2, 4.3, and 4.4, respectively. In Section 5, we discuss various open research issues and future directions of research. We conclude the paper in Section 6.

## 2 Building blocks for programmable wireless networking

Programmable devices are envisioned to be a key component of future programmable networks. In this section,

we discuss various techniques and architectures that have been proposed to realize the benefits associated with programmable wireless networks. In particular, we elaborate upon the trends of software-defined radio, cognitive radio, MAC programmable wireless devices, programmable wireless testbeds, and programmable radios in this particular order.

### 2.1 Software-defined radios (SDR)

The defining characteristic of a SDR is that it implements most of the basic building blocks of physical (PHY) layer radio communication in software. With the hardware stripped down to the elements essential to all radio communication, custom blocks traditionally implemented in hardware - e.g., filters, amplifiers, modulators, demodulators, mixers, etc. - are now implemented in software (Figure 1). This implies that appropriate programming of the generic radio hardware can in principle allow it to support arbitrary technologies. The SDR technology was a significant paradigm shift ushering in a new era of programmable wireless devices. Thus, using an SDR, an operator could program a wireless device to support any of the myriad of wireless technologies [20]. This opened up an unprecedented opportunity for creating a programmable *wireless* device for the first time [21-23].

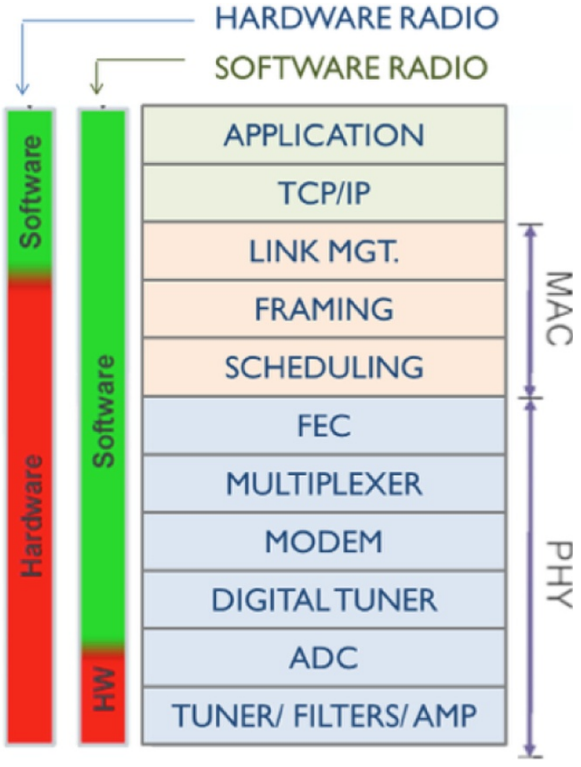


**Figure 1 The hardware/software division of radio tasks in software-defined radio vs. a traditional hardware radio.**

The precise definition of SDRs is debated, with no clear consensus on how reconfigurable must a radio be to be deemed an SDR. Clearly, it is a bit of a stretch to call every radio with a digital signal processor (DSP) as an SDR. A working definition provided in [21] is that an SDR is 'a radio that is substantially defined in software and whose physical layer behavior can be significantly altered through changes to its software'. The SDR forum defines an 'ultimate software radio' (USR) as 'a radio that accepts fully programmable traffic and control information and supports a broad range of frequencies, air-interfaces, and applications software'. [21]. In [24], two extremes SDR platforms are discussed: The first type is an SDR that is composed of programmable components, such as field programmable gate arrays (FPGAs), DSPs, etc., which are *programmed* directly; the other extreme is a highly *configurable* chipset-based SDR which is 'programmed' by setting configuration registers in the chip to determine the choice of frequency, coding, and PHY- and MAC-level protocol details. Most practical SDRs lie between these two extremes [24].

While traditionally SDRs have mainly been used in military settings due to excessive cost, the technology has now matured to a stage where its form and cost is amenable to non-military markets [24]. While the SDR of 1990s was the size of a small refrigerator and could easily cost more than $100,000, today the size of an SDR is akin to the size of a computer battery and it can cost less than $500, extrapolating the trend, it is reasonable to assume that future pricing and form factor of SDRs will match that of a typical consumer electronic device [24]. The democratization of SDR technology will conceivably revolutionize wireless and mobile networking, e.g., a consumer will not be limited to any single wireless protocol with a wireless device. This will lead to unprecedented flexibility as technologies (such as Wi-Fi and Bluetooth) will no longer be 'baked' into the hardware, but will be software applications, or applets, that any SDR could support. Due to their versatile nature, SDRs are radio chameleons potentially running a telephony protocol (such as CDMA) at a given time and switching to a completely different data communication protocol (such as Wi-Fi or WiMAX) next moment [24].

A prominent and popular example of SDR platform is to use the universal software radio peripheral (USRP) hardware kit [25] along with the open-source GNU Radio software toolkit [26] that implements in software various necessary signal processing blocks. The USRP hardware digitizes the received analog signal and imports it into a computer so that it may be processed by GNU Radio software (or similar toolkits such as the OSSIE framework based on the 'joint tactical radio system' (JTRS) software communications architecture [27]). Such an arrangement allows building a *custom radio* that can be programmed to

support an arbitrary wireless technology through appropriate signal processing blocks that operate on the signal received or to be transmitted. Some example of SDR platforms include Implementing Radio in Software (IRIS) [28], USRP [29], Wireless Open-Access Research Platform (WARP) [30], Airblue [31], SORA [32], OpenRadio [33], and Ziria [34,35]. We provide a brief description of these platforms in Table 1 and refer to the original papers for details.

To summarize, SDRs are envisioned as an essential component of future programmable wireless devices. Their importance can be gauged from the fact that almost all advanced wireless programmability techniques (such as cognitive radio and programmable wireless processors, etc.) are SDR-based.

### 2.2 Cognitive radios (CR)

CRs have evolved from the concept of SDRs [55], and it represents the next major milestone after SDR in the drive towards powerful programmable wireless *devices*. Joseph Mitola coined the term 'cognitive radio' in 1999 when he envisioned a broadening of the SDR concept. In particular, Mitola anticipated that incorporation of substantial artificial intelligence (AI) in the form of machine learning, knowledge reasoning, and natural language processing will transform SDRs into cognitive radios that will optimize network performance by sensing, learning, and reacting to environmental stimuli [56]. In other words, CR allows an SDR to reprogram itself autonomously based on network conditions. In a modern setting, this is achieved by incorporation of a cognitive engine that employs various AI-based techniques to build a knowledge base, based on which reasoning is performed to make 'optimal' decisions [57].

CRs are viewed as an essential component of next-generation wireless networks [58,59] and have a wide range of applications including intelligent transport systems, public safety systems, femtocells, cooperative networks, dynamic spectrum access, and smart grid communications [57,58]. CR can dramatically improve spectrum access, capacity, and link performance while also incorporating the needs and the context of the user [57]. Although cognitive behavior of CRs can enable diverse applications, perhaps the most cited application of cognitive radio networks (CRNs), which are networks where nodes are equipped with CRs, is dynamic spectrum access (DSA) [60]. DSA is proposed as a solution to the problem of 'artificial spectrum scarcity' that results from static allocation of available wireless spectrum using the command-and-control licensing approach [60]. Under this approach, licensed applications represented by primary users (PUs) are allocated exclusive access to portions of the available wireless spectrum prohibiting other users from access even when the spectrum is idle. With most of the radio spectrum already being licensed in this fashion, innovation in wireless technology is constrained. The problem is compounded by the observation, replicated in numerous measurement-based studies world over, that the licensed spectrum is grossly underutilized [58,60]. The DSA paradigm proposes to allow secondary users (SUs) access to the licensed spectrum subject to the condition that SUs do not interfere with the operations of the primary network of incumbents.

While programmable wireless devices (such as SDRs and CRs) do serve as the building block for programmable wireless networking infrastructures, it is pertinent to note here that the task of building programmable wireless *networks* is much more nuanced. Various vexing problems (such as routing [61], security [62], etc.) need to be solved while taking into account the network-wide behavior [63]. Historically, most of the CR research has focused on optimizing at a device level, with network-level programmability being a recent afterthought [64]. In the subsequent sections, we will see how trends of software-defined networks (Section 4.1) and cognitive networks (Section 4.2) allow us to extend the programmability concepts to network proportions.

### 2.3 MAC programmable wireless devices

In the past few years, numerous new wireless technologies, with distinct MAC protocols, have been proposed to serve a variety of niche wireless applications. Since there is no universal and one-size-fits-all MAC protocol that will work equally well for all such scenarios, there is a lot of interest in creating programmable wireless devices which will implement, what may be effectively called, *software-defined MAC*. A majority of current wireless devices do not support SDRs, or even software-defined MAC, and effectively can support only a single technology. Although SDRs offer great flexibility in altering its PHY later characteristics, supporting programmatic MAC on SDRs also entails significant research challenges [32,65].

In recent times, there has been work in supporting programmable, or software-defined MAC, on *commodity wireless devices*. In 2005, Neufeld et al. proposed SoftMAC [45] as a software system that allows development of new MAC protocols on commodity 802.11 network interface cards (NICs) to build a SDR with predefined PHY characteristics but with a flexible MAC layer. SoftMAC was extended by the MultiMAC framework [46], proposed by Doerr et al., to provide support for DSA applications in CRNs. In particular, MultiMAC supports multiple concurrent MAC layers that can be switched with minimal impact. MultiMAC aimed at dynamically reconfiguring MAC and PHY layer properties to select appropriate MAC component on per node/flow basis thereby acting as a stepping stone to the concept of 'cognitive MAC'

**Table 1 Representative summary of important programmable networking components and platforms**

| Component category | Project and reference | Year | Brief summary |
|---|---|---|---|
| Software-defined radio (SDR) platforms | | | |
| | IRIS [28] | 2004 | Implementing Radio in Software (IRIS) project developed at Trinity College, Dublin |
| | USRP [29] | 2005 | Flexible SDR development platform, often used with GNUradio, manufactured by Ettus/ NI |
| | WARP [30] | 2008 | Wireless Open-Access Research Platform (WARP) developed by Rice University |
| | Airblue | 2010 | An FPGA based SDR platform that can run at high speeds compatible to commodity hardware |
| | SORA [32] | 2011 | Programmable SDR platform, developed by Microsoft, for commodity multi-core PCs |
| | OpenRadio [33] | 2012 | Programmable wireless data plane that can programmed across the wireless stack |
| | Ziria [34,35] | 2013 | SDR framework developed at Microsoft, comprising a programming language and an optimizing compiler, that can synthesize efficient SDR code from high-level PHY description |
| Cognitive radio (CR) platforms | | | |
| | BEE2 [36] | 2005 | Reconfigurable hardware platform developed at University of California, Berkeley |
| | KNOWS [37] | 2007 | CR hardware platform, for operation in TV white spaces, developed by Microsoft |
| | WinC2R [38] | 2008 | CR hardware platform developed by the WINLAB at Rutgers University |
| Programmable network components | | | |
| | Virtual Switches [39] | 1996 | Proposed virtualizing ATM switches as part of the xbind [39] project (OpenSig framework) |
| | Switchlets [40] | 1998 | Proposed dynamically loadable code on a (partition of) ATM switches as part of the Tempest [40] project subscribing to OpenSig framework |
| | Virtual base stations [41] | 1998 | Proposed as part of the Mobiware [41] project subscribing to the OpenSig framework |
| | Routelets [42] | 1999 | proposed in the Genesis [42] project subscribing to the OpenSig framework |
| | Click [43] | 1999 | Software architecture for building flexible and configurable routers |
| | XORP [44] | 2003 | An open programmable router platform for research experimentation |
| | SoftMAC [45] | 2005 | Proposed a software system for developing new MAC protocols on commodity 802.11 NICs |
| | MultiMAC [46] | 2005 | Dynamically reconfigures MAC and PHY properties to select appropriate MAC component on per node/ flow basis |
| | NetFPGA [47,48] | 2007 | Programmable and extensible router with embedded FPGA |
| | RouteBricks [49] | 2009 | Software router architecture (Click based) that parallelizes router functionality |
| | SwitchBlade [50] | 2010 | FPGA-based platform for deploying custom protocols with programmability and performance |
| | Ansari et al. [51] | 2010 | Programmable decomposable MAC framework |
| | TRUMP [52] | 2011 | Programmable component-based MAC framework |
| | Wireless MAC processor [53] | 2012 | Composition of custom MAC protocols by programming with basic MAC commands |
| | MAClets approach [54] | 2012 | Programmable framework that allows installing MAC stacks as 'applets' |

which supports intelligent reconfiguration of MAC and PHY layers.

Other ways have also been proposed for building MAC programmable wireless devices. One way of doing this is by creating an abstraction of a *wireless MAC processor* with an instruction set representing common MAC actions, events, and conditions which can be programmed through an API to compose any custom MAC protocol [53]. Another approach, known as the MAClet approach, is to conceive the entire MAC protocol stack akin to a Java applet which can be loaded onto a MAC processor and run [54]. While these approaches could be conceivably implemented on FPGA-based SDR platforms, such as WARP [30] or USRP [25] in a straightforward manner, the main contribution of the works [53,54] has been to implement these approaches on a commodity Broadcom wireless NIC. In [66], a new service-oriented architecture for programmable wireless interfaces is proposed which replaces the traditional PHY and MAC layers with a *i*) *a platform layer*, which exposes static primitives for managing hardware events and frame transmissions and *ii*) *three layers of functionalities* - state machines, functions, and services - that expose a programmable interface to upper layers. The proposed approach differs from SDR solutions since the adaptation and customization is accomplished through programmable interfaces exposed at a layer higher than the PHY layer. Besides these aforementioned works [53,54,66], there have been other *component oriented design* [67] efforts for composing customizable MAC protocols from a set of basic functional components [51,52]. For a detailed survey of dynamically adaptable protocol stacks in general, the interested reader is referred to [68].

### 2.4 Programmable routers

Programmable routers have been developed that incorporate programmable datapath processing capabilities in commodity $\times 86$ architectures to perform custom protocol operations and arbitrary payload processing more flexibly and at a lower cost. These programmable routers are not specific to wireless technologies but we discuss this technology in this section because these routers can potentially be very useful in the context of programmable wireless networking. The Click programmable router [43] is an early influential software router which snaps together various modular 'elements' to assemble the router logic. Although Click offers the capability of rapid prototyping and deployment and decent performance for a software router running on a PC, any purely software-based approach will be hard pressed to satisfy the demanding performance requirements of modern networks. However, with the recent advances in the processing and input/output (I/O) performance of commodity servers, software routers running on $\times 86$ architecture offers a disruptive lower cost and more flexible proposition.

More recently, programmable routers with reprogrammable hardware such as FPGAs have been proposed to simultaneously address the needs for flexibility, extensibility, and performance for the forwarding plane. Prominent projects in this category include the NetFPGA project [47], the RouteBricks project [49], and the SwitchBlade project [50]. Extensible open-source *control plane software* also exists with the Extensible Open Router Platform (XORP) open-source software suite [44] being a prominent example; XORP defines a fully extensible platform, suitable for both research and deployment, which builds upon the extensible Click framework in its forwarding plane.

To wrap this section, we note that we have provided a representative summary of various architectural components of programmable networking, including a summary of programmable MAC devices, is provided in Table 1. The characteristics of SDR, CR Programmable MAC, and programmable routers are also juxtaposed in Table 2 for a ready comparison.

## 3 Review of programmable networking architectures

### 3.1 Traditional programmable networking architectures

With the lack of programmability complicating networking innovations, it was the early 1990s when work on creating programmable network started in earnest [19]. At the time there were two major, slightly differing schools, that advocated programmable networks: the first group proposed the 'OpenSig' approach [19] while the second group furthered the 'Active Networking' approach [69]. The general consensus that emerged was that the programmability solution lies in separating the control software from the hardware and in having open interfaces for management and control. The building blocks for creating programmable networks started appearing thereafter in the form of various programmable networking components (such as the Click modular router [43], etc.). More recently, with the emergence of datacenters, virtualization, and cloud computing technology, the requirement of programmability has become mainstream. In the remainder of this subsection, we will outline these developments in more detail.

#### *3.1.1 The OpenSig approach*

In the mid 1990s, the *OpenSig approach* [70] advocated both the separation of the data plane and the control plane for *ATM networks* and the usage of *open interfaces* for signalling between these two planes. The main idea was that with separated control and data planes, and an open standard interface, the ATM switches would become remotely programmable and thus more manageable. The

**Table 2 Comparison of SDR, programmable MAC, and CR paradigms**

| | Software-defined radio (SDR) | Cognitive radio | Programmable MAC | Programmable routers |
|---|---|---|---|---|
| Raison d'être | Software defined ability to adapt/program PHY and MAC layer characteristics. | Using 'cognition' to drive the capability of adapting (typically providing by SDRs) to optimize the network performance. | Supporting custom creation of MAC protocols, rather than hard-wired MAC, through appropriate vendor-independent abstractions. | Supporting custom data plane processing to facilitate customized processing of the packets in the data line. |
| Applications | Reconfigurability, building block of cognitive radio, interoperability, more degrees of freedom. | Dynamic spectrum access (DSA); interoperability and improved handovers; link optimization (modulation, power, topology, etc.); better resource utilization; increased capacity, reliability, and security; technology neutral coexistence. | Defines primitives for composing custom MAC protocol logic which can program the whole radio protocol stack independently of the platform (analogously to Java Applet). | Useful for developing software-defined routers with customized data planes that allows custom protocol operations and/or any arbitrary payload processing at the network-layer. |
| Strengths | Lower life cycle cost, increased interoperability with multiple waveforms, field upgradable, demonstrates hardware flexibility. | They inherit the strengths of SDR (being SDR-based typically). They can learn about the environment and self-optimize by modifying radio parameters accordingly to ensure certain QoS. Hardware and policy flexibility. | Provides some capabilities of a full SDR on commodity WLAN hardware allowing support for arbitrary MAC protocols. This results in a cheaper solution that is also simpler. The main strength of this approach is programmability. | Flexibility in programming arbitrary processing in the forwarding plane. This can allow routers to diversifying beyond a forwarding only regime into the territory of *middleboxes* to define customized data plane processing of packets. |
| Weaknesses | Basic hardware of SDR is typically more expensive that single-mode hardware radio. | Numerous technical hurdles must still be overcome for CR to be ready for implementation in a real world scenario. A commercial fully functional CRN is yet to emerge. | More work needs to be done to support more sophisticated abstractions widely on commodity devices. | Not deployed widely. May not match the processing ability of fixed routers with predefined data plane processing. |

OpenSig community actively worked on standardizing such an open interface, and a number of experimental networks set up in various research institutes explored these proposals. The Tempest framework [40], based on the OpenSig philosophy, allowed multiple control planes to simultaneously control a single network of ATM switches. The main reason of OpenSig approach could not quite become mainstream was the static nature of the interfaces it defined [14].

#### *3.1.2 Active networking*

The *Active Networking* (AN) approach [16,69] was popularized at the same time as OpenSig, i.e., in the mid 1990s, when the Internet was rapidly commercializing and experiencing the need of more flexible control. The AN approach aimed at creating programmable networks that can allow rapid network innovations. The AN research - driven mainly by the efforts of the Defense Advanced Research Projects Agency (DARPA) - was motivated by the need to rapidly commission new services and dynamically configure networks in run-time. It was perceived that the static nature of OpenSig networks could not support these needs.

The main idea of AN is to *actively* control network nodes so that the network nodes may be programmed to execute arbitrary mobile code as desired by the operator [16]. The value proposition of such an approach was that it would enable new innovative applications, that leverage computation with the network, and that it would increase the rate of innovations by decoupling services from the underlying infrastructure [71,72]. The flexibility offered by such an approach, on the other hand, was also accompanied by concerns about its performance and security implications.

The AN approach consisted of two programming models: *i*) the *capsule model*, where the data packets contained not only the data to be communicated but also in-band instructions to execute; and the *ii*) *programmable switch model*, in which the out-of-band mechanisms were utilized to execute code at various nodes [18,69]. While it is the capsule model - which was the more radical approach, significantly different from the traditional operational paradigm of networking - that is most closely associated with AN, it is fair to say that both these models have bequeathed valuable legacies inherited by modern programmable networking frameworks. In particular, the capsule approach attracted interest since it could provide a clean method of upgrading data plane processing along an entire network path [18,72]. Using the capsule approach, numerous services such as active load balancing, multicasting, caching, etc. could be supported [69,73].

The AN framework was vigorously pursued by the research community in the mid and late 1990s - helped by the interest and generous funding of DARPA. Various influential projects were initiated in this time frame with some prominent AN projects being the ActiveWare project [74] at MIT, the CANES [75] project at Georgia Tech, the SwitchWare project [76] at University of Pennsylvania, the ANTS project at University of Washington [77,78], and the Tempest project [40] at Cambridge University. More details about these, and other important AN projects, can be seen in Table 3 and in the survey paper [69].

The AN paradigm was the first in a series of *clean-slate Internet redesign* proposals [18]. Many of the programmable networking concepts that appear eminently modern - such as separation of control plane and data plane, remote control of data planes, virtualization, network APIs, etc. - have in fact germinated from the AN community. However, despite the ground breaking nature of the AN paradigm, the AN paradigm failed to really catch on. One reason for its failure to capitalize on the intense interest around was the lack of a compelling application use case in the AN approach that could work pragmatically within the existing framework. Secondly, AN emphasized the flexibility of providing network end users the chance to program the network which never became a popular use case.

### 3.2 Virtualization and cloud computing

Virtualization is a technique, fundamental to various disciplines of computer science, which allows *sharing* of resources while providing *abstractions* identical, for all practical purposes, to that of the original resource.

Virtualization has been especially influential in the modern era of large-scale datacenters. Prior to the popularization of virtualization technology, various concerns (such as security, isolation, performance) dictated that servers be dedicated for particular applications (e.g., dedicated web servers, database servers, etc.) and provisioned for peak load. This led to gross underutilization with 10% to 20% utilization of resources being commonplace. This has motivated the creation of a new 'virtual machine' (VM) abstraction using which multiple VM instances, completely isolated from each other, can be created on the same physical machine. These virtual machines provided an interface to end applications that was identical to that of the underlying physical server. With the programmability features of VM cloning and mobility, which allows taking VM snapshots and transporting to any physical server that is currently underutilized, physical resources can now be shared both efficiently and securely. Due to these desirable properties, virtualization has truly become an indispensable component of modern computing.

The popularity of compute virtualization in the datacenter environment has spawned two further trends: *i*) cloud computing and *ii*) network virtualization.

**Table 3 Representative summary of important programmable networking concepts**

| Project | Framework | Year | Summary |
|---|---|---|---|
| Active control of network nodes | | | |
| ANTS [77,78] | Active Networking | 1997 | Java-based active networking toolkit proposed as part of the MIT's ActiveWare [74] project. |
| SwitchWare [76] | Active Networking | 1998 | Active networking project at Uni. of Pennsylvania which focused on both security and performance issues. |
| CANEs [75] | Active Networking | 1998/9 | Composable Active Network Elements (CANEs) project at Georgia Tech. |
| Separation of the control Plane (CP) and the data plane (DP), and remote control of DP | | | |
| Tempest [40] | OpenSig | 1998 | Programmable framework for safe control of ATM switches. Allowed multiple control architectures to coexist on the same network, and a safe partitioned environment for third party, or dynamically loaded, active code. |
| GSMP [79] | Pre-SDN | 2002 | General Switch Management Protocol (GSMP) proposed by an IETF working group to control a label switch. |
| ForCES [80] | Pre-SDN | 2004 | It defines a standardized interface between a network's control elements (CEs) and forwarding elements (FEs). |
| RCP [81] | Pre-SDN | 2004 | This work proposed separating routing from routers and outsourcing it to a separate router control platform (RCP). |
| SoftRouter [82] | Pre-SDN | 2004 | SoftRouter proposed separation of the control plane functions from the packet forwarding functions. |
| 4D [83] | Pre-SDN | 2005 | 4D proposed an architecture with *decision*, *dissemination*, *discovery*, and *data* - i.e., the 4D - planes, to separate decision logic from distributed systems issues. |
| Routing as a service [84] | Pre-SDN | 2006 | Proposed the provision of offering customized route computation as a service by third party providers. |
| PCE architecture [85] | Pre-SDN | 2006 | Path-computation-element (PCE)-based architecture (RFC 4655) where the PCE is an application located within a network node, or on an out-of-network server. |
| CogNet [63] | Pre-SDN | 2006 | CogNet proposed separated CP and DP with an extensible global CP controlling the separated DPs through an API. |
| IRSCP [86] | Pre-SDN | 2006 | Proposed Intelligent Route Service Control Point (IRSCP) that allowed route selection to be performed outside the routers through external network intelligence. |
| SANE [87] | Pre-SDN | 2006 | SANE is a enterprise network security/ protection architecture implemented through a 'logically centralized' server. |
| Ethane [88] | Cusp of SDN-era | 2007 | Ethane proposed fine-grain control of simple flow-based Ethernet switches through a centralized controller. |
| OpenFlow [8] | SDN | 2008 | OpenFlow defines a southbound API/protocol standardized by ONF through which a separated dedicated controller can control multiple DPs remotely. |
| Open APIs | | | |
| xbind [39] | OpenSig | 1996 | Toolkit developed at Columbia Uni. for creating *broadband kernels* - that program broadband ATM nets like PCs. |
| Mobiware[41] | OpenSig | 1998 | Programming QoS-aware middleware for mobile multimedia networking developed at Columbia University. |
| NetScript | OpenSig | 1999 | Language for programmable processing of packet streams. |
| OpenFlow [8] | SDN | 2008 | Southbound API standardized by ONF. |
| Floodlight API [8] | SDN | 2012 | A RESTful northbound API between the controller platform and the SDN Applications. |
| Juniper APIs [89] | SDN | 2012 | JunOS SDK, XML API (NetConf), and Contrail REST API supported. |
| Cisco ONE [90] | SDN | 2012 | Network APIs (including southbound API) specified by Cisco; supports EEM (tcl), Python Scripting. |
| OpenStack APIs [91] | Cloud/ SDN | 2012 | OpenStack Neutron (formerly Quantum) is a OpenStack subsystem for managing networks in a cloud environment. |
| Network virtualization | | | |
| Virtual switches [39] | OpenSig | 1996 | Proposed virtualizing ATM switches as part of the xbind [39] project. |
| Switchlets [40] | OpenSig | 1998 | Proposed dynamically loadable code on a (partition of) ATM switches as part of the Tempest [40]. |
| Virtual base stations [41] | OpenSig | 1998 | Proposed as part of the Mobiware [41] project subscribing to the OpenSig framework. |
| Routelets [42] | OpenSig | 1999 | Proposed in the Genesis [42] project. |
| PlanetLab [92,93] | Overlays Networks | 2003 | Proposed overlays, virtualized slicing, and programmability for accelerating innovations in the Internet community. |
| FlowVisor [94] | SDN | 2009 | Virtualizes OpenFlow-based SDNs by carving out virtualized "*slices*" out of production networks [95]. |
| SecondNet [96] | SDN/ Datacenters | 2012 | Proposes a virtual datacenter (VDC) abstraction as the unit of resource allocation for multiple tenants in the cloud. |
| Network virtualization platform (NVP) [97] | SDN/ Datacenters | 2014 | Proposed a overlay-based network virtualization platform (NVP) for use in production multitenant datacenters. |

The main insight of *cloud computing* is to provide services in a virtualized datacenter, provisioned programmatically through APIs via the web, as a service in *utility computing* style [98]. Although utility computing was conceived as early as 1961 by John McCarthy [99], it is only recently that cloud computing has turned this vision into a reality. The cloud paradigm is differentiated from traditional datacenters mainly in the dynamism of service provisioning which has been made feasible by virtualization technology and advances in web APIs. The ability to *program* services has led to great innovations and has democratized computing largely by making computing resources available on per-use pricing. The 'holy grail' of the cloud computing paradigm is the vision of installing a generic 'network fabric' [100] which can be then programmed to provide any service without any need of manual configuration of network nodes. The implementation of such a fabric-based virtualized datacenter has proven itself elusive, due to the complexity of virtualizing networks, so much so that it is now a common sentiment in the networking industry that networking is the bottleneck in datacenter innovations[b]. With traditionally vertically integrated network devices, supporting cloud-era applications entails the undesirable burden of manually configuring various network switches through vendor-specific command-line-interfaces (CLIs) - a process that is cumbersome and error prone [101].

With the presence of 10s or 100s of VMs per machine, a software-based *hypervisor switch*, inside the physical server, takes care of inter-VM networking. A significant tipping point was recently witnessed when estimated number of physical ports were overtaken by virtualized ports - a significant inflection point in networking history indeed [102]. This has significant architectural implications. In particular, it has been highlighted that using an hypervisor overlay with a networking fabric constructed out of SDN technology (to be covered in Section 3.3) can become the functional equivalent of the traditionally influential *end-to-end principle* [103]. In addition, the virtualization/SDN hybrid architecture will also subsume the functionality of MPLS and middleboxes to offer a clean split between the core and the edge. In this new architecture, the SDN-based fabric will become the new core, while the hypervisor switches will be the new edge. We shall see later that these edge devices consist of hypervisor switches (e.g., Open vSwitch [104]) that are *software defined* and thus are *programmable* (using protocols such as OpenFlow [8]). This paradigm shift to software control fundamentally changes the pace of innovation and opens up a world of new possibilities.

While VMs have unshackled applications from being tied to particular physical servers, traditional *network virtualization* techniques (such as VLANs, VPNs, and overlay networks) do not offer an analogous*virtual network* (VN) abstraction that decouples the network from the physical infrastructure. This VN abstraction should, like the VM abstraction does for the server, ensure detachment of the virtual network from the physical infrastructure as well as isolation between multiple tenants sharing the same infrastructure, while providing the same interface as the original network. There was a notable early work on network virtualization in the OpenSig era: the Genesis project [42] proposed, in 1999, a *virtual network kernel* that was capable of *spawning* virtual network architectures on-demand. The term 'spawning' is used in Genesis as a metaphorical reference to the use of this term in the field of operating systems where it refers to the process of creating a new process that runs on the same hardware - analogously, spawning a network means creating a new network architecture on the same infrastructure. This concept, although important and novel, is distinct from the modern virtualization abstraction of a VN. Just like a VM is a software container - encapsulating logical CPU, memory, storage, networking, etc. - providing an interface identical to a physical machine to an application, a VN is a software container - encapsulating logical network components, such as routers, switches, firewalls, etc. - that presents an interface identical to a physical network to network applications. The VN abstraction for wireless networks is visually depicted in Figure 2. This abstraction allows great flexibility to IT managers as the physical network can now be managed as a 'fabric' offering some transport capacity that can be used, programmed, and repurposed as needed.

Virtualization is also a popular method in the Internet community for introducing innovations in production networks with minimal intervention through the use of *overlay networks* [92,93]. An overlay network is essentially a virtualized logical network built on top of a physical network with tunnels interconnecting edge devices. The overlay network is typically decoupled from the underlying physical network through dual address spaces representing a tunnel encapsulation with the virtual address space on the inside and the physical address space on the outside. The overlay network appears to the nodes connecting to it as a native network with the possibility of multiple overlays existing on the same underlying physical infrastructure (which allows support for multitenancy). Each overlay network is effectively a distinct logical network which can support service properties such as an arbitrary policy of L2, L3, and access control list (ACL) processing distinct from the physical network. This makes overlay networking a popular technique for supporting disruptive innovations in networks without requiring interventions in the core network [92,93].

Interested readers are referred to comprehensive surveys on network virtualization [105,106] and cloud computing [98] for more details.

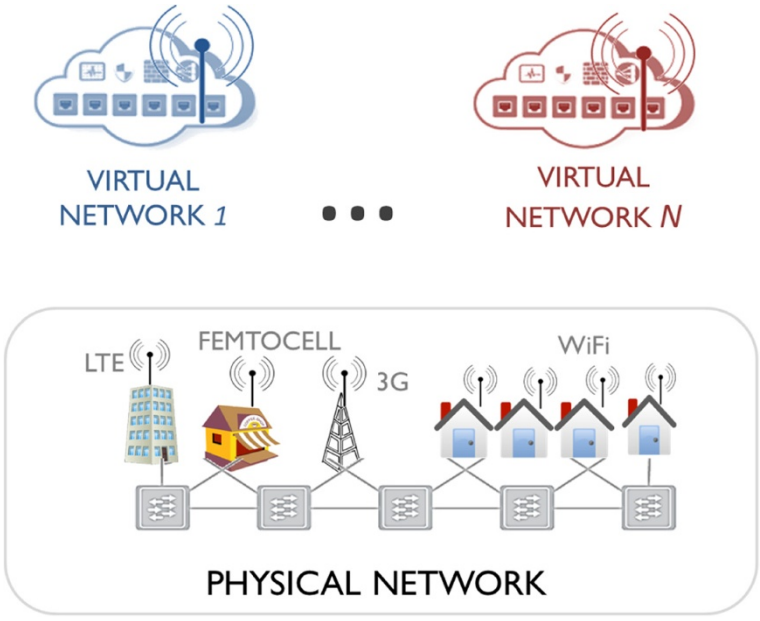


**Figure 2 Using virtualization, multiple virtual networks can coexist on the underlying physical infrastructure substrate in a decoupled fashion.**

### 3.3 Software-defined networking (SDN)

The major insight of 'software-defined networking' (SDNs) is to allow horizontally integrated systems by allowing the separation of the control plane and the data plane [11,107] while providing increasingly sophisticated set of abstractions. SDN has revolutionized the networking industry by providing architectural support for 'programming the network'. SDN promises to be a major paradigm shift in networking landscape leading to improved and simplified networking management and operations. While conceived mainly in academia, SDN has been taken up by the industry by gusto with numerous success stories [108,109]. SDN has also been seen recent successful industrial deployments [12]. Although SDN and active networking paradigms share a common motivation, i.e., of creating programmable networks, both of them are different in their focus; active networks strived more for data-plane programmability while SDN's focus has been on control-plane programmability [18].

Although the term SDN has only been coined in 2009, the idea of SDN has a rich intellectual history. In particular, it is the culmination of many varied ideas and proposals in the general field of programmable networks [11,18] with many of the initial ideas of programmable networking (of 'open interfaces' and 'separation of control and data plane') espoused by the OpenSig and Active Networking community now maturing in the form of the SDN) architecture. While the SDN architecture is very similar to the AN architecture, it has become more popular due to technological advances, compelling use cases, and importantly, certain pragmatic design choices. In particular, SDN has become popular largely due to the need of virtualization in modern datacenters and cloud computing which require network virtualization support due to their dependence on automated provisioning, automation, and orchestration. While AN focused on developing radically new data plane abstractions, the SDN approach has focused more on newer control plane abstractions (which arguably addresses a bigger pain point). The SDN architecture is different from the AN architecture since the former has emphasized on the separation of the control plane and the data plane [18] which was not integral to the AN architecture.

With the growing popularity of SDN, various industrial stakeholders have jumped on the SDN bandwagon to exploit its early success, and the term 'SDN' has seen a considerable broadening. To analyze and reason about SDN, it is, therefore, vital that we define it precisely. There are three key characteristics of SDN. Firstly, there is a separation of the data plane and the control plane. Secondly, a single control plane (or controller) may control multiple data planes (or the datapath of switches/router). Lastly, SDNs incorporate modularity in the control plane through which high-level abstractions can be used by network control programs. The distinction between traditional and SDN network architecture can be clearly observed in Figure 3b. To summarize these views, SDN deals with abstractions and mechanisms for creating a general, horizontal networking platform.

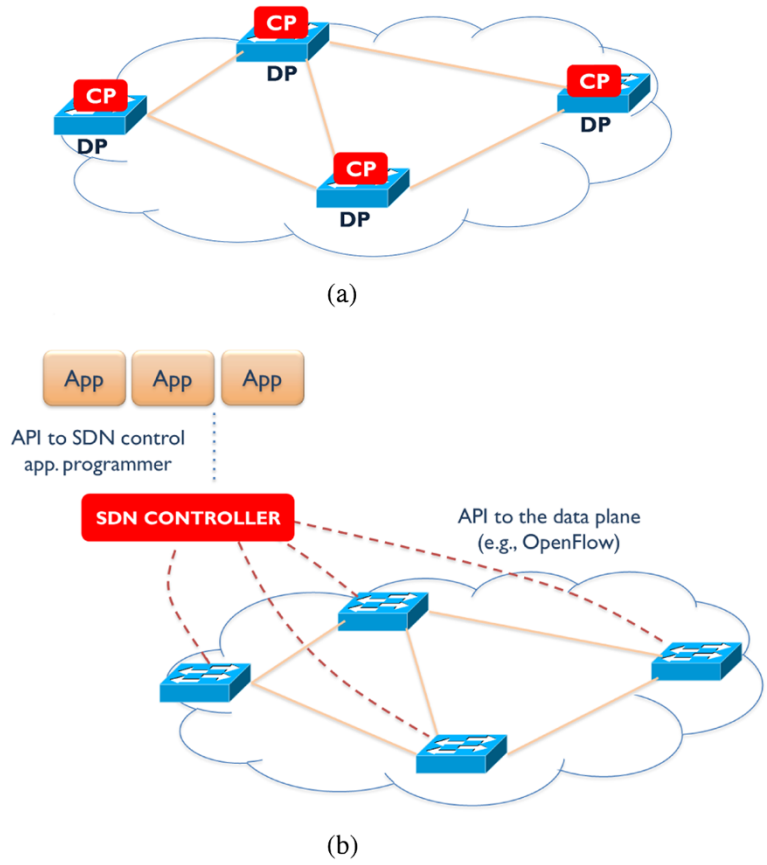


**Figure 3 Comparison of traditional and SDN network architectures. (a)** In traditional networking, the control planes (CP) and the data planes (DP) are colocated on devices to ensure decentralized network control. **(b)** In SDNs, the DPs and CPs are separated with a centralized controller controlling multiple DPs while supporting a *southbound* API to the DPs and a *northbound* API to the SDN applications.

By providing abstraction layers, it is possible to program new applications on central controllers for a wide variety of purposes. These *applications* include setting up virtual networks, enforcing quality of service (QoS), explicit routing, etc. The most fertile application area of SDN has been datacenters and campus networks [8]; however, SDN has also been proposed in many other settings such as service providers, carrier networks, and wireless networks. Specifically, SDN has been applied to wireless networks in varied settings such as wireless sensor networks (WSNs) and wireless mesh networks (WMNs) [110].

Although networks have always been software defined[c], writing, managing, and updating the networking software could be done only by the developers employed by the vendors. This 'closed' network architecture stifled innovation as the network was inherently non-programmable, and all new feature requests had to be routed to the networking vendor for implementation. SDN changes this paradigm by opening up the network through the simple, yet powerful, concept of separation of the control plane and the data plane. This separation, coupled with newer control abstractions, forms the core of the new SDN architecture. The development of SDNs is supported by a burgeoning open-source community. With the separation of the control plane from the data plane, it is possible for third party/open-source developers to write program applications for the controller. This allows networks to employ programmable commodity hardware rather than closed vendor hardware, increasing flexibility and development while reducing costs.

The Open Network Foundation (ONF) is an organization that is working on developing and maintaining standards for SDNs. Broadly speaking, there are two main APIs in the SDN architecture: *i*) the *southbound API* defines an interface between a centralized network controller[d] and networking devices [8], while *ii*) the *northbound API* defines the interface exposed by the controller to the network applications. There also has been activity in the Internet Engineering Task Force (IETF) community on standardizing SDN (e.g., with the I2RS [111] and ALTO [112] projects). In addition, there is the OpenDaylight[e] project under the Linux Foundation which

is a community-led and industry-supported framework which aims at fostering innovation in the SDN space through the best practices of open-source development best practices.

*OpenFlow* [8] represents perhaps the most readily recognized protocol associated with SDN[f]. OpenFlow is an example standard southbound API which has been standardized by the ONF. The standardization of OpenFlow has propelled it as the principal SDN control plane abstraction, enabling thereby numerous innovations [113]. With the control logic implemented in a separate controller and a standardized control API between the controller and the data planes, the vision of programming the network using a high-level control language can be achieved. While the current control API defined by OpenFlow is fairly primitive (and has been compared to Assembly language), it is a matter of time before higher level control languages are developed that offer more sophisticated abstractions. Indeed, work in this direction is already underway [17]. The seemingly innocuous refactoring of the functionality from individual devices to the centralized controller unleashes a powerful new paradigm offering abstractions which facilitate simplified, efficient, and scalable management of network operations and services.

The development of the SDN architecture has also led to the development of the 'network operating system' abstraction. The role of an operating system (OS) is to manage the complexity of various components, that a computer is composed of, and to present a simplified programming interface to the application programmer. In a similar vein, it is envisioned that the network OS (NOS) will manage the various tasks necessary to manage the network (such as exchange of distributed state and computation of routes, etc.) and present a simplified interface to network application programmers. The NOS is typically implemented at the SDN controller(s). An NOS is expected to implement a *state management layer*, managing distributed state in the network to provide a consistent network view, and provide an API to network applications to facilitate high-level programming. Various NOS have been implemented for SDN including the seminal work for NOX [114] and subsequent efforts for ONIX [115] and ONOS [116].

Since the proposal of the SDN architecture [8] in 2008, many research works have focused on the development of higher layer protocols and applications that can leverage and exploit the programmability offered by the SDN architecture. In particular, routing, transport layer, and management frameworks have been proposed that work with OpenFlow and SDN. The routing proposals include *i*) Quagflow [117], which partners the open-source routing software Quaqqa [118] with OpenFlow and *ii*) RouteFlow approach [119] which can be used to provide 'virtual routers as a service' in SDN environments [120]. The transport layer protocol proposals include the work on OpenTCP [121]. Finally, there has been work on supporting multimedia delivery with QoS with the OpenQoS [122].

The SDN architecture, as has been highlighted before, uses many of the programmability concepts of earlier projects. In particular, it builds upon earlier proposals to *i*) separate the control plane and the data plane (early proposals include SoftRouter [82], 4D [83], RCP [81], and work in the ForCES working group [80]), *ii)* control multiple data planes from a separate controller (like the Tempest framework [40]), and *iii*) utilize open interface for communications between the separate controller and the data planes (like the OpenSig framework [70]). A representative summary of various programming concepts, many of which SDN exploits, is shown in Table 3.

Despite the fact that SDN utilizes many of the active networking projects, SDN has become more popular than its predecessors due to the various technology pushes (e.g., advances in computing and networking technology) and application pulls (e.g., datacenter and cloud services, network virtualization, etc.) and greater industrial acceptance due to certain pragmatic design choices [18]. The long-term success of SDN would require innovations in new abstractions for the control and data plane balanced with a pragmatic strategy for its deployment.

## 4 Dominant trends in programmable wireless networking

In this section, we focus on prominent trends in wireless networking that have the potential to play a major part in creating future programmable wireless networks. In particular, we will discuss SWNs, CWNs, and VWNs, and CbWNs (Table 4). Our generalized treatment of wireless networking will subsume discussions on both technologies that have evolved from their telecom roots (such as 4G networks such as WiMAX and long-term evolution (LTE)) and also those that have predominantly data networking foundations (such as Wi-Fi).

### 4.1 Trend 1: software-defined wireless networks (SWNs)

With increasing deployment and diversification of wireless technology, managing wireless networks has become very challenging. SDN is a promising architecture that can be used for conveniently operating, controlling, and managing wireless networks. As discussed in Section 3.3, the defining characteristic of SDN is generally understood to be the separation of the control and data plane. The presence of programmable controllers enables us to call these networks 'software defined'.

The development of *programmable wireless networks* requires changes not only in the control plane but also in the data plane. In particular, the wireless data plane needs

**Table 4 Representative summary of important trends in wireless networking**

| Application/network settings | Representative references | Description/main idea |
|---|---|---|
| Trend 1: software-defined wireless networks (SWNs) | | |
| Applications | | |
| Programmable wireless data planes | OpenRadio [33]; OpenRoads [123] | Proposed wireless-specific programmable data planes supporting OpenFlow/ SDN |
| Real-time analytics/reconfiguration | Metsch et al. [124] | Proposed using real-time analytics for service operation and management in mobile networks |
| Distributed mobility management | [125]; [126]; DMM RFC [127] | Proposes using SDN concepts for routing optimization to support DMM |
| Efficient resource utilization | OpenFlow Wireless [128] | Allows infrastructural sharing using SDN principles |
| Mobile traffic offloading | SoftOffload [129] | Proposed an SDN-based programmable framework, SoftOffload, for mobile traffic offloading. |
| Traffic engineering | Traffic Engineering Survey [130] | Presents a comprehensive survey of the state of the art of TE in SDNs |
| Optimized management | OpenFlow Wireless [128] | Allows optimized management of diverse wireless technologies using the OpenFlow protocol |
| Heterogeneous technology handover | Yi et al. [131] | SDN can facilitate handovers between heterogeneous technology and across service providers |
| Service orchestration | Odin-based LVAP abstraction [132] | Programmatic Orchestration of Wi-Fi Networks |
| Security enhancement | Ding et al. [133] | Proposed an SDN-based framework for security enhancement in wireless mobile networks |
| Network settings | | |
| WLAN-based SWNs | Odin [132,134] | SDN benefits include flexible control, better management, rapid innovations, etc. |
| Cellular mobile SWNs | MobileFlow [135] | Proposed a software-defined mobile network (SDMN) architecture |
| | SoftRAN [136] | Proposed a SDN-based RAN architecture based on virtualization for LTE |
| | SoftCell [137] | Proposed a SDN-based flexible cellular core network architecture |
| WSN-based SWNs | Luo et al. [138] | Using SDN principles in WSNs to allow flexible and optimized resource utilization |
| LRPAN-based SWNs | Costanzo et al. [139] | Using SDN in LRPANs for flexible management along with efficient resource utilization |
| Trend 2: cognitive wireless networks (CWNs) | | |
| Applications | | |
| Cognitive networking | [64,140-142] | Allows optimization/ decision making from the perspective of the overall network |
| Adaptive routing | Routing survey papers [61,143] | AI-enabled routing techniques/ protocols for network-optimized routing are presented |
| Dynamic spectrum access | DSA Survey [144] | Allows a secondary network to coexist with incumbent users belonging to the primary net |
| Parameter optimization | Optimization survey [145] | Surveys self-organization paradigms and optimization approaches for CRNs |
| Optimized MAC | MAC Surveys [146,147] | Presents a comprehensive survey of optimizing MAC protocols for CRNs |
| Enhanced reliability | Reliability tutorial [148] | Presents a tutorial of how CRNs can improve reliability of wireless networks |
| Improved security | Security Survey [149] | Surveys security threats in CRNs and how they can be addressed |
| Network settings | | |
| IEEE 802.11-based CWNs | [150,151]; IEEE 802.11af [152] | These proposals address the issue of embedding cognition in IEEE 802.11 networks |
| IEEE 802.22-based CWNs | Cordeiro et al. [153] | Proposes a wireless regional area network (WRAN) to operate in TV-bands |
| Cognitive white space networks | Yuan et al. [37] | Proposes the use of white spaces in TV-band space for dynamic spectrum access |
| Cognitive sensor networks | Akan et al. [154] | Proposes a hybrid of CRNs and wireless sensor networks |
| Cognitive vehicular networks | Di et al. [155] | Proposes the use of cognitive technology to interconnections of vehicular systems |

**Table 4 Representative summary of important trends in wireless networking** *(Continued)*

| | | |
|---|---|---|
| Trend 3: virtualizable wireless networks (VWNs) | | |
| Applications | | |
| Multi-tenancy support | MobileFlow [135] | Proposed a virtualized SDN-based framework suitable for multi tenant mobile networks |
| Multi-provider support (infrastructure sharing for MVNOs) | Virtualization of 4G/ 5G RAN [156], WiMAX BS [157], LTE [158] | Virtualization allows better support for multi-tenancy and multi-provider and infrastructure sharing, which is convenient both in terms of user experience and economics |
| Virtualized NIC abstraction | Commodity WLAN card [159] | TDM-based wireless virtualization to create a virtual WLAN using commodity hardware |
| Virtual-APs | Hamaguchi et al. [160] | Virtual AP that uses virtualization technology to optimize deployment of AP |
| Network settings | | |
| WLAN-based VWNs | Commodity WLAN card VWN [159] | Virtualization of commodity WLAN technology |
| | Virtual Wi-Fi [161] | Virtual Wi-Fi to support fully functional wireless functionality inside VMs |
| | multi-purpose AP (MPAP) [162] | Proposed MPAP for virtualizing heterogeneous technologies on a SDR |
| SDN-based VWNs | LVAP (based on Odin) [132] | Proposed Odin, based on SDN, to allow orchestration of programmable WLANs |
| | OpenAPI [163] | Proposed virtualizing the access network via Open APIs |
| | Virtual router as a service [120] | Proposed virtual-routers-as-a-service based on the RouteFlow architecture [119] |
| | eNodeB virtualization [164] | Proposed using OpenFlow for eNodeB virtualization in 4G-LTE networks |
| Cellular mobile VWNs | Virtualization of RAN [156] | Proposed network virtualization substrate (NVS) to be used in LTE RANs |
| | WiMAX BS [157] | Proposed virtualizing resources in a cellular WiMAX base station to enable MVNOs |
| | LTE [158]; eNodeB virt. [164] | Proposed virtualization of LTE environments |
| CRN-based VWNs | Spectrum Virtualization Layer [165] | This work proposed a virtualized layer for supporting DSA in general wireless networks |
| Trend 4: cloud-based wireless networks (CbWNs) | | |
| Applications | | |
| Computation offloading | Yang et al. [166]; CloneCloud [167] | Utilized computation offloading through elastic execution between mobile device and cloud |
| Centralized (remote) management | RFC 5412 [168]; Dalvi et al. [169] | Proposed centralized cloud-based approaches for managing WLAN |
| Real-time reconfiguration | Misra et al. [170] | QoS-guaranteed bandwidth redistribution among gateways in mobile cloud computing |
| Mobile cloud computing | Survey [171] | A comprehensive survey of mobile cloud computing technology and applications |
| | Wireless network as a service [172] | Investigates the pragmatism of having wireless networking as a service |
| | Wireless network cloud [173] | Proposed wireless network cloud (WNC) to operate a wireless access network in cloud mode |
| | RAN as a service [174] | Proposed an architecture for offering cloud-based RAN-as-a-service |
| Cloud-based virtualized WNs | CloudMAC [175] | Proposed an OF-based architecture for cloud processing of 802.11 MAC |
| Network settings | | |
| Cloud-based cognitive WNs | TV white space and clouds [176] | CWNs can perform increasingly complex tasks by offloading these computations to the cloud |
| Cloud-based cellular WNs | Cloud-based 5G RAN [177] | Proposed using cloud technologies for flexible 5G radio access networks |
| | Cloud-based LTE [178] | Investigated how cloud computing can be applied to LTE cellular systems. Also, evaluated OpenStack, Eucalyptus, and OpenNebula for this task |

to be redesigned to define new, more useful, abstractions. To put things into perspective, the current data plane abstraction offered by OpenFlow supporting switches is based on primitive *match-action* paradigm. To lend greater support to innovations in control plane functionality, the data plane functionality has to evolve to support more sophisticated, and useful, abstractions. Research on newer data plane abstractions is being vigorously pursued with the use of programmable hardware being popularly proposed [179]. The vision of programmable wireless networks thus requires synergy in multiple related domains and would require innovations in both the data plane and the control plane of wireless networks. Using SDN technology for wireless networking will extend the benefits of SDNs - simplification, flexibility, evolvability, and rapid innovation - to wireless environments [139].

In this section, we will initially discuss applications of SWNs in Section 4.1.1. We will then discuss architectures for providing software programmable wireless data planes (Section 4.1.2) as well as how SDN techniques have been applied in various settings of wireless networking (Section 4.1.3). While the finer details are application dependent, all SWNs seek to *i*) attempt to make management of networks a lot more easier, *ii*) allow the same hardware to support multiple wireless protocols, and *iii*) provide an abstraction layer to allow all, or some part, of the wireless architecture to be programmable. These aims are facilitated through the separation of the control and data planes which allows a separate controller to programmatically reconfigure network properties. The theme of providing abstractions for programmability thus pervades the SWN approaches we have discussed in this section.

#### *4.1.1 Applications of SWNs*

The biggest promise of SDN is to simplify and improve network management and operations [180]. We discuss some main applications of SDNs in this regard next.

- *Coexistence of diverse technologies.* SDN can be used to integrate diverse wireless technologies and facilitate optimized management and coexistence of diverse wireless technologies. For example, Yap et al. [128] have proposed supporting diverse wireless technologies using the OpenFlow protocol in the *OpenFlow Wireless* project. SDN can also be used to coexistence of diverse technology through the support of *mobile offloading*, or Wi-Fi roaming, using which traffic from a mobile cellular network is outsourced to a Wi-Fi data network to improve the quality-of-experience (QoE) of the end user.
- *Mobile offloading.* SDN can also facilitate handovers between heterogeneous technologies and across service provider. There has been work in using OpenFlow-based controller in managing mobile offloading applications in mobile networks such as 3G or 4G/LTE. There are currently various roaming solutions which facilitate homogeneous Wi-Fi handover (such as IEEE 802.11f, 802.11k, and 802.11r) as well as heterogeneous Wi-Fi handover such as IEEE 802.21. Yi et al. [131] have proposed utilizing SDN techniques to implement heterogeneous technology handover. In another recent work, Ding et al. have proposed SoftOffload [129] as a SDN-based programmable framework that enables mobile traffic offloading.
- *Efficient resource utilization.* We have already pointed out that OpenFlow Wireless [128] can be used to support diverse wireless technologies. In addition, OpenFlow Wireless, among various other SDN-based works, can facilitate more efficient resource utilization and better *infrastructural sharing*. The trend of VWNs, which we will discuss later in Section 4.3, can complement SDN in facilitating better resource utilization.
- *Better QoS/QoE.* The separation of the control and data planes and the usage of a centralized controller can facilitate better provisioning of network-wide QoS/QoE policies. As a particular example, Sivaraman et al. [163] have proposed to virtualize the access infrastructure of internet service providers (ISPs) and home network using SDN APIs to enable better QoS/QoE through dynamic controlled sharing among user traffic streams.
- *Traffic engineering or adaptive routing.* SDN can be used to dramatically improve the network utilization through better traffic engineering (with Google's traffic engineering work B4 [12] being a prime example on how traffic engineering can lead to vastly improved performance). The SDN architecture can facilitate efficient traffic engineering and/or adaptive routing in wireless and mobile networks [130].
- *Distributed mobility management (DMM).* DMM is a an architectural framework for evolving mobile IP services from the currently deployed mobile core networks (which have serious scalability/reliability issues due to their hierarchical centralized architecture) to a more scalable model with distributed operations [127,181]. Current IP mobility frameworks force a mobile network operator (MNO) to deploy central entities (like the home agent, local mobility anchor, packet gateway, etc.) in charge of coordinating the mobility management. The current architecture suffers from the problems of suboptimal routing, lack of scalability, and lack of reliability. SDN-based solutions can be used to implement a partially distributed model in which the control plane and data planes are managed separated. The split of control/data plane using SDN

principle allows mobility anchors to optimally route traffic [181].

- *Programmatic network service orchestration.* There is also the growing trend of using SDN for performing dynamic programmatic service orchestration such as proposed in the recent work proposed by Schulz-Zander et al. [132].
- *Real-time analytics/reconfiguration.* There is renewed interest in the Internet community in deploying the centralized SDN architecture for optimized reconfiguration of network infrastructure based on real-time analytics performed at the SDN controller(s). In a recent work, Metsch et al. [124] have proposed using real-time analytics for service operation and management in mobile networks.
- *Security enhancement.* The clean separation of data and control planes in the SDN architecture and the consolidation of the control functionality at the centralized controller allows improved implementation of enterprise-level security policy. In this regard, Ding et al. [133] have proposed an SDN-based framework for security enhancement in wireless mobile networks. Some applications of SDN techniques in mobile networks for improving security include *i*) implementation of 'intrusion detection systems' (IDS), *ii*) prevent denial of service (DOS) attacks near the wireless edge, and *iii*) implement secure handoffs in mobile networks [133].

#### *4.1.2 Software-defined programmable wireless data planes*

In this section, we will discuss two prominent architectures of software-defined programmable wireless data planes named OpenRadio and OpenRoads, respectively.

**OpenRadio** The *OpenRadio* system [33] defines a novel design of a wireless data plane that allows programming of the entire wireless stack through a modular and declarative programming interface. OpenRadio proposes to refactor the functionality of wireless protocols into two parts. The *processing plane* deals with programs and algorithms that process data using the underlying hardware. The *decision plane*, on the other hand, is responsible for making logical decisions on the data being processed by the processing planes. It should be observed that the concepts of the processing and decision planes are subtly analogous to that of the data and control planes in the SDN world, respectively. OpenRadio is themed in the mold of both SDRs and SDNs. OpenRadio uses an abstraction layer for managing wireless protocols on generic hardware configured through software like SDRs, while also allowing the separation of protocol from hardware similar to SDNs. OpenRadio can support different wireless protocols, like Wi-Fi, WiMAX, and LTE, etc. though a common hardware, thereby significantly reducing costs and making it easier to configure, optimize, and even define protocols. OpenRadio's major strength is its ability to detach protocol from hardware and to bind the former with software to allow increased flexibility. With newer wireless protocols regularly being rolled out, the ability to reprogram functionality centrally and programmatically is of great convenience. OpenRadio can also be used for cell-size based optimization in cellular networks and for management of frequency spectrum in the presence of multiple heterogeneous cell stations to avoid interference [33].

**OpenRoads or 'OpenFlow Wireless'** A seminal development in the field of programmable SWNs has been the *OpenRoads* project [123] - known also as *OpenFlow Wireless* [128]. OpenRoads provides a complete platform that can be used to apply SDN principles in wireless environments and thereby create a programmable wireless data plane. One particularly appealing benefit of OpenRoads is that it allows efficient handover between diverse wireless technologies, by 'flattening' multiple vertically integrated wireless technologies, to allow seamless mobility for clients of mobile wireless networks. In [128], the feasibility of this SDN-based approach is explored for mobility management with vertical handovers between IEEE 802.11 and IEEE 802.16 networks. OpenRoads employs OpenFlow and the Simple Network Management Protocol (SNMP) on wireless routers. OpenFlow provides means to manage the forwarding plane while SNMP allows configuration of these wireless devices. FlowVisor and SNMP demultiplexer are used to divide and make the control more scalable. Each controlling flow is given a particular 'flow value' to ensure that different controlling flows are isolated from one another so that only those flows that are intended for particular devices would be installed. High-level control interfaces are used upon OpenFlow to provide communication between different devices, configuration of these wireless devices, and management of flow.

#### *4.1.3 Network-specific categorization of SWNs*

We will now detail different wireless networking projects that have incorporated SDN principles. These projects vary in the manner in which they employ SDN principles as well as in the nature of wireless networks (wireless local area networks (WLANs), cellular networks, sensor networks, and personal area networks).

**WLAN-based SWNs** *Odin* [134] is a proposed SWN architecture that employs the principles of SDNs in WLANs. In its popular form, WLAN decisions are made by clients and not the WLAN infrastructure itself. For example, a client decides which access point it prefers to join rather than the infrastructure deciding it for the

client. In WLANs, association of clients with specific access points keeps on changing with client mobility. This poses a significant challenge to any potential SDN-oriented WLAN architecture as it would be difficult for controller programmers to keep track of the ever changing association between access points and clients. The Odin architecture suggests the usage of light virtual access points (LVAPs). LVAPs virtualize access point-client association and decouples it from physical access points. Whenever a client connects to the WLAN network, it is allotted an identification number on its LVAP that remains fixed regardless of its associated physical access point. The complexities of the physical access point are thus hidden from central controller programmers. The Odin program offers many *advantages*. Odin provides seamless mobility between access points as the need to constantly establish new connections with physical access points changes. Additionally, flexible routing policies further allow load balancing. Furthermore, with an improved overview of the network, it is possible to reduce interference and eliminate issues such as hidden node problems.

**Cellular mobile SWNs** Recently, there has been significant interest in improving the performance of cellular mobile networks through SDN principles. In particular, frameworks have been proposed that incorporate SDN principles into 3GPP evolved packet core (EPC) mobile carrier networks (the MobileFlow project [135]) and 4G LTE cellular networks (the SoftRAN project [136] and the SoftCell project [137]). The main *advantages* of the cellular mobile SWN approach include better management of radio resources, more flexible routing, real-time monitoring, better mobility support, and the ability to offload data to Wi-Fi networks [182,183].

There are a few problems with the current LTE architecture: *i*) centralized data flow as all the data passes through the packet gateway (P-GW), *ii*) centralized monitoring and control is not scalable and is expensive, and *iii*) base stations and infrastructure are difficult to configure. The first two problems are related with the central control and monitoring of LTE networks. Thus, a possible solution would need to distribute some of the control and monitoring responsibilities leading to a hybridized control plane. This seems to be a departure from one of the fundamental principles of SDNs, i.e., centralized control. Solution to the third problem lies in adapting an SDN-based architecture so that remote applications may be used for the tasks. As discussed earlier, OpenRadio provides an ideal modular interface to configure base stations remotely and conveniently.

**Wireless sensor network (WSN)-based SWNs** WSNs have been popular within the research community but have always being considered as an application-specific technology. Treating WSNs as application-specific technology leads to the problem of resource underutilization with potentially multiple application-specific WSNs being deployed over a shared area where a single WSN may have sufficed. Incorporating SDN in sensor networks would provide solution to these problems [138]. Separation of control and data planes would provide abstraction, helping to manage and control the network. By employing sensor network-based SWN, network controllers could set policies and quality of services to support multiple potential applications. This would also allow usage of the same sensor nodes for multiple application/purposes. This again increases resource utilization and optimization.

**Low-rate personal area network (LR-PAN)-based SWNs** SDN attributes can also be used to great advantage in LR-PANs [139]. All LR-PANs essentially employ the same 802.15 data link layer [184,185]. Differences in higher layers of their respective protocol stacks lead to different LR-PAN protocols such as ZigBee, Bluetooth, etc. This leads to incompatibilities in communication between different nodes. By using the same tools that are used in OpenRadio, we could separate hardware from protocol and use an abstraction layer to program and define different wireless protocols. This would allow us to run different wireless LR-PAN protocols on the same wireless device. In this way, it would be possible for nodes to be dynamically associated with many networks at a time, allowing us to use network resources more efficiently. The separation of data and control planes extends the usual SDN benefits of simpler management, flexible control, and efficient resource utilization to LR-PAN SWNs.

### 4.2 Trend 2: cognitive wireless networks (CWNs)

It has been highlighted earlier that the predominant focus of most of the existing CRN research has been on enabling intelligent device-level behavior, with a notable exception being some work on *cognitive networks* [63,64,140-142]. Cognitive networks, in contrast to cognitive radios, are characterized by their *network-level* intelligent and self-aware behavior. In this paper, we refer to such cognitive networks as 'cognitive wireless networks' (CWNs). CWNs employ a cognition loop (as can be seen in Figure 4) to observe the environment, orient itself, and thereafter decide/plan to arrive at the best decision according to the networking/user and application context.

While device-level reconfiguration capabilities (e.g., SDRs and CRs) and network-level reconfiguration capabilities (e.g., SDN) will undoubtedly be a big part of future programmable wireless networking, the resulting programmable wireless architecture will still not be fully automated unless AI techniques are incorporated

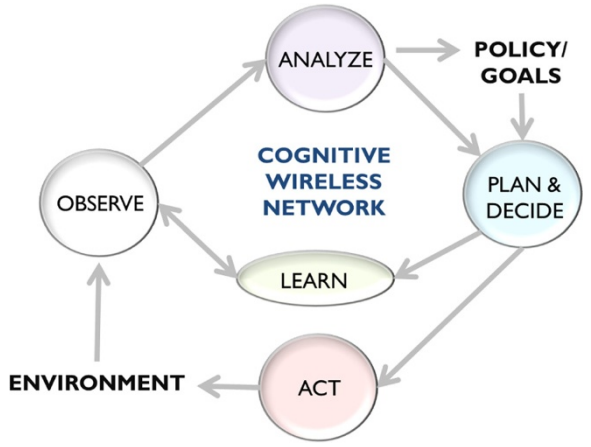


**Figure 4 Cognitive wireless networks (CWNs) which include an embedded cognitive engine.** Which can observe network conditions, orient itself with the context, learn from experience, and decide to act, all while taking into account end-to-end network goals.

into the core of the framework. In addition to programmable data plane and programmable control plane, both offering various useful abstractions, future wireless programmable networking also requires a *knowledge plane* [186]. Since CRNs inherently embody AI techniques with wireless communications, it seems natural to explore using CRs, along with the capabilities of SDN and SDRs, and to provide mechanisms for implementing the knowledge plane of future programmable wireless devices.

In future work, the hybrid use of SDN and CRN technology could plausibly lead to a more powerful programmable wireless networking paradigm. While the CogNet project [63] did propose an architectural model of separated control and data planes with an extensible global control plane controlling the separated data planes through an API - which is similar in spirit to the SDN architecture - no concrete proposal has followed this initial conception. This area is ripe for further exploration to exploit the best of CRN and SDN worlds.

We note here that CWNs are *autonomously self-programmed networks*, i.e., CWNs incorporate the ability of autonomously adapting, or *programming*, itself so that operation parameters are optimized to fulfill the desired goals of performance. This conception of programmability in CWNs is distinct from the traditional view of programmability (which also applied to SWNs) which implicitly assumes non-autonomous programming. Future programmable wireless networking will arguably employ both autonomous and non-autonomous programming to reap the benefits of both approaches.

In the following subsections, we will introduce applications of CWNs and will provide a network-specific categorization of CWNs.

#### 4.2.1 Applications of CWNs

- *Dynamic spectrum access* [144]. An important adaptive feature of CWNs is DSA which allows reconfiguration of operating frequency of a SU to allow communication in licensed spectrum. This depends critically on spectrum sensing (which is performed to detect the presence of primary users, or PUs) which is used to ensure that incumbent licensed users, represented by the PUs, are not interfered with. In certain cases, spectrum sensing can be avoided and a database lookup specifying the activity pattern of the PU suffices [144]. Many IEEE standards (such as IEEE 802.11, 802.15, and 802.16) incorporate some basic cognitive radio functionality such as dynamic frequency selection (DFS) and power control (PC) which facilitate *coexisting networks* sharing the same frequency [187]. While DSA is the most popularly cited application of CRNs, developing network-level intelligence in CRNs will enable numerous other applications - including the ability to reprogram itself optimally according to the network conditions.
- *Cognitive networking.* Cognitive networking broadly encompasses models of cognition and learning that have been defined for CRs while emphasizing an end-to-end network-wide scope. Such cognitive networks can perceive current conditions to plan, decide, and act while catering to the *overall network*'s end-to-end goals [64,142]. Figure 4 serves to illustrate the vision of cognitive networks. To help CRNs become *cognitive networks* (CN), it is imperative that intelligence be integrated into the fabric of CRN architecture and protocols across the stack. The cognitive networking vision foresees an intelligent network capable of setting itself up given high-level instructions and which can continually adapt and manage itself according to changing environmental conditions to optimize network-wide performance metrics.
  The grand vision of cognitive networking can enable various interesting applications. In particular, the applications of *network-optimized MAC protocols* [146,147], *adaptive routing* [61,143], and *improved security* [149] hold special promise and can enable significant advancement in the state of the art in wireless networking services.
- *Parameter optimization.* In typical wireless networks, it has been observed that PHY and MAC layers offer many *knobs* that can be tweaked to optimize performance which may be measured through some *meters*. In [60], many examples of knobs and meters at the PHY and MAC layers have been provided in the context of CWNs. Since wireless networks often operate in dynamic conditions, configuring the knobs optimally is not a trivial problem. Various AI-based

techniques have been proposed in the literature to assist CWNs in their quest of performing autonomous parameter optimization adaptations in such settings [145]. Apart from artificial intelligence, CWNs also borrow techniques and tools from various other fields such as game theory, control theory, optimization theory, metaheuristics, etc. [59].

- *Enhanced reliability*. To be widely deployed, it is important that next-generation wireless networks provide enhanced reliability independently of the state of the network and the radio transmission medium. Unfortunately, the wireless transmission medium is particularly error prone. CRs have been proposed as a solution to this problem, and they can combat various failures through efficient failure recovery and prevention mechanisms. This can allow CWNs to provide consistent QoS under all circumstances. A detailed survey of various failure conditions and how CRs can be used to provide enhanced reliability in the face of such failures is provided in [148].
- *Adaptive optimization*. CWNs offer the ability to adaptive optimize against the varying wireless environmental conditions using AI-based techniques. CWN typically utilizes adaptive frameworks such as reinforcement learning, learning automata, game theory, etc., for adaptively optimizing the parameters of the network. There are numerous optimization-based applications of CWNs including dynamic spectrum access [144], parameter optimization [145,188], optimized MAC [146] and routing [61,189], enhanced reliability [148] and security [149,189], QoS assurance and management [190,191], channel assignment [192], etc.

#### *4.2.2 Network-specific categorization of CWNs*

The use of CR technology has been proposed in various network settings. This allows the CWN trend to manifest itself in different network settings providing in each case the benefits of cognitive networking. There has been work in applying cognitive technology to *IEEE 802.11 networks*. Niyato et al. have proposed using CRs for opportunistic channel selection in IEEE 802.11-based WMNs. There also has been work in implementing white space networking with commodity IEEE 802.11 WLAN NICs [151]. CRs have also been used in the IEEE 802.22 wireless regional area network (WRAN) standard which is proposed to be used in the TV bands. Other white space networking projects include the work of Yuan et al. [37] which has focused on using white spaces in the TV band for DSA. There also has been working in embedding CR technology in WSNs and vehicular networks (VANETs) to help realize *cognitive sensor networks* [154] and *cognitive vehicular networks*, respectively.

### 4.3 Trend 3: virtualizable wireless networks (VWNs)

Virtualization has transformed both the operational efficiency and the economics of the compute industry and more recently, the datacenter environment. With the growing role of virtualization in networking, it is highly likely that future programmable wireless networks will be virtualization based.

There is also growing interest among major network service providers to decouple the functionality of telecom devices from dedicated devices to enable 'network functions virtualization' (NFV) (as can be seen in Figure 5) which will enable implementation of network functions (such as mobile network node, etc.) on servers in datacenters [193]. The concept of NFV extends to any data plane packet processing and control plane function in mobile or fixed networks including, but not limited to, mobile network nodes and traditional switching devices such as routers, switches, home gateways, etc. NFV technology also allows the providers to make the data plane programmable which will facilitate in orchestrating middlebox functionality efficiently.

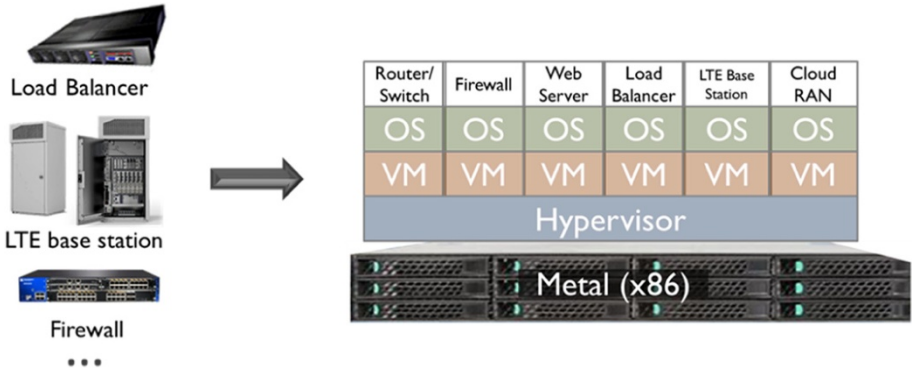


**Figure 5 NFV is used to convert fixed-function hardware appliances into virtualized cloud software instances.** That run on commodity infrastructure hosted in cloud datacenters.

Traditionally, it is not uncommon for a single packet to undergo processing in the data plane through multiple middleboxes that are used to augment data plane processing by L2 and L3 switches. This is in contrast to the traditional mainframe like model of building middleboxes based on monolithic platforms housed in fixed location in the network (typically at the edge of carrier's core network). This traditional architecture suffered from being rigid, static, and resistant to attempts at automation and orchestration. The functionality of network infrastructure orchestration can be implemented in the NFV framework, on the other hand, by dynamic 'network service chaining' (NSC) [194] through virtual network functions (such as ACLs, QoS, load balancing, etc.) running as virtualized cloud instances in datacenters.

There has been a lot of work done in exploiting SDN principles to enable network virtualization solutions. An initial SDN use case, espoused in [8], was allowing researchers to run experimental protocols in virtualized 'slices' of the production network. While the concept of slicing network through virtualization technology predates SDN - e.g., it has been used in VINI [195], PlanetLab [92], Emulab [196], and more recently in the NSF funded Global Environment for Network Innovations (GENI) project [197] - Casado et al. have extended this idea further by proposing a *network hypervisor* to virtualize the network's forwarding plane [198]. The concept of a network hypervisor is analogous to the traditional hypervisor concept. The network hypervisor implements a network-wide software layer through which it is aimed that multiple virtualized networks, that are decoupled from their underlying hardware instantiation, can be supported. In such an environment, the network state (forwarding and configuration) is decoupled from the underlying hardware, and thus, networks can be created, moved, cloned, and deleted just like VMs in the server world - all in software. A network hypervisor, FlowVisor [94], has been developed for OpenFlow-based SDN environments that allows carving out of virtualized *slices*, that are isolated from each other and controlled by a separate NOS, out of OpenFlow production networks [95].

There also has been work in exploiting SDN in addition to NFV to bulid service chains. OpenFlow has also been used for designing and prototyping high-speed networking by a reusable platform *OpenPipes* [199]. Using an OpenFlow network, new systems can be constructed quickly by OpenPipes, like the Click modular router, by 'plumbing' modules - be they implemented in CPU, FPGA, and ASIC - together in a pipeline. OpenPipes also allows flexible migration of modules (implemented in software or hardware, or both) from one subsystem to another, even in a running system.

It is pertinent here to clarify the connections between SDN and network virtualization (NV). Since both these technologies return some similar benefits, it is a common mistake to equate these two technologies [18]. The SDN architecture is characterized by its emphasis on the separation of the control plane and the data planes and the potential management of multiple data planes through the separated control plane. NV, on the other hand, is characterized by its emphasis on a new 'virtual network' (VN) abstraction that decouples the virtual network from the physical infrastructure. It is another myth that NV is just an application of SDN. It is worth stressing that NV is a *solution* while SDN is an *architecture* - while NV can be implemented more easily using the architectural flexibility offered by SDN, implementation of SDN is not a prerequisite for NV. It has been argued quite convincingly that NV is a distinct entity, important in its own right [18,103,200], which may turn out to be even bigger than the current SDN fad sweeping the networking community [103].

In summary, the cloud computing concepts, proposed originally for datacenters, are likely to play a big part in creating future programmable *wireless* networks. In a few years time, it is anticipated that mobile carrier and telecom networks will increasingly emulate datacenters and clouds in their reliance on commodity hardware, virtualization technology, and open software and interfaces in a break from the current scenario in which telecom service providers are full of proprietary vertically integrated hardware appliances. In particular, the combination of cloud computing/network virtualization (including network functions virtualization) allow programmability that leads to unprecedented flexibility in rapidly creating, deploying, and managing novel services in virtualized settings as per the demands of users. This can create a new service-oriented architecture for wireless networking where heterogeneous wireless access technologies may coexist and converge as *extended cloud infrastructure* [201]. For a more comprehensive discussion of wireless virtualization, the interested reader is referred to a specialized book on this topic [201].

In the remainder of this section, we will discuss applications of VWNs and will provide a network-specific characterization of VWN applications.

#### 4.3.1 Applications of VWNs

- *Convergence of different technologies.* Wireless technologies are proliferating at a breakneck pace, and in such a dynamic ecosystem, technologies that facilitate multi-technology convergence will become increasingly important. VWNs are an important multi-faceted trend that promise to facilitate coexistence of diverse technologies.
- *Efficient resource sharing.* An important application pull and use-case of VWNs in general is the convenience of supporting multiple virtual owners (also known as *tenants*) on shared infrastructure.

The trend of VWNs provides better support for multi-tenancy and multi-provider and infrastructure sharing, which is convenient both in terms of user experience and economics [202]. Each tenant is provided with a 'slice' - which is an virtualized abstraction of shared wireless infrastructural components - with a committed service-level agreement (SLA) to allow efficient resource sharing of expensive infrastructure. For instance, in the service provider domain, there is a need of supporting multiple virtual network operators (VNOs) - mobile VNOs (MVNOs) in the context of mobile carrier networks [203] - on shared infrastructure to lower the capital and operational expenditures (CAPEX and OPEX). The trend of VWNs can also be used with other trends such as SWNs to facilitate multi-tenancy as has been proposed in the MobileFlow framework [135] to build multi-tenant mobile networks.

- *Virtualization of network components.* There has been a lot of work done in virtualizing various aspects of wireless network components. In practice, the trend of VWNs encompasses many sub-solutions offering varying level of programmability and granularity of control. For example, VWNs encompass in the context of cellular networks, the virtualization of 4G/5G RAN [156], WiMAX BS [157], and LTE [158]; in a similar vein, virtualized abstractions have been proposed for WLAN environments such as the virtual NIC abstraction [159] and the virtual access point (AP) abstraction [160]. As we shall see later in section, virtualization has been applied in the context of WLANs, cellular networks, etc.

#### 4.3.2 Network-specific categorization of VWNs

In the following, we shall discuss the application of virtualization in four environments: *i*) WLANs, *ii*) software-defined VWNs, *iii*) cellular mobile carrier networks, and *iv*) CRNs.

**WLAN-based VWNs** With the widespread use of IEEE 802.11 WLANs (Wi-Fi) - and the pervasive commissioning of Wi-Fi hotspots in campuses, offices, business centers, airports, shopping centers, etc. - the wireless signal is almost ubiquitously available. There has been a lot of interest in exploiting this common infrastructure to support multi-tenant and multi-provider environments. The concept of 'slices' proposes to provide virtualized environment that runs on top of common shared infrastructure. The general area of network virtualization (NV) is explored in depth in [106], and the interested reader is referred to this paper, and the references therein, for more details. Some prominent contributions that have proposed virtualization for WLANs include wireless virtualization on commodity 802.11 hardware [159], the use of virtual access points (VAPs) [132,160], virtual Wi-Fi [161], and building multi-purpose access point (MPAP) virtualization architecture [162].

**Software-defined VWNs** There is a lot of interest in exploiting SDN technologies to enhance the functionality and utility of VWNs. With management protocols such as control and provisioning of wireless access points (CAPWAP) [204], it will be possible to implement infrastructure-wide virtualization using functions consolidated in a centralized software controller. In an SDN context, the FlowVisor [94] project has proposed mechanisms for ensuring isolation needed to support virtualization. In OpenRoads [123], OpenFlow has been extended to wireless APs with forwarding planes virtualized through FlowVisor. In [163], Sivaraman et al. have proposed to address the impasse on service quality in access networks by virtualizing the wireless access network via open APIs using SDN techniques. In a recent work, Schulz-Zander et al. [132] have proposed using SDN techniques using their Odin work for programmatic orchestration of Wi-Fi networks.

**Cellular mobile VWNs** With the mobile traffic increasing exponentially [1], mobile carrier wireless networks are an attractive setting for wireless virtualization. Various works, focusing on mobile carrier VWNs, have exploited virtualization technology, with some sample works being Costa et al. work on RAN virtualization [156], Bhanage et al. work on WiMAX base station virtualization [157], and Zaki et al. work on LTE virtualization [158].

**CRN-based VWNs** Relatively less work has been done on CRN-based VWNs. In [165], Tan et al. have presented a novel spectrum virtualization layer, that runs directly below the wireless PHY layer, that presents a seamless interface to the upper layers while allowing DSA. In [205], Nakauchi et al. have proposed AMPHIBIA as a cognitive virtualization platform that can be used to dynamically reconfigure wireless networks by integrated CR and network virtualization technology. More research needs to be done to explore the interplay between cognitive radio technology and virtualization technology.

### 4.4 Trend 4: cloud-based wireless networks (CbWNs)

The emergence of cloud computing is promising to revolutionize IT by democratizing and commoditizing computing. The main insight of cloud computing is to provide computing as a utility (i.e., using a pay-for-what-you-use pricing model) via a *centralized* setup exploiting economies of scale. The widespread popularity of cloud computing is being driven by advances in computing, virtualization, and web technology, along with

support for 'programming the network' through SDN and web APIs. The ability to program the network through the high-level abstractions defined by cloud/SDN API allow unprecedented dynamism in service provisioning, network management, and control. The broad idea behind cloud-based management of wireless networking to implement CbWNs is illustrated in Figure 6.

A recent trend in cloud computing is 'mobile cloud computing' (MCC) which integrates cloud computing in mobile environments to enable mobile wireless clients to perform computations in the cloud. A major challenge in MCC is to ensure that the mobile users are provisioned the necessary QoS even if the interfacing gateway changes due to node mobility. Misra et al. have proposed an auction-based QoS utility maximization-based approach for providing QoS-guaranteed bandwidth shifting and retribution in mobile clouds [170]. MCC can be used to overcome various performance impediments in mobile environments due to battery life, storage, bandwidth, and environment. These applications are discussed further in a detailed survey by Dinh et al. [171].

In the following subsections, we will introduce applications of CbWNs and will provide a network-specific categorization of CbWNs.

#### 4.4.1 Applications overview of CbWNs

- *Centralized management.* A major application of CbWNs is *centralized remote management* of wireless networks using which cloud-based provisioning of services can be performed scalably on the level of a large enterprise and/or service provider. The 'control and provisioning of wireless access points' (CAPWAP) protocol is a control and management (C&M) protocol defined by IETF in RFC 5415 [204] which is aimed at relocating some important functionality from the hardware AP equipment to an external controller. The CAPWAP protocol allows a software-defined architecture and is amenable to implementation via the cloud. There exists significant interest in the research community [169] in proposing efficient approaches for central management of Wi-Fi networks. Various industrial solutions, such as Meraki Networks [206] and Aruba Networks [207], have also been proposed that perform cloud-based management of WLANs.
- *Zero-touch auto-configuration.* The centralized management paradigm of CbWNs can allow *zero-touch auto-configuration* of wireless APs regardless of the location. The centralized management of wireless networks also allow cloud-based performance management and the use of *advanced data analytics* for optimization performance including *real-time reconfiguration* of wireless parameters.
- *Context-aware mobile services.* The centralized management paradigm of CbWNs can also facilitate provisioning of context-aware mobile services. As an example, Papakos et al. have proposed Volare as a

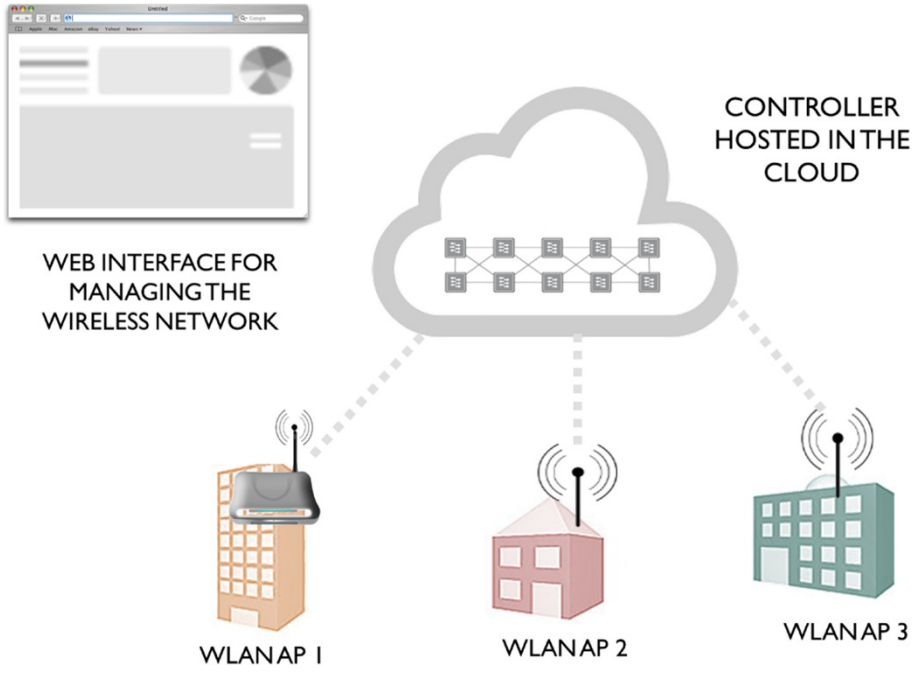


**Figure 6** Cloud-based wireless networks (CbWNs).

framework for context-aware adaptive cloud service discovery for mobile systems [208].

- *Optimized operations and management.* Apart from the ability to perform zero-touch auto-configuration, CbWNs can exploit their centralized management architecture to make dynamic changes in real-time in response to changing network conditions through the cloud-based management setup. As an example, Misra et al. [170] have proposed a framework for optimized QoS-guaranteed bandwidth redistribution among gateways in the settings of MCC. In addition, there has been work proposed for Papagianni et al. [209] for optimal allocation of VM resources in cloud computing environments with heterogeneous resources.
- *Computation offloading.* Another major application is computation offloading through which energy or computation-constrained devices can scalably handle complex tasks by offloading their computation to the cloud. The *cloudlet* approach [210] is to provision a trusted well-connected and resource-rich cluster of computers which is available for use by mobile devices that are close. When computation offload to the cloud is undesirable due to any reason (such as cost, delay, etc.), then mobile users can connect to a nearby cloudlet.
  Various other works have been proposed for offloading in a mobile cloud computing environment. Yang et al. have proposed techniques for effective offloading for resource-constrained mobile devices running resource-rich mobile Internet applications [166]. Chun et al. have proposed CloneCloud which allows unmodified mobile applications in an application-level VM to seamlessly offload part of their execution from the mobile devices onto device clones operating in the cloud [167]. Vestin et al. have proposed CloudMAC [175] architecture for managing enterprise WLANs through a virtual AP abstraction provided by the cloud. In the CloudMAC framework, OpenFlow is used to manage the network, and the physical CloudMAC APs are now simplified devices only tasked with relaying MAC frames between the virtual APs and mobile stations while the build of the processing and management functionality is now executed in the cloud.
- *Cloudifying mobile networks.* An extensive amount of work has focused on integrating cloud computing with mobile networks (including 4G [178] and 5G networks [177]) to improve the efficiency of next-generation mobile carrier networks [172]. In particular, there has been work done in extending cloud computing 'as-a-service' provisioning style to radio access networks [174,177].

#### 4.4.2 Interplay with other programmable wireless trends

In this section, we will describe the interplay of CbWNs with the other three trends of programmable wireless networking (SWNs, CWNs, and VWNs) that we have studied.

**Cloud-based software-defined WNs** Cloud technology and SDN technology have been used in previous work to introduce programmability features in wireless networking. As part of the NFV drive, there is significant interest in the mobile industry for transforming fixed network functions typically implemented in proprietary hardware devices (such as radio network controller, LTE base stations, etc.) to software-based virtualized instances running in the cloud. In parallel as part of the 'software-defined datacenter' (SDDC) drive, datacenters are increasingly turning to SDN technology for better management.

**Cloud-based cognitive WNs** There have been a few proposals for utilizing cloud technology to improve the performance in CWNs. In particular, cloud technology has been proposed for use with CWNs for cooperative spectrum sensing in the TV white spaces [176] and for a fast processing of vast volumes of data [211]. In the future, CWNs will perceivably leverage cloud technology increasingly to exploit its scalable computation capability along with its inherent programmability.

**Cloud-based virtualable WNs** Cloud technology has recently been used with VWNs to provide its scalability and service benefits in such environments. The CloudMAC project has proposed virtualizing APs in a datacenter [175]. Vassilaras et al. present an approach in [172] to provide wireless networking as a service (in utility computing style associated with cloud computing). Other VWN projects that have utilized cloud computing include the 'wireless network cloud' (WNC) project [173]. Due to their popular usage and associated benefits, cloud technology has also been proposed for use with SDRs [212] and in CRs [211].

#### 4.4.3 Network-specific categorization of CbWNs

As discussed in this paper, the use of cloud technology has been proposed in various wireless network settings. The trend of CbWNs is most prominent for WLANs, mobile cellular networks, and CRNs (as has been discussed already in this paper). In addition, cloud computing paradigm has also been applied to the setting of WSNs [213]. In WSNs, the network nodes are limited in their processing capability, battery life, and communication resources. WSNs can utilize the CbWN trend to offload its computation to the cloud and increase its efficiency.

## 5 Open research issues and challenges

Research in programmable wireless networks has indeed gained momentum as highlighted in this paper. However, many issues still remain to be resolved in order to fully realize the potential benefits associated with this paradigm. We highlight a few important research issues in this domain.

### 5.1 Building software-defined cognitive wireless networks

An initial promise of SDR was seamless interworking with a plethora of technologies through software-defined adaptations. The vision of CRNs, on the other hand, has evolved from the foundations of SDRs and aims to provide users with seamless holistic experience that integrates potentially heterogeneous technologies. The interplay of cognitive radios with the SDN architecture appears to be a viable and promising hybrid technology that can be used to create programmable wireless networks. We refer to this hybrid technology as 'software-defined cognitive wireless networking' (SCWN). Numerous interesting use-cases can plausibly emerge if we synergize the mainly centralized operational paradigm of SDNs with the mainly distributed operational paradigm of CRs. While the emphasis of SDN architecture has been on the separation of control and data planes, it is worth exploring if a combined SDN and CR architecture can help realize the vision of programmable wireless networks having a 'knowledge plane' as envisioned by Clark et al. [186].

### 5.2 Development of wireless-specific network APIs

The utility of any programming paradigm depends greatly on the abstractions available and the standardization of APIs. For the vision of programmable wireless networking to become established, it is important that there is progress in developing useful network APIs offering sophisticated high-level abstractions for wireless networking. In the SDN community, while numerous southbound APIs have been proposed, there is a lack of clarity and consensus about what abstractions and interface a northbound API should expose. While OpenFlow, an example southbound API, has become wildly popular, it is quite primitive in its functionality [17] - the metaphor of Assembly language programming is often used to describe direct OpenFlow programming. To increase the rate of innovation, it is important that higher level languages are developed that can be used by network programmers through a high-level standardized interface. In this regard, more work is required for both southbound and northbound API. Since a network application programmer will interface with the controller through a northbound API, quick consensus on the development of an effective northbound network API is of paramount importance.

### 5.3 Integrating wireless and cloud technologies

The paradigm of cloud computing - which is itself based on web technology, programmability through APIs and virtualization - is likely to play a big role in future wireless programmable networks. There is already a lot of work on integration of SDN and cloud technology [10]. Future designers of programmable wireless devices will be well-served by exploiting the performance and scalability advantages offered by cloud computing in their designs. Already, there has been significant work in incorporating cloud technology into existing frameworks (as has been discussed in Section 4.4), and this trend looks set to continue well into the future.

### 5.4 Wireless internet of things

While traditionally, the Internet communication paradigm has revolved around human consumption of Internet services, it is envisioned that in the future, networking will create many novel services through machine-to-machine communication by creating an Internet of things [214]. In particular, the convenience of untethered mobile communication facilitated by wireless communication can create a future wireless Internet of things. This is a potential future research area envisioned to have a significant impact on the community and the way stakeholders interact with the services provided by the Internet.

### 5.5 Balancing centralized and distributed paradigms

SDNs have proposed a separation of the control plane and data plane with the control logic placed on a separated controller. Pragmatic concerns about performance and security have dictated that this controller be implemented as a distributed system. Hence, although the controller is 'logically centralized', it is implemented as a distributed system - this has led to coining of the awkward term 'logically centralized control'. This draws our attention to the perennial tension between distributed and centralized control. The Internet's community has traditionally favored the distributed control paradigm due to its scalability. However, architectural ossification and inflexible network control has led through a rethink to the centralized SDN paradigm. Like the pre-SDN era, not all tasks can be, or should be, exclusively centralized or distributed. The modern shift to a centralized paradigm is sometimes codified in the mantra, 'centralize what you can, distribute what you must'. Finding the right balance between centralized and distributed control is an important fundamental design choice which needs careful evaluation. Also, it is important to address the scalability and performance concerns associated with the centralized control to make it a viable practical architecture. Future wireless networks will have to seamlessly manage the delicate balance between the centralized and distributed control

paradigms of current technologies - such as WiMAX and Wi-Fi - and the centralized aspects and distributed aspects of future network architecture such as SWNs and CWNs, respectively.

## 6 Conclusions

In this paper, we have provided a general overview of architectural techniques useful for building next-generation programmable wireless networks. We have seen that the seemingly disparate schemes of software-defined radio, cognitive radio networking, software-defined networking, and programmable wireless processors are in fact themed on a common goal of creating flexible 'programmable wireless networks'. A self-contained tutorial is provided for these architectures followed by a detailed survey of their applications. We also proposed synergizing these technologies into newer hybrid technologies. In particular, we proposed a new framework of *software-defined cognitive wireless networking* which will employ both SDN and CRN principles to potentially open up new use cases. We have also highlighted important research issues in this field and identified future research work.

## Endnotes

[a] Tennenhouse et al. also made the sobering observation in [16] that object-oriented approaches to networking are proposed every 5 to 10 years with little impact on mainstream research. Time will tell how transformative the modern proposals (e.g., SDN languages approach) will be in the long run.

[b] James Hamilton, the architect of Amazon's cloud, made the now famous remark in 2010 that 'datacenter networks are in my way' bemoaning the lack of network programmability. See Hamilton's blog post at http://perspectives.mvdirona.com/2010/10/31/DatacenterNetworksAreInMyWay.aspx for more perspective.

[c] David Clark, the former chair of the Internet Architecture Board (IAB), summarized the role of software implementations in the networking community ethos when he said, 'We reject kings, presidents and voting. We believe in rough consensus and *running code*'.

[d] The centralized SDN network controller can itself be built as a distributed system to be scalable and avoid a single point of failure.

[e] http://www.opendaylight.org.

[f] OpenFlow should not be confused with the overall SDN *architecture* since OpenFlow, popular as it is, is but only one *protocol* that implements the southbound API as envisioned in the SDN architecture.

**Competing interests**
The authors declare that they have no competing interests.

**Authors' information**
Junaid Qadir is an Assistant Professor at the School of Electrical Engineering and Computer Sciences (SEECS), National University of Sciences and Technology (NUST), Pakistan. He is also the Director of the Cognet Lab at SEECS. He completed his BE in Electrical Engineering from UET, Lahore, Pakistan and his PhD from University of New South Wales, Australia in 2008. His research interests include cognitive radio networks, wireless networks, and software-defined networks.
Nadeem Ahmed received the BE degree from the University of Engineering and Technology, Lahore, Pakistan and the MS and PhD degrees in computer sciences from University of New South Wales (UNSW), Sydney, Australia in 2000 and 2007, respectively. He is currently an Assistant Professor at the School of Electrical Engineering and Computer Sciences (SEECS), National University of Sciences and Technology (NUST), Pakistan. His research interests include wireless sensor networks, software-defined networks and mobile ad-hoc networks.
Nauman Ahad is presently working as a software engineer with xFlow Research Inc. Nauman graduated from the School of Electrical Engineering and Computer Sciences (SEECS), National University of Sciences and Technology (NUST), Pakistan with a BE in Electrical Engineering in 2014. He was an internee in the Cognet research lab at SEECS, NUST, in the summer of 2013.

**Author details**
[1] School of Electrical Engineering and Computer Sciences (SEECS), National University of Sciences and Technology (NUST), Islamabad 44000, Pakistan. [2] xFlow Research Inc., Islamabad 44400, Pakistan.

Received: 29 January 2014 Accepted: 10 October 2014
Published: 22 October 2014

**References**

1. D Raychaudhuri, NB Mandayam, Frontiers of wireless and mobile communications.Proc. IEEE. **100**(4), 824–840 (2012)
2. Akamai State of Internet Report Q1 (2013). http://www.akamai.com/stateoftheinternet/. Accessed 30 Sept 2013
3. T Koponen, S Shenker, H Balakrishnan, N Feamster, I Ganichev, A Ghodsi, P Godfrey, N McKeown, G Parulkar, B Raghavan, J Rexford, S Arianfar, D Kuptsov, Architecting for innovation. Comput. Comm. Rev. **41**(3), 24–36 (2011)
4. B Liskov, in *Distributed Computing*. The power of abstraction (Springer, Berlin Heidelberg, 2010), pp. 3–3
5. J Rexford, The networking philosopher's problem. Comput. Comm. Rev. **41**(3), 5–9 (2011)
6. Scott Shenker's talk at Open Networking Summit (2011). [Online] http://www.youtube.com/watch?v=YHeyuD89n1Y. Accessed 12 Sept 2013
7. J Rexford, P Zave, Report of the DIMACS working group on abstractions for network services, architecture, and implementation. Comput. Comm. Rev. **43**(1), 56–59 (2012)
8. N McKeown, T Anderson, H Balakrishnan, G Parulkar, L Peterson, J Rexford, S Shenker, J Turner, OpenFlow: enabling innovation in campus networks. Comput. Comm. Rev. **38**(2), 69–74 (2008)
9. M Andreessen, Why software is eating the world. Wall Street Journal (2011). [online] http://online.wsj.com/article/SB10001424053111903480904576512250915629460.html, Accessed 22 Sept 2013
10. Openstack's neutron plugin for "network-as-a-service". [online] https://wiki.openstack.org/wiki/Neutron. Accessed 22 Sept 2013
11. B Nunes, M Mendonca, X Nguyen, K Obraczka, T Turletti, A survey of software-defined networking: past, present, and future of programmable networks. IEEE Commun. Surv. Tutorials. **PP**(99), 1–18 (2014). IEEE, NY USA
12. 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, A Vahdat, in *Proceedings of the ACM SIGCOMM 2013*. B4: Experience with a globally-deployed software defined WAN (ACM, NY USA, 2013), pp. 3–14
13. Cisco's use of programmable networks for powering 'Internet of Everything'. [Online] http://tinyurl.com/o9y4mza. Accessed 1 Oct 2013
14. AT Campbell, HG De Meer, ME Kounavis, K Miki, JB Vicente, D Villela, A survey of programmable networks. Comput. Comm. Rev. **29**(2), 7–23 (1999)

15. JM Smith, SM Nettles, Active networking: one view of the past, present, and future. IEEE Trans. Syst. Man Cybern. C Appl. Rev. **34**(1), 4–18 (2004)
16. DL Tennenhouse, DJ Wetherall, in *DARPA Active Networks Conference and Exposition, 2002. Proceedings*. Towards an active network architecture (IEEE, NY USA, 2002), pp. 2–15
17. N Foster, A Guha, M Reitblatt, A Story, MJ Freedman, NP Katta, C Monsanto, J Reich, J Rexford, C Schlesinger, D Walker, R Harrison, Languages for software-defined networks. IEEE Comm. Mag. **51**(2), 128–134 (2013)
18. N Feamster, J Rexford, E Zegura, The road to SDN. Queue. **11**(12), 20 (2013)
19. AT Campbell, I Katzela, K Miki, J Vicente, Open signaling for ATM, internet and mobile networks (OPENSIG'98). Comput. Comm. Rev. **29**(1), 97–108 (1999)
20. DA Burgess, HS Samra, The OpenBTS Project (2008). [Online] http://openbts.sourceforge.net. Accessed 22 Sept 2013
21. JH Reed, *Software Radio: A Modern Approach to Radio Engineering*. (Prentice Hall Professional, USA, 2002)
22. WH Tuttlebee, *Software Defined Radio: Enabling Technologies*. (Wiley, USA, 2003)
23. T Ulversoy, Software defined radio: challenges and opportunities. IEEE Commun. Surv. Tutorials. **12**(4), 531–550 (2010). IEEE, NY USA
24. C Partridge, Realizing the future of wireless data communications. Commun. ACM. **54**(9), 62–68 (2011). ACM, NY USA
25. Ettus Research, acquired recently by National Instruments (NI). [Online] http://www.ettus.com. Accessed 12 Sept 2013
26. GNU Radio. [Online] http://www.gnuradio.com. Accessed 12 Sept 2013
27. OSSIE Virginia Tech. [Online] http://ossie.wireless.vt.edu/. Accessed 12 Sept 2013
28. P Mackenzie, *Software and Reconfigurability for Software Radio Systems*. (PhD thesis, Ph. D dissertation, Trinity College Dublin, Ireland, 2004)
29. M Ettus, USRP User's and Developer's Guide. Ettus Research LLC. [Online] http://www.olifantasia.com/gnuradio/usrp/files/usrp_guide.pdf Accessed 12 Sept 2013
30. A Khattab, J Camp, C Hunter, P Murphy, A Sabharwal, EW Knightly, WARP: a flexible platform for clean-slate wireless medium access protocol design. ACM SIGMOBILE Mobile Comput. Commun. Rev. **12**(1), 56–58 (2008). ACM
31. MC Ng, KE Fleming, M Vutukuru, S Gross, Arvind, H Balakrishnan, in *Proceedings of the 6th ACM/IEEE Symposium on Architectures for Networking and Communications Systems*. Airblue: a system for cross-layer wireless protocol development, (2010), p. 4. ACM
32. K Tan, H Liu, J Zhang, Y Zhang, J Fang, GM Voelker, Sora: high-performance software radio using general-purpose multi-core processors. Commun. ACM. **54**(1), 99–107 (2011). ACM
33. M Bansal, J Mehlman, S Katti, P Levis, in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*. Openradio: a programmable wireless dataplane (ACM, 2012), pp. 109–114
34. G Stewart, M Gowda, G Mainland, B Radunovic, D Vytiniotis, Ziria: wireless programming for hardware dummies. Tech. Rep. MSR-TR-2013-135, Microsoft Systems, November 2013
35. G Stewart, M Gowda, G Mainland, B Radunovic, D Vytiniotis, D Patterson, in *Proceedings of the 2014 ACM conference on SIGCOMM*. Ziria: language for rapid prototyping of wireless PHY (ACM, 2014), pp. 357–358
36. SM Mishra, D Cabric, C Chang, D Willkomm, B Van Schewick, S Wolisz, B Brodersen, in *New Frontiers in Dynamic Spectrum Access Networks (DYSPAN), 2005. First IEEE International Symposium on*. A real time cognitive radio testbed for physical and link layer experiments (IEEE, NY USA, 2005), pp. 562–567
37. Y Yuan, P Bahl, R Chandra, PA Chou, JI Ferrell, T Moscibroda, S Narlanka, Y Wu, in *New Frontiers in Dynamic Spectrum Access Networks, 2007, DySPAN 2007. 2nd IEEE International Symposium on*. KNOWS: cognitive radio networks over white spaces (IEEE, 2007), pp. 416–427
38. Z Miljanic, I Seskar, K Le, D Raychaudhuri, The WINLAB network centric cognitive radio hardware platform: WiNC2R. Mobile Network Appl. **13**(5), 533–541 (2008)
39. AA Lazar, K-S Lim, F Marconcini, xbind: The system programmer's manual. report # 452-96-16. tech. rep., Center for Telecommunications Research (CTR), Columbia University, 1996
40. S Rooney, JE van der Merwe, SA Crosby, IM Leslie, The tempest: a framework for safe, resource assured, programmable networks. IEEE Comm. Mag. **36**(10), 42–53 (1998)
41. O Angin, AT Campbell, ME Kounavis, R-F Liao, The mobiware toolkit: programmable support for adaptive mobile networking. IEEE Pers. Comm. **5**(4), 32–43 (1998)
42. A Campbell, H De Meer, M Kounavis, K Miki, J Vicente, D Villela, in *Open Architectures and Network Programming Proceedings, 1999. OPENARCH'99. 1999 IEEE Second Conference on*. The genesis kernel: a virtual network operating system for spawning network architectures (IEEE, NY USA, 1999), pp. 115–127
43. R Morris, E Kohler, J Jannotti, MF Kaashoek, in *ACM SIGOPS Operating Systems Review*. The click modular router, vol. 33 (ACM, NY USA, 1999), pp. 217–231
44. M Handley, O Hodson, E Kohler, XORP: an open platform for network research. Comput. Comm. Rev. **33**(1), 53–57 (2003). ACM, NY USA
45. M Neufeld, J Fifield, C Doerr, A Sheth, D Grunwald, in *ACM Workshop on Hot Topics in Networks (HotNets)*. SoftMAC: flexible wireless research platform (ACM, NY USA, 2005)
46. C Doerr, M Neufeld, J Fifield, T Weingart, DC Sicker, D Grunwald, in *New Frontiers in Dynamic Spectrum Access Networks (DYSPAN), 2005. First IEEE International Symposium on*. MultiMAC - an adaptive MAC framework for dynamic radio networking (IEEE, NY USA, 2005), pp. 548–555
47. JW Lockwood, N McKeown, G Watson, G Gibb, P Hartke, J Naous, R Raghuraman, J Luo, in *Microelectronic Systems Education, 2007. MSE'07. IEEE International Conference on*. NetFPGA–an open platform for gigabit-rate network switching and routing (IEEE, NY USA, 2007), pp. 160–161
48. J Naous, G Gibb, S Bolouki, N McKeown, in *Proceedings of the ACM Workshop on Programmable Routers for Extensible Services of Tomorrow*. NetFPGA: reusable router architecture for experimental research (ACM, NY USA, 2008), pp. 1–7
49. M Dobrescu, N Egi, K Argyraki, B-G Chun, K Fall, G Iannaccone, A Knies, M Manesh, S Ratnasamy, in *Proceedings of the 22nd ACM Symposium on Operating Systems Principles (SIGOPS)*. Routebricks: exploiting parallelism to scale software routers (ACM, NY USA, 2009), pp. 15–28
50. MB Anwer, M Motiwala, Mb Tariq, N Feamster, Switchblade: a platform for rapid deployment of network protocols on programmable hardware. Comput. Comm. Rev. **40**(4), 183–194 (2010). ACM, NY USA
51. J Ansari, X Zhang, A Achtzehn, M Petrova, P Mahonen, in *New Frontiers in Dynamic Spectrum, 2010 IEEE Symposium on*. Decomposable MAC framework for highly flexible and adaptable MAC realizations (IEEE, NY USA, 2010), pp. 1–2
52. X Zhang, J Ansari, G Yang, P Mahonen, in *New Frontiers in Dynamic Spectrum Access Networks (DySPAN) 2011 IEEE Symposium on*. Trump: supporting efficient realization of protocols for cognitive radio networks (IEEE, NY USA, 2011), pp. 476–487
53. I Tinnirello, G Bianchi, P Gallo, D Garlisi, F Giuliano, F Gringoli, in *INFOCOM, 2012, Proceedings, IEEE*. Wireless MAC processors: programming MAC protocols on commodity hardware (IEEE, NY USA, 2012), pp. 1269–1277
54. G Bianchi, P Gallo, D Garlisi, F Giuliano, F Gringoli, I Tinnirello, in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*. MAClets: active MAC protocols over hard-coded devices (ACM, NY USA, 2012), pp. 229–240
55. FK Jondral, Software-defined radio: basics and evolution to cognitive radio. EURASIP J. Wireless Commun. Netw. **2005**(3), 275–283 (2005). Springer Heidelberg
56. J Mitola III, *Cognitive Radio Architecture: The Engineering Foundations of Radio XML*. (John Wiley & Sons, USA, 2006)
57. A He, KK Bae, TR Newman, J Gaeddert, K Kim, R Menon, L Morales-Tirado, JJ Neel, Y Zhao, JH Reed, W Tranter, A survey of artificial intelligence for cognitive radios. IEEE Trans. Veh. Tech. **59**(4), 1578–1592 (2010)
58. I Akyildiz, W Lee, M Vuran, S Mohanty, Next generation/dynamic spectrum access/cognitive radio wireless networks: a survey. Comput. Network. **50**(13), 2127–2159 (2006)
59. S Haykin, Cognitive radio: brain-empowered wireless communications. IEEE J. Sel. Area Comm. **23**(2), 201–220 (2005)
60. BA Fette, *Cognitive Radio Technology*. (Academic Press, Elsevier, USA, 2009)

61. J Qadir, Artificial intelligence based cognitive routing for cognitive radio networks. arXiv preprint arXiv:1309.0085 (2013). [Online] http://arxiv.org/abs/1309.0085. Accessed 12 Sept 2013
62. G Baldini, T Sturman, AR Biswas, R Leschhorn, G Gódor, M Street, Security aspects in software defined radio and cognitive radio networks: a survey and a way ahead. IEEE Commun. Surv. Tutorials. **14**(2), 355–379 (2012). IEEE, NY USA
63. D Raychaudhuri, NB Mandayam, JB Evans, BJ Ewy, S Seshan, P Steenkiste, in *Proceedings of first ACM/IEEE International Workshop on Mobility in the Evolving Internet Architecture*. Cognet: an architectural foundation for experimental cognitive radio networks within the future internet (ACM, NY USA, 2006), pp. 11–16
64. RW Thomas, DH Friend, LA DaSilva, AB MacKenzie, in *Cognitive Radio, Software Defined Radio, and Adaptive Wireless Systems*, ed. by H Arslan. Cognitive networks (Springer, Netherlands, 2007), pp. 17–41
65. G Nychis, T Hottelier, Z Yang, S Seshan, P Steenkiste, in *USENIX NSDI*. Enabling MAC protocol implementations on software-defined radios, vol. 9 (USENIX, 2009), pp. 91–105
66. P Gallo, A Krasilov, A Lyakhov, I Tinnirello, G Bianchi, in *ICT Convergence (ICTC), 2012 International Conference on*. Breaking layer 2: a new architecture for programmable wireless interfaces (IEEE, NY USA, 2012), pp. 342–347
67. D Messerschmitt, Rethinking components: from hardware and software to systems. Proc. IEEE. **95**(7), 1473–1496 (2007). IEEE, NY USA
68. V Gazis, E Patouni, N Alonistioti, L Merakos, A survey of dynamically adaptable protocol stacks. IEEE Commun. Surv. Tutorials. **12**(1), 3–23 (2010). IEEE, NY USA
69. DL Tennenhouse, JM Smith, WD Sincoskie, DJ Wetherall, GJ Minden, A survey of active network research. IEEE Commun. Mag. **35**(1), 80–86 (1997). IEEE, NY USA
70. AA Lazar, Programming telecommunication networks. IEEE Netw. **11**(5), 8–18 (1997). IEEE, NY USA
71. D Wetherall, D Legedza, J Guttag, Introducing new internet services: why and how. IEEE Netw. **12**(3), 12–19 (1998). IEEE, NY USA
72. D Wetherall, in *DARPA Active NEtworks Conference and Exposition, 2002. Proceedings*. Active network vision and reality: lessons from a capsule-based system (IEEE, NY USA, 2002), pp. 25–40
73. K Psounis, Active networks: applications, security, safety, and architectures. IEEE Commun. Surv. Tutorials. **2**(1), 2–16 (1999). IEEE, NY USA
74. ActiveWare project at MIT. http://www.sds.lcs.mit.edu/activeware/. Accessed 12 Sept 2013
75. CANES project at Georgia Tech. [Online] http://www.cc.gatech.edu/projects/canes/. Accessed 22 Sept 2013
76. SwitchWare project at University of Pennsylvania. [Online] http://www.cis.upenn.edu/switchware/. Accessed 22 Sept 2013
77. D Wetherall, J Guttag, D Tennenhouse, Ants: network services without the red tape. Computer. **32**(4), 42–48 (1999). IEEE, NY USA
78. ANTS project at the University of Washington. [Online] http://www.cs.washington.edu/research/networking/ants/. Accessed 22 Sept 2013
79. A Doria, F Hellstrand, K Sundell, T Worster, General switch management protocol (GSMP) v3.IETF Request for Comments: 3292 (2002). [Online] https://www.ietf.org/rfc/rfc3292.txt. Accessed 12 Sept 2013
80. L Yang, R Dantu, T Anderson, R Gopal, RFC 3746: Forwarding and control element separation (FORCES) framework. Network Working Group. **4** (2004). [Online] http://tools.ietf.org/html/rfc3746. Accessed 12 Sept 2013
81. N Feamster, H Balakrishnan, J Rexford, A Shaikh, J Van Der Merwe, in *Proceedings of the ACM SIGCOMM Workshop on Future Directions in Network Architecture*. The case for separating routing from routers (ACM, NY USA, 2004), pp. 5–12
82. T Lakshman, T Nandagopal, R Ramjee, K Sabnani, T Woo, in *Proc. ACM SIGCOMM Workshop on Hot Topics in Networking*. The softrouter architecture (ACM, NY USA, p. 2004
83. A Greenberg, G Hjalmtysson, DA Maltz, A Myers, J Rexford, G Xie, H Yan, J Zhan, H Zhang, A clean slate 4d approach to network control and management. Comput. Comm. Rev. **35**(5), 41–54 (2005). ACM, NY USA
84. K Lakshminarayanan, I Stoica, S Shenker, J Rexford, *Routing as a Service*. Computer Science Division, Technical Report No. UCB/EECS-2006-19. University of California Berkeley, 2004)
85. A Farrel, J-P Vasseur, J Ash, A path computation element (pce)-based architecture. tech. rep., RFC, 4655 (2006)
86. J Van der Merwe, A Cepleanu, K D'Souza, B Freeman, A Greenberg, D Knight, R McMillan, D Moloney, J Mulligan, H Nguyen, M Nguyen, A Ramarajan, S Saad, M Satterlee, T Spencer, D Toll, S Zelingher, in *Proceedings of the 2006 SIGCOMM Workshop on Internet Network Management*. Dynamic connectivity management with an intelligent route service control point (ACM, NY USA, 2006), pp. 29–34
87. M Casado, T Garfinkel, A Akella, MJ Freedman, D Boneh, N McKeown, S Shenker, in *ACM USENIX Security Symposium*. SANE: a protection architecture for enterprise networks (ACM, 2006)
88. M Casado, MJ Freedman, J Pettit, J Luo, N McKeown, S Shenker, Ethane: taking control of the enterprise. Comput. Commun. Rev. **37**, 1–12 (2007). ACM
89. Juniper Networks. http://www.juniper.net. Accessed 12 Sept 2013
90. Cisco's One Platform Kit (onePK). http://www.cisco.com/en/US/prod/iosswrel/onepk.html. Accessed 12 Sept 2013
91. OpenStack's Neutron. https://wiki.openstack.org/wiki/Neutron? Accessed 12 Sept 2013
92. B Chun, D Culler, T Roscoe, A Bavier, L Peterson, M Wawrzoniak, M Bowman, Planetlab: an overlay testbed for broad-coverage services. Comput. Comm. Rev. **33**(3), 3–12 (2003). ACM, NY USA
93. T Anderson, L Peterson, S Shenker, J Turner, Overcoming the internet impasse through virtualization. Computer. **38**(4), 34–41 (2005). IEEE, NY USA
94. R Sherwood, G Gibb, K-K Yap, G Appenzeller, M Casado, N McKeown, G Parulkar, Flowvisor: a network virtualization layer. OpenFlow Switch Consortium, Tech. Rep (2009). [Online] http://archive.openflow.org/downloads/technicalreports/openflow-tr-2009-1-flowvisor.pdf. Accessed 12 Sept 2013
95. R Sherwood, M Chan, A Covington, G Gibb, M Flajslik, N Handigol, T-Y Huang, P Kazemian, M Kobayashi, D Underhill, K Yap, G Appenzeller, N McKeown, Carving research slices out of your production networks with Openflow. Comput. Comm. Rev. **40**(1), 129–130 (2010). ACM, NY USA
96. C Guo, G Lu, HJ Wang, S Yang, C Kong, P Sun, W Wu, Y Zhang, in *Proceedings of the 6th ACM International Conference on emerging Networks EXperiments and Technologies (Co-NEXT'10)*. Secondnet: a data center network virtualization architecture with bandwidth guarantees (ACM, NY USA, 2010), pp. 1–12
97. T Koponen, K Amidon, P Balland, M Casado, A Chanda, B Fulton, I Ganichev, J Gross, N Gude, P Ingram, E Jackson, A Lambeth, R Lenglet, S Li, A Padmanabhan, J Pettit, B Pfaff, R Ramanathan, S Shenker, A Shieh, J Stribling, P Thakkar, D Wendlandt, A Yip, R Zhang, in *USENIX Networked Systems Design and Implementation (NSDI)*. Network virtualization in multi-tenant datacenters (USENIX, 2014)
98. M Armbrust, A Fox, R Griffith, AD Joseph, R Katz, A Konwinski, G Lee, D Patterson, A Rabkin, I Stoica, M Zaharia, A view of cloud computing. Commun. ACM. **53**(4), 50–58 (2010). ACM, NY USA
99. MD Dikaiakos, D Katsaros, P Mehra, G Pallis, A Vakali, Cloud computing: distributed internet computing for it and scientific research. IEEE Internet Comput. **13**(5), 10–13 (2009). IEEE, NY USA
100. M Casado, T Koponen, S Shenker, A Tootoonchian, in *Proceedings of the First Workshop on Hot topics in Software Defined Networks*. Fabric: a retrospective on evolving SDN (ACM, NY USA, 2012), pp. 85–90
101. What's Behind Network Downtime? - IBM. www-05.ibm.com/uk/juniper/pdf/200249.pdf?. Accessed 30 Sept 2013
102. Statistics on "virtual server access port" vs. "physical server access port"; Source: Crehan Research Inc. and VMWare Estimates, March 2003. www.crehan.com?. Accessed 30 Sept 2013
103. Scott Shenker's talk at Stanford (2013). [Online] http://www.youtube.com/watch?v=WabdXYzCAOU. Accessed 12 Sept 2013
104. B Pfaff, J Pettit, K Amidon, M Casado, T Koponen, S Shenker, in *ACM Workshop on Hot Topics in Networks (Hotnets)*. Extending networking into the virtualization layer (ACM, NY USA, p. 2009
105. A Wang, M Iyer, R Dutta, GN Rouskas, I Baldine, Network virtualization: technologies, perspectives, and frontiers. J. Lightwave Technol. **31**(4), 523–537 (2013)
106. N Chowdhury, R Boutaba, A survey of network virtualization. Comput. Network. **54**(5), 862–876 (2010)
107. N McKeown, Software-defined networking. INFOCOM Keynote Talk, Apr 2009. [Online] http://infocom2009.ieee-infocom.org/keynotes.html. Accessed 12 Sept 2013
108. Nicira. [Online] https://www.vmware.com/products/nsx. Accessed 12 Sept 2013

109. Contrail. [Online] http://www.contrail.com. Accessed 12 Sept 2013
110. J Chung, G Gonzalez, I Armuelles, T Robles, R Alcarria, A Morales, in *Communications (LATINCOM) 2012 IEEE Latin-America Conference on*. Experiences and challenges in deploying OpenFlow over a real wireless mesh network IEEE, NY USA, 2012), pp. 1–5
111. I. IW Group, Interface to the routing system (I2RS). Internet Engineering Task Force, 2014 (2014). [Online] Available: http://datatracker.ietf.org/wg/i2rs/charter/. Accessed 12 Sept 2013
112. R Alimi, R Penno, Y Yang, Application-layer traffic optimization protocol. Internet Draft, Internet Engineering Task Force, March 2014 (2014). [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-alto-protocol/. Accessed 12 Sept 2013
113. A Lara, A Kolasani, B Ramamurthy, Network innovation using OpenFlow: a survey. IEEE Commun. Surv. Tutorials. **16** (2013). IEEE, NY USA
114. N Gude, T Koponen, J Pettit, B Pfaff, M Casado, N McKeown, S Shenker, NOX: towards an operating system for networks. Comput. Commun. Rev. **38**(3), 105–110 (2008). ACM, NY USA
115. T Koponen, M Casado, N Gude, J Stribling, L Poutievski, M Zhu, R Ramanathan, Y Iwata, H Inoue, T Hama, S Shenker, in *USENIX OSDI*. Onix: a distributed control platform for large-scale production networks, vol. 10 (USENIX, 2010), pp. 1–6
116. Open networking operating system. http://onlab.us/tools.html. Accessed 12 Sept 2013
117. MR Nascimento, CE Rothenberg, MR Salvador, MF Magalhães, Quagflow: partnering quagga with openflow. Comput. Commun. Rev. **40**, 441–442 (2010). ACM, NY USA
118. Quagga Routing Suite. [Online] http://www.quagga.net. Accessed 12 Sept 2013
119. MR Nascimento, CE Rothenberg, MR Salvador, MF Magalhaes, CN Corrêa, SC de Lucena, The RouteFlow approach to IP routing services on software-defined networks. Brazilian Symposium on Computer Networks and Distributed Systems (SBRC) (2011). [Online] sbrc2011.facom.ufms.br/files/workshops/wpeif/ST02_2.pdf. Accessed 12 Sept 2013
120. MR Nascimento, CE Rothenberg, MR Salvador, CN Corrêa, SC de Lucena, MF Magalhães, in *Proceedings of the 6th International Conference on Future Internet Technologies*. Virtual routers as a service: the routeflow approach leveraging software-defined networks (ACM, NY USA, 2011), pp. 34–37
121. M Ghobadi, SH Yeganeh, Y Ganjali, in *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*. Rethinking end-to-end congestion control in software-defined networks (ACM, NY USA, 2012), pp. 61–66
122. HE Egilmez, ST Dane, KT Bagci, AM Tekalp, in *Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*. OpenQoS: an OpenFlow controller design for multimedia delivery with end-to-end quality of service over software-defined networks (IEEE, NY USA, 2012), pp. 1–8
123. K-K Yap, M Kobayashi, R Sherwood, T-Y Huang, M Chan, N Handigol, N McKeown, *Openroads: empowering research in mobile networks*, vol. 40, (2010), pp. 125–126. ACM, NY USA
124. T Metsch, A Edmonds, P Harsh, F Antonescu, *Real-time Analytics System for Service Operations & Management*. ACM symposium applied computing. (ACM, NY USA, 2014), pp. 1–8
125. H Ali-Ahmad, C Cicconetti, A De la Oliva, V Mancuso, MR Sama, P Seite, S Shanmugalingam, in *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*. An SDN-based network architecture for extremely dense wireless networks (IEEE, NY USA, 2013), pp. 1–7
126. M Karimzadeh, L Valtulina, G Karagiannis, in *The 4th International Conference on Cloud Computing and Services Science (CLOSER)*. Applying SDN/OpenFlow in Virtualized LTE to support distributed mobility management (DMM) (Citeseer, 2014)
127. H Yang, Y Kim, Routing optimization with SDN (2013). http://tools.ietf.org/html/draft-yang-dmm-sdn-dmm-00. Accessed 12 Sept 2013
128. K-K Yap, R Sherwood, M Kobayashi, T-Y Huang, M Chan, N Handigol, N McKeown, G Parulkar, in *Proceedings of the second ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (VISA)*. Blueprint for introducing innovation into wireless mobile networks (ACM, NY USA, 2010), pp. 25–32
129. AY Ding, J Crowcroft, S Tarkoma, in *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*. Poster: Softoffload: a programmable approach toward collaborative mobile traffic offloading (ACM, NY USA, 2014), pp. 368–368
130. IF Akyildiz, A Lee, P Wang, M Luo, W Chou, A roadmap for traffic engineering in software defined networks. Comput. Network. **71**, 1–30 (2014). ACM, NY USA
131. G Yi, S Lee, in *Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication*. Fully distributed handover based on SDN in heterogeneous wireless networks (ACM, NY USA, 2014), p. 70
132. J Schulz-Zander, L Suresh, N Sarrar, A Feldmann, T Hühn, R Merz, in *2014 USENIX Annual Technical Conference (USENIX ATC 14)*. Programmatic orchestration of wifi networks (USENIX Association, 2014)
133. AY Ding, J Crowcroft, S Tarkoma, H Flinck. Software defined networking for security enhancement in wireless mobile networks, vol. 66 (ACM, NY USA, 2014), pp. 94–101
134. L Suresh, J Schulz-Zander, R Merz, A Feldmann, T Vazao, in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*. Towards programmable enterprise WLANs with Odin (ACM, NY USA, 2012), pp. 115–120
135. K Pentikousis, Y Wang, W Hu, Mobileflow: toward software-defined mobile networks. IEEE Commun. Mag. **51**(7), 44–53 (2013). IEEE, NY USA
136. A Gudipati, D Perry, LE Li, S Katti, in *Proceedings of the Second ACM SIGCOMM Workshop on Hot topics in Software Defined Networking*. Softran: software defined radio access network (ACM, NY USA, 2013), pp. 25–30
137. X Jin, LE Li, L Vanbever, J Rexford, *SoftCell: scalable and flexible cellular core network architecture*, vol. 2013. (ACM, NY USA, 2013), pp. 163–174
138. T Luo, H-P Tan, TQ Quek, Sensor openflow: enabling software-defined wireless sensor networks. IEEE Commun. Lett. **16**(11), 1896–1899 (2012). IEEE, NY USA
139. S Costanzo, L Galluccio, G Morabito, S Palazzo, in *European Workshop on Software Defined Networking (EWSDN)*. Software defined wireless networks: unbridling SDNs, vol. 2012 (IEEE, NY USA, 2012), pp. 1–6
140. QH Mahmoud, *Cognitive Networks: Towards Self-Aware Networks*. (Wiley, USA, 2007)
141. B Manoj, RR Rao, M Zorzi, in *Cognitive Wireless Networks*. Architectures and protocols for next generation cognitive networking (Springer, Heidelberg, 2007), pp. 271–284
142. C Fortuna, M Mohorcic, Trends in the development of communication networks: cognitive networks. Comput. Networks. **53**(9), 1354–1376 (2009)
143. M Cesana, F Cuomo, E Ekici, Routing in cognitive radio networks: challenges and solutions. Ad Hoc Networks. **9**(3), 228–248 (2011)
144. Q Zhao, BM Sadler, A survey of dynamic spectrum access. IEEE Signal Process. Mag. **24**(3), 79–89 (2007)
145. Z Zhang, K Long, J Wang, Self-organization paradigms and optimization approaches for cognitive radio technologies: a survey. IEEE Wireless Comm. **20**(2), 36–42 (2013)
146. C Cormio, KR Chowdhury, A survey on MAC protocols for cognitive radio networks. Ad Hoc Networks. **7**(7), 1315–1329 (2009)
147. A De Domenico, EC Strinati, M Di Benedetto, A survey on MAC strategies for cognitive radio networks. IEEE Communicat. Surv. Tutorials. **14**(1), 21–44 (2012)
148. A Azarfar, J-F Frigon, B Sanso, Improving the reliability of wireless networks using cognitive radios. IEEE Communicat. Surv. Tutorials. **14**(2), 338–354 (2012)
149. AG Fragkiadakis, EZ Tragos, IG Askoxylakis, A survey on security threats and detection techniques in cognitive radio networks. IEEE Communicat. Surv. Tutorials. **15**(1), 428–445 (2013)
150. D Niyato, E Hossain, Cognitive radio for next-generation wireless networks: an approach to opportunistic channel selection in IEEE 802.11-based wireless mesh. IEEE Wireless Commun. **16**(1), 46–54 (2009)
151. R Ahuja, R Corke, A Bok, in *New Frontiers in Dynamic Spectrum Access Networks, 2008. DySPAN 2008. 3rd IEEE Symposium on*. Cognitive radio system using IEEE 802.11 over UHF TVWS (IEEE, NY USA, 2008), pp. 1–9
152. AB Flores, RE Guerra, EW Knightly, P Ecclesine, S Pandey, "IEEE 802.11 af: a standard for tv white space spectrum sharing. IEEE Commun. Mag. **51**(10), 92–100 (2013). IEEE, NY USA
153. C Cordeiro, K Challapali, D Birru, N Sai Shankar, in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005*. IEEE 802.22: the first worldwide wireless standard based on cognitive radios (IEEE, NY USA, 2005), pp. 328–337
154. OB Akan, O Karli, O Ergul, Cognitive radio sensor networks. IEEE Netw. **23**(4), 34–40 (IEEE, NY USA, 2009)

155. M Di Felice, R Doost-Mohammady, KR Chowdhury, L Bononi, Smart radios for smart vehicles: cognitive vehicular networks. IEEE Veh. Tech. Mag. **7**(2), 26–33 (2012). IEEE, NY USA
156. X Costa-Perez, J Swetina, T Guo, R Mahindra, S Rangarajan, Radio access network virtualization for future mobile carrier networks. IEEE Comm. Mag. **51**(7), 27–35 (2013). IEEE, NY USA
157. G Bhanage, I Seskar, R Mahindra, D Raychaudhuri, in *Proceedings of the Second ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (VISA)*. Virtual basestation: architecture for an open shared WiMAX framework (ACM, NY USA, 2010), pp. 1–8
158. Y Zaki, L Zhao, C Goerg, A Timm-Giel, in *Wireless and Mobile Networking Conference (WMNC), 2010 Third Joint IFIP*. LTE wireless virtualization and spectrum management (IEEE, NY USA, 2010), pp. 1–6
159. G Smith, A Chaturvedi, A Mishra, S Banerjee, in *Proceedings of the Second ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization*. Wireless virtualization on commodity 802.11 hardware (ACM, NY USA, 2007), pp. 75–82
160. T Hamaguchi, T Komata, T Nagai, H Shigeno, in *24th International Conference on Advanced Information Networking and Applications Workshops (WAINA) 2010 IEEE*. A framework of better deployment for, WLAN access point using virtualization technique (IEEE, NY USA, 2010), pp. 968–973
161. L Xia, S Kumar, X Yang, P Gopalakrishnan, Y Liu, S Schoenberg, X Guo, in *ACM SIGPLAN Notices, vol 46*. Virtual WiFi: bring virtualization from wired to wireless (ACM, NY USA, 2011), pp. 181–192
162. Y He, J Fang, J Zhang, H Shen, K Tan, Y Zhang, Mpap: virtualization architecture for heterogenous wireless aps. Comput. Comm. Rev. **40**, 475–476 (2010). ACM, NY USA
163. V Sivaraman, T Moors, H Habibi Gharakheili, D Ong, J Matthews, C Russell, in *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*. Virtualizing the access network via open apis (ACM, NY USA, 2013), pp. 31–42
164. VD Philip, Y Gourhant, D Zeghlache, in *e-Infrastructure and e-Services for Developing Countries*. Openflow as an architecture for e-node b virtualization (Springer, Heidelberg, 2012), pp. 49–63
165. K Tan, H Shen, J Zhang, Y Zhang, in *Dynamic Spectrum Access Networks (DYSPAN), 2012 IEEE International Symposium on*. Enable flexible spectrum access with spectrum virtualization (IEEE, NY USA, 2012), pp. 47–58
166. K Yang, S Ou, H-H Chen, On effective offloading services for resource-constrained mobile devices running heavier mobile internet applications. IEEE Commun. Mag. **46**(1), 56–63 (2008). IEEE, NY USA
167. B-G Chun, S Ihm, P Maniatis, M Naik, A Patti, in *Proceedings of the Sixth Conference on Computer Systems, EuroSys '11*. Clonecloud: elastic execution between mobile device and cloud (ACM, NY USA, 2011), pp. 301–314
168. P Calhoun, Lightweight access point protocol (2010). http://tools.ietf.org/html/rfc5412. Accessed 12 Sept 2013
169. A Dalvi, P Swamy, B Meshram, in *Computer Networks and Information Technologies*. Centralized management approach for, WLAN (Springer, 2011), pp. 578–580
170. S Misra, S Das, M Khatua, M Obaidat, QoS-guaranteed bandwidth shifting and redistribution in mobile cloud environment. IEEE Trans. Cloud Comput. **99**, 1–1 (2013). IEEE, NY USA
171. HT Dinh, C Lee, D Niyato, P Wang, A survey of mobile cloud computing: architecture, applications, and approaches. Wireless Comm. Mobile Comput (Wiley, USA, 2011)
172. S Vassilaras, GS Yovanof, Wireless going in the cloud: a promising concept or just marketing hype? Wireless Pers. Comm. **58**(1), 5–16 (2011). Springer
173. Y Lin, L Shao, Z Zhu, Q Wang, RK Sabhikhi, Wireless network cloud: architecture and system requirements. IBM J. Res. Dev. **54**(1), 4–1 (2010). IBM
174. L Ferreira, D Pichon, A Hatefi, A Gomes, D Dimitrova, T Braun, G Karagiannis, M Karimzadeh, M Branco, L Correia, An architecture to offer cloud-based radio access network as a service. Networks and Communications (EuCNC), 2014 European Conference on (IEEE, 2014), pp. 1–5
175. J Vestin, P Dely, A Kassler, N Bayer, H Einsiedler, C Peylo, CloudMAC: towards software defined WLANs. ACM SIGMOBILE Mobile Comput. Commun. Rev. **16**(4), 42–45 (2013). ACM, NY USA
176. C-H Ko, DH Huang, S-H Wu, in *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on*. Cooperative spectrum sensing in TV white spaces: when cognitive radio meets cloud (IEEE, NY USA, 2011), pp. 672–677
177. P Rost, CJ Bernardos, A Domenico, M Girolamo, M Lalam, A Maeder, D Sabella, D Wübben, Cloud technologies for flexible 5G radio access networks. IEEE Commun. Mag. **52**(5), 68–76 (2014). IEEE, NY USA
178. AJ Staring, in *Proc. of the 17th Twente Student Conference on IT*. Applying the cloud computing model in LTE based cellular systems (University of Twente, 2012)
179. MB Anwer, N Feamster, Building a fast, virtualized data plane with programmable hardware. Comput. Commun. Rev. **40**(1), 75–82 (2010). ACM, NY USA
180. C Chaudet, Y Haddad, in *Microwaves, Communications, Antennas and Electronics Systems (COMCAS), 2013 IEEE International Conference on*. Wireless software defined networks: challenges and opportunities (IEEE, NY USA, 2013), pp. 1–5
181. JC Zuniga, CJ Bernardos, A de la Oliva, T Melia, R Costa, A Reznik, Distributed mobility management: a standards landscape. IEEE Comm. Mag. **51**(3), 80–87 (2013). IEEE, NY USA
182. LE Li, ZM Mao, J Rexford, in *European Workshop on Software Defined Networking (EWSDN), 2012*. Toward software-defined cellular networks (IEEE, NY USA, 2012), pp. 7–12
183. MJ Randall Schwartz, White paper: Carrier Wi-Fi Offload. http://tinyurl.com/carrierwifi. Accessed 12 Sept 2013
184. J Zheng, MJ Lee, *A comprehensive performance study of IEEE 802.15.4. Sensor Network Operations, ISBN 0-471-71976-5, Chapter 4 218–237*. (IEEE Press, Wiley Interscience, 2006)
185. CK Singh, A Kumar, P Ameer, Performance evaluation of an IEEE 802.15.4 sensor network with a star topology. Wireless Network. **14**(4), 543–568 (2008). Springer
186. DD Clark, C Partridge, JC Ramming, JT Wroclawski, in *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*. A knowledge plane for the internet (ACM, 2003), pp. 3–10
187. M Sherman, AN Mody, R Martinez, C Rodriguez, R Reddy, IEEE standards supporting cognitive radio and networks, dynamic spectrum access, and coexistence. IEEE Commun. Mag. **46**(7), 72–79 (IEEE, NY USA, 2008)
188. A Sarigiannidis, P Nicopolitidis, G Papadimitriou, P Sarigiannidis, M Louta, A Pomportsis, On the use of learning automata in tuning the channel split ratio of wimax networks. IEEE Systems Journal. **PP**(99), 1–13 (2013). IEEE, NY USA
189. S Misra, P Venkata, K Krishna Isaac, N Abraham Sasikumar, S Fredun, An adaptive learning routing protocol for the prevention of distributed denial of service attacks in wireless mesh networks. Comput. Math. Appl. **60**(2), 294–306 (2010). Elsevier, USA
190. S Misra, P Krishna, K Kalaiselvan, V Saritha, MS Obaidat, Learning automata-based QoS framework for cloud IaaS. IEEE Trans. Network Service Manage. **11**, 15–24 (2014). IEEE, NY USA
191. VL Kakali, PG Sarigiannidis, GI Papadimitriou, AS Pomportsis, A novel adaptive framework for wireless push systems based on distributed learning automata. Wireless Pers. Comm. **57**(4), 591–606 (2011). Springer
192. S Misra, PV Krishna, V Saritha, Lacav: an energy-efficient channel assignment mechanism for vehicular ad hoc networks. J. Supercomput. **62**(3), 1241–1262 (2012). Springer
193. Network functions virtualization (NFV), Network functions virtualization: an introduction, benefits, enablers, challenges and call for action. Introductory white paper, SDN and OpenFlow world congress, Oct 2012 (2013). http://www.tid.es/es/Documents/NFV_White_PaperV2.pdf. Accessed 12 Sept 2013
194. W John, K Pentikousis, G Agapiou, E Jacob, M Kind, A Manzalini, F Risso, D Staessens, R Steinert, C Meirosu, in *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*. Research directions in network service chaining (IEEE, NY USA, 2013), pp. 1–7
195. A Bavier, N Feamster, M Huang, L Peterson, J Rexford, VINI veritas: realistic and controlled network experimentation. Comput. Commun. Rev. **36**, 3–14 (2006). ACM, NY USA
196. M Hibler, R Ricci, L Stoller, J Duerig, S Guruprasad, T Stack, K Webb, J Lepreau, in *USENIX Annual Technical Conference*. Large-scale virtualization in the, Emulab network testbed (USENIX, 2008), pp. 113–128
197. C Elliott, GENI: Opening up new classes of experiments in global networking. IEEE Internet Comput. **14**(1), 39–42 (2010). IEEE, NY USA

198. M Casado, T Koponen, R Ramanathan, S Shenker, in *Proceedings of the Workshop on Programmable Routers for Extensible Services of Tomorrow*. Virtualizing the network forwarding plane (ACM, NY USA, 2010), p. 8
199. G Gibb, D Underhill, A Covington, T Yabe, N McKeown, in *Proc. ACM SIGCOMM Conference (Demo), Barcelona, Spain*. OpenPipes: prototyping high-speed networking systems (ACM, NY USA, 2009)
200. Bruce Davie's talk in the Open Networking Summit on NV. [Online] (2013). http://www.opennetsummit.org/pdf/2013/presentations/bruce_davie.pdf. Accessed 12 Sept 2013
201. H Wen, PK Tiwary, T Le-Ngoc, *Wireless Virtualization*, Springer Briefs in Computer Science ((Springer, 2013)
202. P Bosch, A Duminuco, F Pianese, TL Wood, in *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*. Telco clouds and virtual telco: consolidation, convergence, and beyond (IEEE, NY USA, 2011), pp. 982–988
203. S Sachs, J Baucke, in *Proceedings of the 4th Annual International Conference on Wireless Internet. ICST* (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). Virtual radio: a framework for configurable radio networks (ACM, NY USA, 2008), p. 61
204. R. Control and Provisioning of Wireless Access Points (CAPWAP) Protocol Specifications (2009). [Online] https://tools.ietf.org/rfc/rfc5415.txt. Accessed 12 Sept 2013
205. K Nakauchi, K Ishizu, H Murakami, A Nakao, H Harada, *Amphibia: a cognitive virtualization platform for end-to-end slicing*, vol. 2011. (IEEE, NY USA, 2011), pp. 1–5
206. Meraki Networks. [Online] http://meraki.cisco.com. Accessed 12 Sept 2013
207. Aruba Networks. [Online] http://cloud.arubanetworks.com/. Accessed 12 Sept 2013
208. P Papakos, L Capra, DS Rosenblum, in *Proceedings of the 9th International Workshop on Adaptive and Reflective Middleware*. Volare: context-aware adaptive cloud service discovery for mobile systems (ACM, NY USA, 2010), pp. 32–38
209. C Papagianni, A Leivadeas, S Papavassiliou, V Maglaris, C Cervelló-Pastor, A Monje, On the optimal allocation of virtual resources in cloud computing networks. IEEE Trans Comput. **62**(6), 1060–1071 (IEEE, NY USA, 2013)
210. M Satyanarayanan, P Bahl, R Caceres, N Davies, The case for VM-based cloudlets in mobile computing. IEEE Pervasive Comput. **8**(4), 14–23. IEEE, NY USA
211. F Ge, H Lin, A Khajeh, CJ Chiang, AM Eltawil, CW Bostian, W-C Feng, R Chadha, in *Military Communications Conference, MILCOM 2010*. Cognitive radio rides on the cloud (IEEE, NY USA, 2010), pp. 1448–1453
212. IGomez Miguelez, V Marojevic, A Gelonch Bosch, Resource management for software-defined radio clouds. IEEE Micro. **32**(1), 44–53 (2012). IEEE, NY USA
213. W Kurschl, W Beer, in *Proceedings of the 11th International Conference on Information Integration and Web-based Applications & Services*. Combining cloud computing and wireless sensor networks (ACM, NY USA, 2009), pp. 512–518
214. L Atzori, A Iera, G Morabito, The internet of things: a survey. Comput. Network. **54**(15), 2787–2805 (2010). Elsevier

doi:10.1186/1687-1499-2014-172
**Cite this article as:** Qadir *et al.*: **Building programmable wireless networks: *an architectural survey*.** *EURASIP Journal on Wireless Communications and Networking* 2014 **2014**:172.