Wireless Netw (2016) 22:663–677
DOI 10.1007/s11276-015-0973-5

Implementation of OpenFlow based cognitive radio network architecture: SDN&R

Suneth Namal[1] · Ijaz Ahmad[1] · Saad Saud[1] · Markku Jokinen[1] · Andrei Gurtov[2,3]

Published online: 19 June 2015

Abstract The static conventional network architecture is ill-suited to the growing management complexity and highly dynamic wireless network topologies. Software Defined Radio systems and their extension to cognitive and smart radio are characterized by distinct control loops for management which constantly increase network complexity and management inefficiencies, due to clear-cut between radio and core network management. Adding numerous devices and networks together will constantly increase the management cost, thus hinders scalability. Therefore, a holistic solution to synchronize radio and networks status has an elevated demand. To interconnect these systems and devices together, there is a need for a common management interface. OpenFlow is the first standard interface that enables Software Defined Networking (SDN). It can be rolled out in a variety of networking devices to enable improved automation and management by using common Application Program Interfaces to abstract the underlying networking details. The Software Defined Networking & Radio (SDN&R) framework proposed here has a potential combination between SDN and Radio networks to discover the underlying dynamism in cognitive access networks with integrated radio management. By isolating the control plane from the data plane, SDN&R enables a flexible management framework empowered by end-to-end goals through OpenFlow. In this article, we propose, validate, and evaluate the SDN&R architecture. In doing so, first we implement the OpenFlow enabled cognitive basestations (BSs) on Wireless Open-Access Research Platform. Furthermore, we develop software agents on BSs to provide radio status information to the cognitive control application implemented on the SDN controller. The results verify that the proposed framework in-lines with layer-2 or layer-3 forwarding performance. We claim that this work represents the first successful implementation results which synergizes SDN with Cognitive networks that motivates researchers towards SDN based radio resource management.

Keywords Cognitive networks · OpenFlow · Wireless SDN · Virtual SSID · IEEE 802.11 · WARP · 5G networks

1 Introduction

Next generation networks will constitute a major breakthrough into mobile telecommunications standards beyond the current 4G. They will adopt variety of radio access technologies empowered by different types of licensed small cells, relays, and device-to-device networks to serve users with intrinsic Quality-of-Service (QoS) requirements in a heterogeneous manner. Simultaneously, this will increase network complexity and management cost due to

✉ Suneth Namal
sunethn@gmail.com

Ijaz Ahmad
iahmad@ee.oulu.fi

Saad Saud
msaud@ee.oulu.fi

Markku Jokinen
maku@ee.oulu.fi

Andrei Gurtov
gurtov@hiit.fi

[1] Department of Communications Engineering, University of Oulu, Erkki Koiso-Kanttilan katu 3, Oulu 90570, Finland

[2] ITMO University, St. Petersburg 197101, Russia

[3] Department of Computer Science and Engineering, Helsinki Institute for Information Technology (HIIT), Aalto University, PO Box 15600, 00076 Aalto, Finland

specific requirements the wireless networks pose. Consequently, network management is handicapped by the clients of manifold needs added on daily basis.

Specially, to satisfy end-to-end requirements, operators first need a holistic view of the networks which is hampered by the conventional distributed and proprietary management architectures. Having multiple management islands for different radio access technologies can cause many problems that could be avoided with an integrated solution [38]. Bringing different systems into a single controllable domain was a challenge until the time, the concepts of programmable or adaptive networks was firstly introduced [47]. Hereby, the key performance challenge is simultaneously achieving adaptivity to changing network conditions and scalability to large numbers of users [42]. Frequent changes in a distributed control environment sometimes lead to transient overloads caused by the network having to manage traffic from multiple sources that share the same network link, resulting in congestion and instability.

Consequently, it is evident that the scalability introduces an extra overhead on network management which is estimated to be 25 % of the equipment and transmission cost [14]. This anticipates network expansions as operators will progressively cash in on management apart from capital expenditure on the network components. This cost is further incremental with the advent of cognitive radio, which needs a more versatile and flexible framework that is programmable within the timing constraints of a protocol [10]. The lack of automation capabilities makes it difficult to roll out such self-provisioned services and respond to time-sensitive changes in bandwidth requirements.

In cognitive networks, there is a clear trend to deliver advanced packet processing functions (such as virtual routing, security, QoS, etc) using dedicated packet processing blades based on commodity hardware platforms, rather than using high-priced networking equipments [2]. These functions need to be provided in a fully-virtualized environment and must offer a high degree of scalability [5]. However, the traditional networks do not offer this flexibility to realize the expected virtuality on conventional network elements [6].

Along these lines, synchronizing network status with radio status is a critical problem in cognitive networks [23]. That is because, any conflict between core and radio networks will degrade the end-to-end QoS [49]. The lack of a standard interfaces mean operators today must consolidate their Operations Support Systems (OSS)/Business Support Systems (BSS) systems to a vendor-specific network infrastructure. This requires them to either redesign their control applications for each vendor or to limit their services to a single vendor. As a result, these independent management islands shall limit network capabilities and cause many synchronization problems [49]. In particular, when core and radio networks are managed by distinct control loops, Service Level Agreements (SLAs), user profiles, and security delegations must be seamlessly synchronized. To overcome this we need a unified management system to control both core and access networks simultaneously through the same control loop.

The next breakthrough into network management was enabled with OpenFlow, which is a Software Defined Networks (SDN) component [44]. SDN advocates separating the network control plane from the data plane [13]. This migration of control, formerly tightly bound in individual network devices, into computing devices accessible through well-defined APIs. These APIs enable the underlying infrastructure to be abstracted for applications and network services [31]. Thereby, a network can be thus viewed as a logical or virtual entity. Hence, paradigm of SDN devotes a more centralized approach of network control with the holistic view [29], it effectively solves many pending problems in wireless networks [12].

SDN provides tools which enable operators to transform their networks from an undifferentiated bit-transport commodity to a value-based resource through direct linkage; therefore a greater relevance to revenue generating consumer and enterprise applications [29]. It does so with both lower CAPEX and lower OPEX, allowing more competitive pricing. As a result, SDN networks are independent of the cost model shown in [14], and may scale while lowering the management cost per device. That is because, a SDN controller has access to the forwarding elements and the statistics which can be combined to automate the radio control to effectively response the network dynamism as well as to enhance the experience on business applications and services. Furthermore, by autonomously and intelligently setting the transmission parameters (e.g., power, modulation, waveforms, resource blocks), radio networks can minimize the interference through a SDN based radio control application.

This essentially includes automating the networks by sensing contextual changes and adapting to them and applying control loop systems to learn and update itself for future actions without human intervention [1, 32]. Thereby, OpenFlow creates a standard around how the management interface or controller communicates to the equipment, therefore vendors can design their products and applications without limiting to the management requirements [33]. Thus, someone else can create a management plane knowing well that it will control the network elements. This flexibility has accelerated the innovations and faster the evolution of integrating different systems to a single control loop.

In this paper, we propose, implement, and validate the novel software defined networking and radio architecture (SDN&R). This paper provides a new dimension of radio control and management by utilizing an integrated SDN controller with an intention to motivate the researchers and industries towards SDN controlled wireless networks. Below, we summarize the benefits of proposed SDN&R framework that enables cognitive BSs understand OpenFlow protocol and perform radio resource allocation based on a controller application. By doing so, we

- provide a unified framework for radio and network resource management,
- reduce both CAPEX and OPEX per device [20],
- enable end-to-end goals such as resource allocation, guaranteed QoS,
- provide a uniform vendor-agnostic interface for global communication,
- improve spectrum efficiency in a cognitive network with collaborative decision making among access and core networks for optimized and application specific performance improvements,
- virtualize network resources for flexible management.

The rest of this paper is organized as follows: Sect. 2 describes the related work, Sect. 3 presents the SDN&R architecture, Sect. 4 describes OpenFlow implementation on Wireless Open-Access Research Platform (WARP), Sect. 5 provides the cognitive radio properties, Sect. 6 describes the results and Sect. 7 concludes the paper.

2 Related work

As the enterprise edge transitions to an all wireless network, SDN and OpenFlow are emerging as a way to bring new levels of agility to organizations beyond the data center networks, where SDN first gained traction. SDN has been gaining momentum in both the research community and industry. For example, Google has switched over to OpenFlow and SDN for its inter-datacenter networks [26], and NTT Communications has announced OpenFlow-based Infrastructure as a Service (IaaS) [8]. Similarly, many network vendors have announced OpenFlow-enabled switches as commercial products and have outlined their strategies for SDN [16, 22]. However, without bringing SDN to the edge of the network, its true promise is lost.

In [7], authors present an architecture that effectively uses the frequency spectrum combining Software Defined Radio (SDR) and SDN characteristics. This study proves that the co-existence of SDR and SDN is necessary, and the best effect can be achieved only by co-existence and mutual compliments. The results in [7] were obtained by developing a MATLAB based simulation model. The Cross Layer Controller between SDR and SDN is an important component of this architecture, as it is a decision-maker, supervisor, and administrator, and is the only component across the two layers. Amongst similar other efforts, [45] and [37] analyze the controller architectures for cognitive radio and OpenFlow enabled infrastructure. These architectures had proposed the fundamental requirements for SDN based radio controller design.

In fact, we explore how SDN can be utilized to support heterogeneous environments consisting of both infrastructure-based and infrastructure-less networks. To make the case for SDN in heterogeneous networks (HSDN) [30], we examine application scenarios in which HSDN is a key enabling technology. Over time, there has been many other studies correspond to SDN and radio integration. In [3, 48], authors present OpenRoads or “OpenFlow Wireless” solution which is developed to support heterogeneous networking, where one can move seamlessly between different radio technologies. OpenRoads incorporates multiple wireless technologies, specifically Wi-Fi and WiMAX, where Wi-Fi access points (APs) and WiMAX BSs contain flow-tables, which are controlled remotely via OpenFlow protocol. However, OpenRoads does not take-over the control of radio.

SoftRAN [15] is a software defined centralized control plane for radio access networks that abstracts all BSs in a local geographical area as a virtual big-basestation. However, this will not fit in a cognitive environment as each BS must be independently managed for optimal spectrum efficiency in a cognitive network. In [46], an initial distributed hierarchical architecture for heterogeneous radio access networks based on OpenFlow is presented. Figure 1 depicts a layered abstraction of the previously proposed solution. Accordingly, the proposal describes a mobility handover scenario of LTE femtocells and Wi-Fi. This proposal further describes an architecture for cognitive information processing for handover given with the thoughts of event-classification.

CellSDN [27] is another solution, which proposes a solution to manage the forwarding elements via local agents. The logical agents may limit the visibility of the network to some extend. In contrast, CellSDN addresses specific cellular network requirements, such as real-time session management. In [21], a cognitive control loop based on a cognitive model for efficient problem resolving and accurate decision making is proposed. Apart from the existing control loops, the proposed control loop provides reactive, deliberative and reflective loops for managing systems based on analysis of current status. This is a potential extension to the SDN enabled cognitive network architecture presented here.

In [18], a framework for a virtualization based SDN resource management platform for cognitive radio

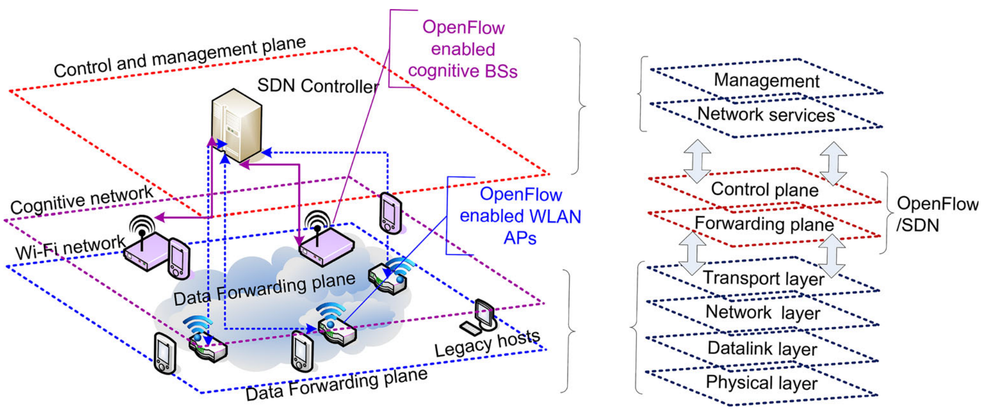


Fig. 1 Multi-tier heterogeneous software defined network architecture on the testbed

networks (CRNs) is presented. It also introduces a semi-decentralized control scheme that allows the CRN BS to delegate some of the management responsibilities to the network users. CRN resource virtualization allows dynamic, infrastructure free and efficient resources allocation to the cognitive radio users. The main objectives of the proposed framework is to improve the network performance while reducing the control overhead. In doing so, cognitive radio users reduce reliance on the CRN BSs and the remaining physical network resources. However, SDN&R architecture differs from [18], since control is centralized and SDN realization is integrated to the BSs to understand the OpenFlow protocol and to perform radio resource management based on a SDN-controller application.

The dynamism of programmable networks also introduces new security challenges that demand innovative solutions. Efficient detection and reconciliation of potentially conflicting flow rules imposed by dynamic OpenFlow applications is thus a critical challenge. Therefore, authorization and security constraint enforcement have an additive value [39]. The communications between the controller and switches are carried over a TCP connection that can optionally be protected by TLS with mutually authenticated certificates signed by a private key corresponding to a mutually trusted public key. In OpenFlow, this channel vulnerable to man-in-the-middle attacks [4]. In [36], authors present a secure channel establishment and authentication procedure. In SDN&R framework, even though we have not specified an integrated security framework, the latter solutions could be easily adapted.

3 Solution overview: SDN&R architecture

3.1 Solution architecture

In Fig. 2, we illustrates a modular representation of the proposed SDN&R network architecture. This architecture

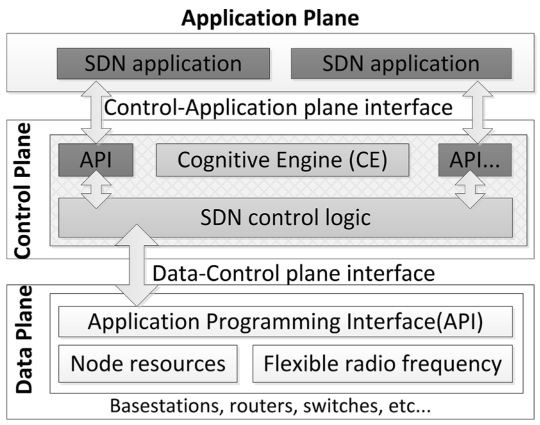


Fig. 2 SDN based cognitive network architecture: a modular representation

comprises of three layers: data plane, control plane, and application plane. The Application Programmable Interfaces (APIs) interconnect the applications and services with the control plane. The APIs in the data plane support the concept of sharing or virtualizing the underlying resources, for example, which network element ports are SDN-controlled, or the details of the virtualizing the spectrum that are exposed to the SDN applications, while isolating one customer's service from another.

In this architecture, the APIs may provide a common interface to communicate with the network controller while the SDN control logic maps and arbitrates between the networking requirements from all the SDN applications. In doing so, the controller's north bound and south bound APIs must be coded to achieve a combined goal. Thereby, the SDN&R architecture propose a well defined Cognitive Engine (CE) application for radio resource management. The CE obtains a complete view of the shared spectrum at the control layer and takeover the control of radio network parameters.

SDN&R architecture implements BS to CE communication by using a messaging protocol which uses UDP encapsulation. The cognitive process shall make decisions correspond to new radio configurations, scheduling, offloading, etc. After receiving the control messages, the network entities act according to the actions described in the messages. These messages shall follow the preconfigured flows or new flows computed by the controller. At the boot-up, the BSs have no entries on their flow-tables. Initially, the BS has no instructions on how to handle the first packet. Therefore, the first packet will be always forwarded to the SDN controller [29]. Upon receiving the packet, the controller must configure flows on switches along the path from source to destination.

3.2 Packet processing in SDN&R architecture

The SDN controller is connected to the BSs via secure channels that are used to manage the flow entries in the SDN enabled cognitive BSs. When the controller receives a "*packet_in*" message from the BS, the controller first examines the packet header and checks whether a new flow entry need to be created or the actions to be applied on the packet (Ex: receiving the first control message). If the packet requires a new entry, the controller will send a "*flow_mod*" message to the corresponding switches which then install new flows. Together with it, the controller sends a "*packet_out*" message to instruct them to send the packet out of a specific port. By utilizing these message types, the controller can add, update, and delete flows. In Fig. 3, we present the flowchart of packet processing in a cognitive BS. Below we describe each and every step shown in Fig. 3. Upon receiving the first packet, a BS:

- performs a table look-up in the first flow table and based on pipeline processing, may perform table lookups in consecutive flow tables,
- extracts the packet match fields from the packet and use them for table lookups depend on the packet type – typically includes various packet header fields, such as Ethernet source address or IPv4 address.
- In addition to packet headers, matches can also be performed against the ingress port and metadata fields. Metadata may be used to pass information between tables in a switch.
- If a corresponding flow entry is not found, such packets may be dropped or forwarded to the SDN controller as the rules defined on the BS.
- If a packet is received at the controller, the controller checks with the cognitive API for resource availability and insert the new flow entry to the corresponding BS. Then, the cognitive API acquires the unused bandwidth and updates the resource table.
- Thereby, the counters associated with the selected flow entry must be updated and the instruction set included in the selected flow entry must be applied.
- If the corresponding flow entry is found on the BS, processing is faster and the cognitive API will immediately assign resources as it is instructed by the cognitive application on the SDN controller.
- However, the cognitive algorithm may try to negotiate with the neighbouring BSs, if resources must be rescheduled for the best optimization.
- Finally, the action set will be executed.

In a nutshell, SDN&R architecture is characterized by three key attributes:

Logically centralized intelligence In SDN&R architecture, network control is isolated from forwarding using a standardized southbound interface: OpenFlow. By centralizing network intelligence, decision-making is facilitated based on a global view of the network (includes radio as well), as opposed to today's networks, which are built on an autonomous system view where nodes are unaware of the overall state of the network.

Radio resource abstraction Services are abstracted from the underlying network technologies and from the control layer to ensure portability and future-proofing of investments in network services, the network software reside in the control layer. For example, from the BS to

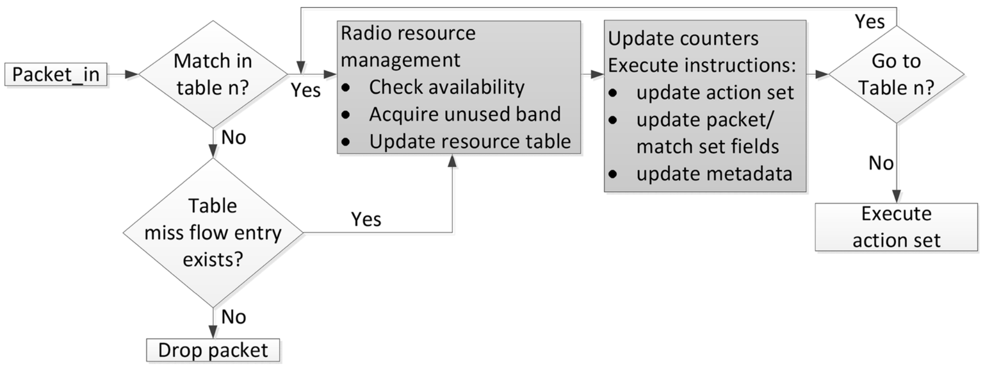


Fig. 3 Packet processing in a cognitive basestation

the enhanced packet core (EPC). Multi-tenancy allows each network slice to have a distinct policy, whether that slice is controlled by a Mobile Virtual Network Operator (MVNO), Over-The-Top (OTT) content provider, single mobile operator, virtual private enterprise network, governmental public services network, or other entity. Such services can readily be offered on a temporary basis, such as video feeds for a sporting or news event. *Wireless programmability* The programmability enables the management paradigm to be replaced by automation, influenced by rapid adoption of the clouds and radio resources. By providing open APIs for applications to interact with the wireless networks, SDN&R framework can achieve unprecedented innovation and differentiation.

4 Implementing SDN&R on WARP

This section describes the implementation of SDN&R architecture. The OpenFlow cognitive solution is implemented on WARP by modifying the reference design 17.1 [34, 41]. OFDM reference design is an integration of several generic components including MAC application, WARPMAC framework, OFDM PHY, and support peripherals. The Linux Enriched (LE) WARP proposed here modifies software system design on WARP's Field Programmable Gate Array (FPGA). LE-WARP design comprises of the novel dual core architecture where MAC layer tasks and PHY control is carried out in one processing core (PPC0) while the other processing core (PPC1) is equipped with Linux OS. Because of this, LE-WARP architecture supports for,

- flexible application layer in an open source environment,
- fully functional TCP/IP stack with IPv4 which shall be also extended for IPv6 support and,
- Ethernet Medium Access Control (MAC) on FPGA.

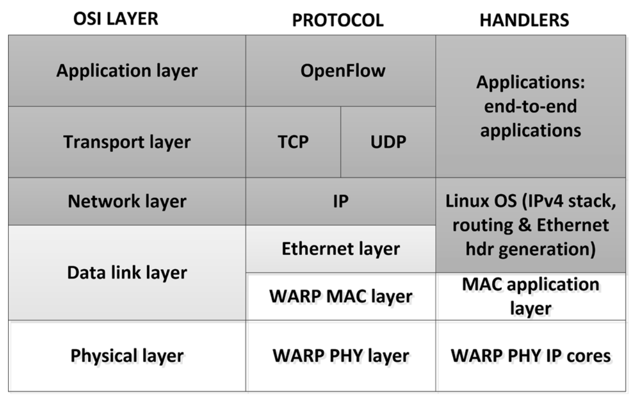


Fig. 4 Protocol stack of the OpenFlow enabled LE-WARP design

Figure 4 presents the modified protocol stack of LE-WARP. The left side is the OSI layer and the right side is the handling of the applications and systems. The dark highlighted areas indicate the newly added layers in the LE-WARP design while the slightly highlighted areas indicate the reuse of existing WARP features with slight modifications. Thus, it can be deduced that this system can handle any end-to-end application deployable on its operating system. This protocol stack also poses a need for adding a new processing core in the OFDM reference design, which is influencing the final design.

In system design perspective, as two processors (PPCs) are on the WARP FPGA, only one PPC is utilized for PHY and MAC layer handling [19]. This architecture is shown in Fig. 5. The PPC0 is thus almost similar as above referred OFDM reference design which implements real-time network stack on each WARP node. The Ethernet IP core was moved to PPC1 to ensure separate channel MAC and PHY layers for wireless and wired interfaces. In this design, PCC1 is expected to run Linux OS and OpenFlow version 1.0.0 on top of it. By running OpenFlow on PCC1, the FPGA creates a "*datapath*" that listens for connections from "*secchan*" on a Unix domain socket to serve the physical ports. Rest of IP cores related to MAC/PHY layer e.g. OFDM transmitter, automatic gain control (AGC), packet detector, radio control were left with PPC0 and the inter-processor communication was carried out using additional mailbox and shared memory.

5 Radio properties of SDN&R framework

The board is designed around a single chip direct-conversation radio transceiver. This chip modulates a carrier wave to baseband analog signal when the board is in the transmitting phase and demodulates it to the baseband signal when the board is receiving. The board also produces a Received Signal Strength Indication (RSSI) signal when it is in the receiving mode. This signal is used to detect received packets when card is controlled by WARP FPGA.

The data gathering properties of the test environment make it possible to monitor the network in real-time. Messaging protocol, developed for test environment, provides the mechanism to send and receive control messages between nodes. The nodes of the test environment can be controlled by control messages. Since a cognitive entity is able to get feedback from nodes and via control messages, it can control the behavior of the network.

Table 1 presents the physical layer properties that are supported on the test environment while the frame structure is presented in Fig. 6. The TDMA frame is segmented into two equal length parts for downlink and uplink phases. Each downlink and uplink frame is then further segmented

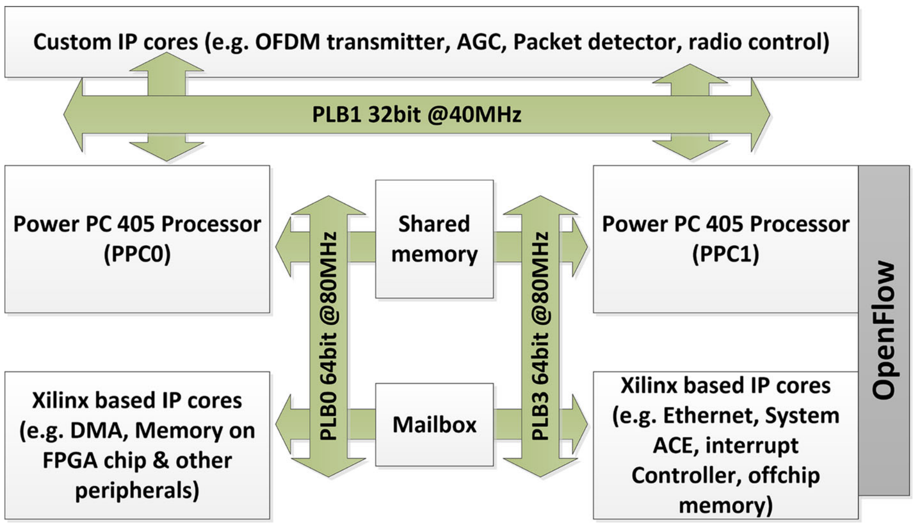

Custom IP cores (e.g. OFDM transmitter, AGC, Packet detector, radio control)
PLB1 32bit @40MHz
Power PC 405 Processor (PPC0)
Shared memory
Power PC 405 Processor (PPC1)
PLB0 64bit @80MHz
PLB3 64bit @80MHz
Xilinx based IP cores (e.g. DMA, Memory on FPGA chip & other peripherals)
Mailbox
Xilinx based IP cores (e.g. Ethernet, System ACE, interrupt Controller, offchip memory)
OpenFlow

Fig. 5 System level block diagram for OpenFlow enabled LE-WARP design

Table 1 The physical and RF properties on the wireless link between UE and BS

Physical layer properties	Fixed
Bandwidth	10 MHz
Sub-carriers	48
FFT size	64
Cyclic prefix	1.6 µs
Symbol time	8 µs

Physical layer properties	Tunable
Modulation	BPSK, QPSK, 16QAM
Coding rate	1/2, 2/3, 3/4, 1
Centre frequency	2412–2484, 5180–5805 MHz
Tx Power	−12 dBm to +18 dBm (63 steps)

into N sub-slots. The frames are divided into four different types of sub-slots, BEACON, ACK, DATA and GUARD TIME slots. During the beacon slot, synchronizing information, uplink scheduling and ACK slot scheduling is sent. When a new user joins BS, it will respond to the beacons with the ACK packet on the uplink phase and BS will allocate one uplink and downlink data slot from the next TDMA frame for it. When the user receives scheduling information that contains the allocated data slot for it, user can start transmitting to the BS on that slot.

Scheduling of the TDMA frame slots at the BS level will allocate the first free downlink and uplink slot for the joining user. After this, the scheduling information is passed to the cognitive application, which can finalize the scheduling and allocate the slots for the users according to current traffic data and user requirements.

6 Performance evaluation and validation of SDN&R

6.1 Performance evaluation of SDN&R

We evaluate SDN&R for different scenarios: Layer-2 forwarding, Layer-3 forwarding, and OpenFlow forwarding. For Layer-2 forwarding, we use Linux based Bridge-tools to set the Layer-2 forwarding of the Linux Kernel on the

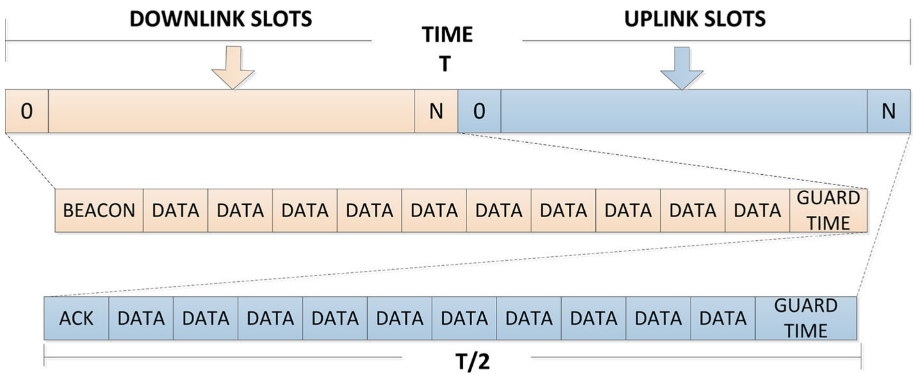


Fig. 6 Frame structure of LE-WARP MAC protocol

WARP platform; then, by using the Bridge-tools we will configure the two ports of the Cognitive BS (i.e. the Ethernet and wireless interfaces of the WARP platform) to act as a bridge and to bring up a pseudo-interface for bridging. Next, for the Layer-3 forwarding test, we use IP forwarding utility to set the Layer-3 Kernel properties. In Layer-3 forwarding test, the forwarding table is filled with C-class networks, and for OpenFlow test, OpenFlow rules are created using both different UDP source and destination port numbers. However, an extra note is needed for the Layer-2 forwarding table.

Before the Layer-2 forwarding test is started, we have to fill the Layer-2 forwarding table by sending synthetic traffic. To do this, we have to send as many different packets with different destination MAC address as of the size of our forwarding table. Layer-2 forwarding table is usually created automatically by adding entries through the backward learning algorithm: entries are removed after a timeout expiration. Thus, before starting any test, we send through the BS a proper number of packets with different source addresses by using "*pktgen*" utility in Linux. Herein, a script that generates a fixed number of packets with different source MAC addresses is used. Consequently, scripting is used to fill-up Layer-3 and OpenFlow forwarding tables too.

Then, we start comparing the three technologies for averaged processing latency as a function of the forwarding table size. On one hand, using a single flow permits to have reduced latency, because it minimizes the table lookup cost and obtains the benefits from caching techniques. On the other hand, multiple flows increase the table look-up cost and minimize the benefits of caching. For Layer-3 forwarding test, we configure the two ports of the WARP design which brings-up the interfaces to perform Layer-3 forwarding. Finally, for OpenFlow test, we use OpenFlow based "*dpctl*" to configure port-based forwarding. As expected, larger the size of the forwarding table, lower the achieved performance. Figure 7 presents the packet processing latency at the Kernel level for different sizes of forwarding tables. When the forwarding table grows by size, packets consume more time to pick the correct flow.

On one hand, since WARP platform is not optimized for bridging or switching, the Layer-2 forwarding performance compared to Layer-3 or OpenFlow forwarding is considerably low, which means packet processing latency in Layer-2 is high. On the other hand, the limited memory restricts the maximum number of flows in forwarding tables. Layer-2 forwarding has the lowest performance compared to others. Linux Kernel works on "store-and-forward" method which holds the fragments until an entire valid Ethernet frame is buffered, this is the reason for reduced Layer-2 performance of SDN&R.

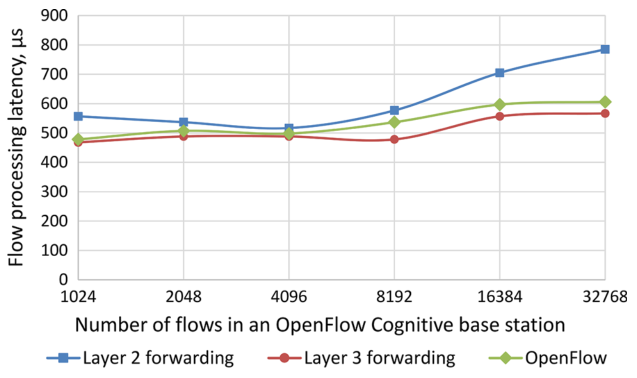


Fig. 7 Layer-2, Layer-3, and OpenFlow forwarding performance of SDN&R

In particular, combined with a large forwarding table, the packet processing takes more time. It should be noted here that the processing latency is stable in-between 1024 (2^{10})–2048 (2^{12}) flows. The packet processing latency with Layer-2 forwarding starts to grow only after this point. However, there is no significant difference with Layer-3 or OpenFlow forwarding as compared to Layer-2 forwarding, though only a slight increment of processing latency is noted with them only when there are more than 2^{12} flows.

6.2 Enhanced management efficiency

A successful network management system needs extensible, agile, efficient and scalable framework consisting of several other key elements. Amongst them, what is most important is an all-in-one solution which guarantees tight coupling between logical elements in a network. The SDN&R architecture has a well defined control and management layer that virtualize the network functions to a separate and logical control layer which provides an all-in-one solution. This architecture adds the features below that improve the overall management efficiency.

- Uniform and aligned architecture: highly automated, extensible, easily scalable architecture and an interface that hides the complexity of the underlying hardware and software.
- Unified information schema and an agile processing engine: ability to collect data across servers, extensive reporting capabilities, flexibility to export to use in other applications and data, vendor, and technology agnostic platform, that rapidly accommodates new technologies.
- Virtualized management and flexible pricing: takes advantage of virtual environments, including rapid addition of network management servers, as well as load balancing and fail-over. Dynamic, scalable license

allocation and flexible cost structure that adapts to different business models.

6.3 VoIP performance over SDN&R

The WARP platform we have used to implement the SDN&R framework provides 376 18kb RAM blocks (6.7 Mb total) on-chip, which is quite a limited memory. As a result, the flexibility to test different other applications on this platform is limited. Because of this reason, the experiments was bounded to VoIP performances. The quality of a VoIP service mostly depends on the voice codec, which is determined by the network capability and the number of users to be accommodated [9]. Thus, to improve the quality of a VoIP service, two factors should be considered: VoIP flow control and optimized codec selection. Figure 8 illustrates the testbed which interconnects the OpenFlow wireless environments to the wired core.

The validation testbed consists of NetFPGA research platforms [28], a switch which we have configured with the latest OpenFlow-enabled software version K.15.06.5080 [36] and two OpenFlow 1.3 enabled Wi-Fi access points. For testing, we used Nokia Internet tables on the Wi-Fi network. The OFDM radio access network is completely implemented on the WARP platform while utilizing the same WARP infrastructure in the client mode for evaluation. The same platform operates at the Master mode as a base-station with enabled OpenFlow support.

A BS has 16 slots that can be allocated to the uplink and the downlink. Initially, the test is performed only with one client, where whole 16 slots are assigned to a single client. In this case, a half of the bandwidth is assigned to uplink and the remaining half to the downlink (i.e. 8 slots in each direction). While adding an additional client, the bandwidth is equally and dynamically allocated among them. That is, four uplink and downlink slots on each client as shown in Fig. 9. The cognitive application equally distributes the radio resources, because the target application (i.e. VoIP) demands equal bandwidth in both directions. Thus, throughput will be dropped by half; hence, a downgrade of call quality is expected.

However, OpenFlow allows to prioritize traffic in such cases to assure VoIP QoS. Prioritizing VoIP traffic over the network yields low latency and jitter. Policy based network management, bandwidth reservation, type of service, class of service, and Multi-Protocol Label Switching (MPLS) are all widely used techniques for prioritizing VoIP traffic in traditional networks [11]. According to ITU G.114 [17], VoIP latency is characterized by the one-way latency or delay. ITU G.114 recommendation specifies it below 150 ms for guaranteed QoS although 151 ms to 300 ms one-way delay might be acceptable provided that organizations are aware that the transmission time will affect the quality [25]. However, it should be noted that one-way latency above 300 ms is not acceptable for network planning with voice services.

Furthermore, it is well known that measuring one-way latency of VoIP packets traversing though a network is not straight-forward. This is because the network architectures are based on end-to-end principles, in which the endpoints are connected to different networks independently—typically without clock synchronization. Thus, we have utilized

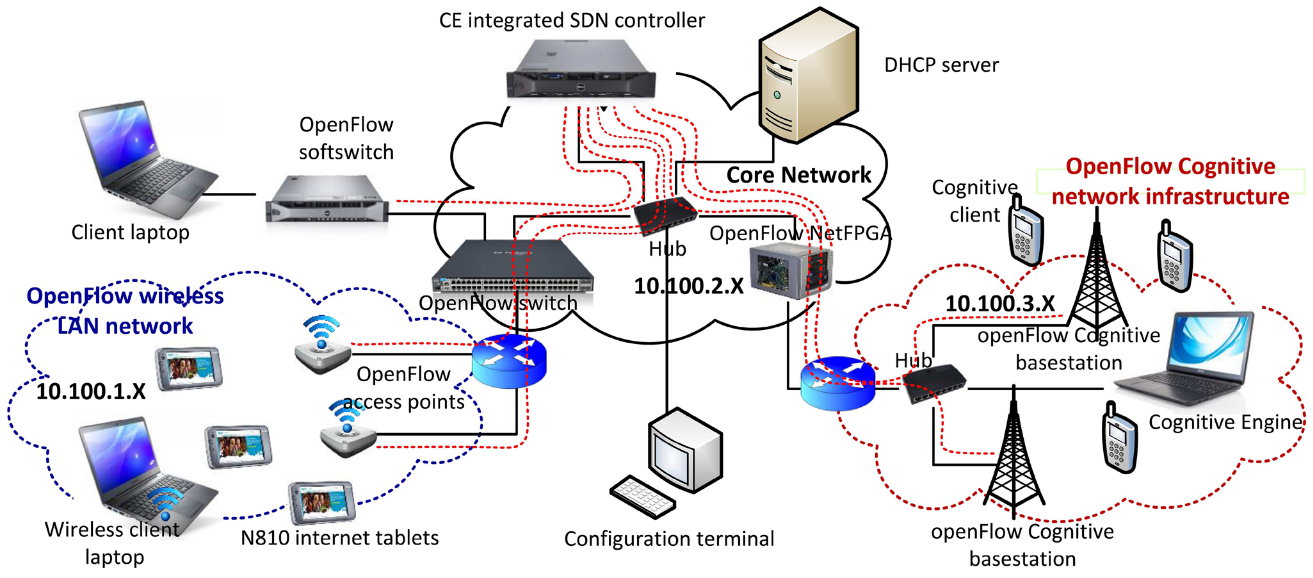


Fig. 8 Testbed implementation for VoIP measurements. Testbed consists of three sections: (1) core (LAN) segment, (2) OpenFlow WLAN segment, and (3) OpenFlow Cognitive segment. The *dotted red color lines* indicate the control channels to the controller (Color figure online)

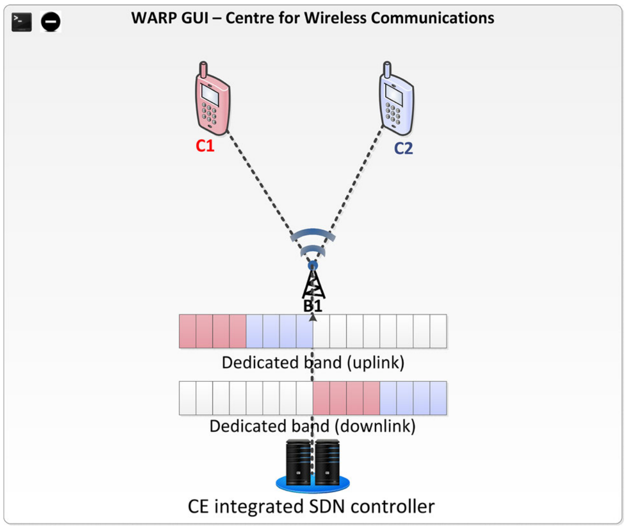

WARP GUI – Centre for Wireless Communications
C1
C2
B1
Dedicated band (uplink)
Dedicated band (downlink)
CE integrated SDN controller

Fig. 9 GUI to the cognitive engine (CE) integrated with SDN controller

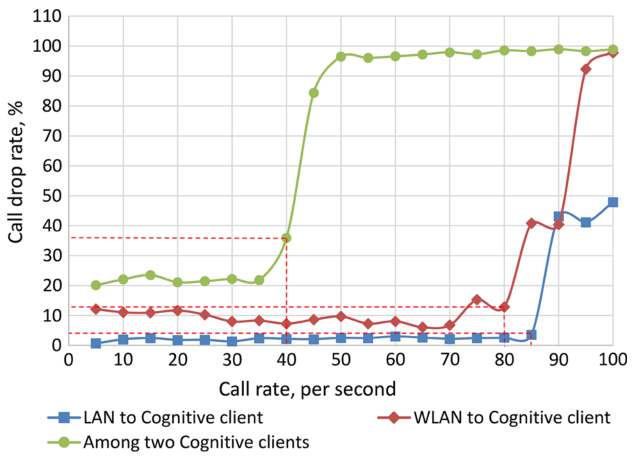


Fig. 10 Call drop rate for SIP based communication

the Round-Trip Time (RTT) to estimate the one-way latency. One-way latency is assumed to be half of the duration of RTT assuming the fact that processing delay and serialization delay is negligible compared to propagation delay. Propagation delay is the delay caused by the length a signal must traverse. Processing delay defines many different causes of delay (actual packetization, compression, and packet switching) and is caused by devices that forward the frames through the network, and finally, serialization delay is the time taken to actually place a bit or byte onto an interface or communicating medium [40].

Said that, we consider below 150 ms one-way latency for better audio quality and assume the flows exceeding this limit will be rejected by the system. At the time of initiating a call, the SDN controller configures a new flow, if a flow does not exist between the caller and the callee. By adding a modification to the controller, it monitors the round trip delay on this flow by sending an ICMP ping request. The calls exceeding the threshold are then rejected and counted as dropped calls. Based on this argument, we present the call drop rate in Fig. 10. Herein, the drop rate among different network segments is illustrated. It is important to note that the experiments are performed over 60 seconds periods for each different rate. The call duration is set to 100 ms and the standard deviation of the call duration is bounded to 30 ms.

The objective behind shortening the call duration is to load the system with as many number of flows to evaluate its capacity. Thus, the number of concurrent calls will keep increasing over a period of one minute until the line-rate is reached. Therefore, the possibility for dropping a call increases over the time from the start of the test. The measurements in this experiment were performed with SIP [43], which has similarities with the HyperText Transfer Protocol (HTTP). We have considered three scenarios,

(i) Communication between LAN and cognitive network
Call drop rate between LAN and cognitive network starts to grow at a rate of 85 calls per second (cps). It is the rate at which communication reaches to the line-rate and starts to drop calls above the delay threshold (150 ms). In this case, the latency is mostly rendered by the propagation delay between cognitive BS and its clients.

(ii) Communication between OpenFlow WLAN and cognitive networks
Call drop rate between OpenFlow WLAN and cognitive networks illustrates the impact of propagation delay. Here, the drop rate is always high compared to previous case as WLAN adds more propagation delay compared to wired networks. Thus, the threshold is reached at a rate of 80 cps. However, it should be noted here, in both cases, the BS allocates all the slots to a single client.

(iii) Communication between two cognitive networks on the same BS
When evaluating the drop rate between two cognitive clients connected to the same BS, the resources will be equally shared. The control application integrated to SDN controller does equal allocation, such that each would get only four slots in either direction. This utterly cut-down the throughput by half and thus, increases the response time. This is because, there is a high competition to acquire the limited radio resources on the BS. Therefore, at a rate of 40 cps, the call drop rate increases at an alarming rate. In fact, this

is due to the delay introduced by the resource sharing algorithm occupied in the SDN controller and the wireless propagation delay.

Each SIP transaction consists of a SIP request, and at least one response. The IP Service Level Agreements (SLAs) for VoIP call setup is measured with the total time from when an client sends a call request, to when a response from the server indicates that either the called number rang or the called party answered the call. SIP requests and responses may be generated by any SIP user agent; that is clients (UACs), which initiate requests, and servers (UASes), which response to them.

Figure 11 depicts the call response time. The mean call response time between LAN and cognitive clients below 90 cps is around 46 ms. Above this rate, the calls are highly probable to drop; thus, expected SLAs would not met. Beyond 85 cps, the same behavior is noticed between WLAN and cognitive networks. In this case, the response time is high compared to the previous case as wireless propagation delay and processing is counted in, i.e. the mean call response time is 88 ms. Since, it is well below the one-way delay threshold of 150 ms, the expected voice quality can be offered. Finally, the call response time between two cognitive clients connected to the same BS is noticed considerably high compared to previous cases due to on-air communication with OFDM radio.

Above 35 cps, the response time among two cognitive clients remarkably increases with the call rate. Below 40 cps, the mean response time is around 144 ms, which is still acceptable according to the ITU-T recommendations. At this point, the cognitive BS has shared the resources equally among two clients, which means the bandwidth is reduced by half compared to the previous cases. This happens to reach voice threshold faster.

In Fig. 12, we demonstrate the number of successful calls over a period of one minute at different rates. Herein, the SIPp module is used to generate calls at a given rate. From the start, due to call aggregation in the system the response time starts to grow over time. Therein, the number of successful calls converges with the increasing call rate. Thus, new calls will be dropped as they would not meet the expected VoIP threshold. It is important to note that the cognitive network is a bottleneck in the network due to limited bandwidth it supports. SDN&R is using 10 MHz bandwidth with Fast Fourier Transform (FFT) size of 64 and cyclic prefix of 16 samples. This means, one OFDM symbol takes 64 + 16 samples, and with 10 MHz sampling, OFDM symbol duration is 8 µs. But not all sub-carriers are used for data transmission, therefore, only 48 out of 64 sub-carriers carry data. Here, we use QPSK (2 bits/sub-carrier) modulation and coding rate one, which means to transmit (48×2) bits, it would theoretically take 8 µs; thus maximum theoretical throughput can be calculated as, (96 bits/8 µs) = 12 Mbit/s. Since, radio spectrum is equally divided on uplink and downlink, the maximum throughput is only 6 Mbit/s.

When there are two clients connected to the same BS, the maximum theoretical throughput is half of this, i.e. 3 Mbit/s. Practically, actual throughput is always below calculated. Figure 12 depicts the number of successful calls. Wireless communication reckons propagation delay which limits the number of successful calls. The wireless propagation delay on the number of successful calls is noticed by the difference between LAN to cognitive clients and WLAN to cognitive clients above 70 cps. In these two scenarios, the number of total calls generated over one minute at 70 cps should be 4200, which exactly in-lines with the experimental results shown in Fig. 12. Above 70 cps, calls will start to drop due to the limited bandwidth supported by the cognitive network. It should be noted here that BS has assigned all the slots to the cognitive client.

In doing so, BS allocates 16 slots to the primary user but none to the secondary users. The cognition process, which handles this resource allocation is performed by the

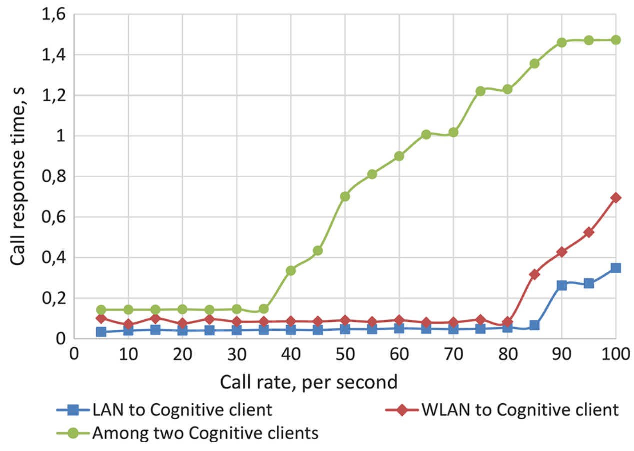


Fig. 11 Call response time for SIP based VoIP communication

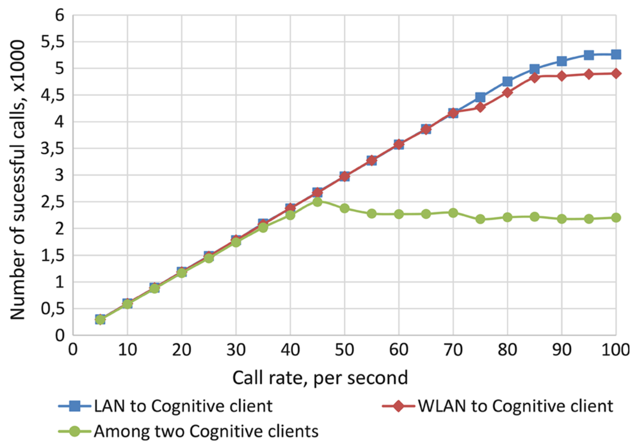


Fig. 12 Number of total successful VoIP calls over one minute

cognitive application on the SDN controller. Third graph in Fig. 12, indicates resource sharing between two cognitive clients. Theoretically, we shall expect 3 Mbps channel per client, although it comes around 2.5 Mbps on the testbed. Which means, the line-rate is reached at around 2300 calls. As a summary, it is important to note that Figs. 10, 11, and 12 comply to each other and demonstrate the correlation of results.

Finally, Fig. 13 presents the call response time characteristics as a percentage of total number of calls. In other words, it is the distribution of response time over defined periods of delay. We have chosen six call rates for the illustration that demonstrate the impact of cognition process. At a rate of 10 cps, the call response time between LAN and cognitive clients has an average of 40 ms. As shown in the graphs, 98 % of calls have a response time below 200 ms. This, in fact, meets the one-way delay constraint recommended by ITU-T. Simultaneously, between WLAN and cognitive clients, 93 % of calls have response time below 200 ms.

As per the standard, the response time above 200 ms to receive the final response, a SIP server starts to send intermediate responses [35]. This consumes an additional bandwidth for signaling while entering to a never-ending loop. Therefore, with SIP, assuring the response time below 200 ms is necessary. The communication between two cognitive clients however exceeds this limit and reduces the successful call rate.

According to Table 2, the communication between two cognitive clients enters to the mentioned loop at 40 cps, whereas both LAN to cognitive and WLAN to cognitive calls encounter this at 90 cps. The set of graphs shown in

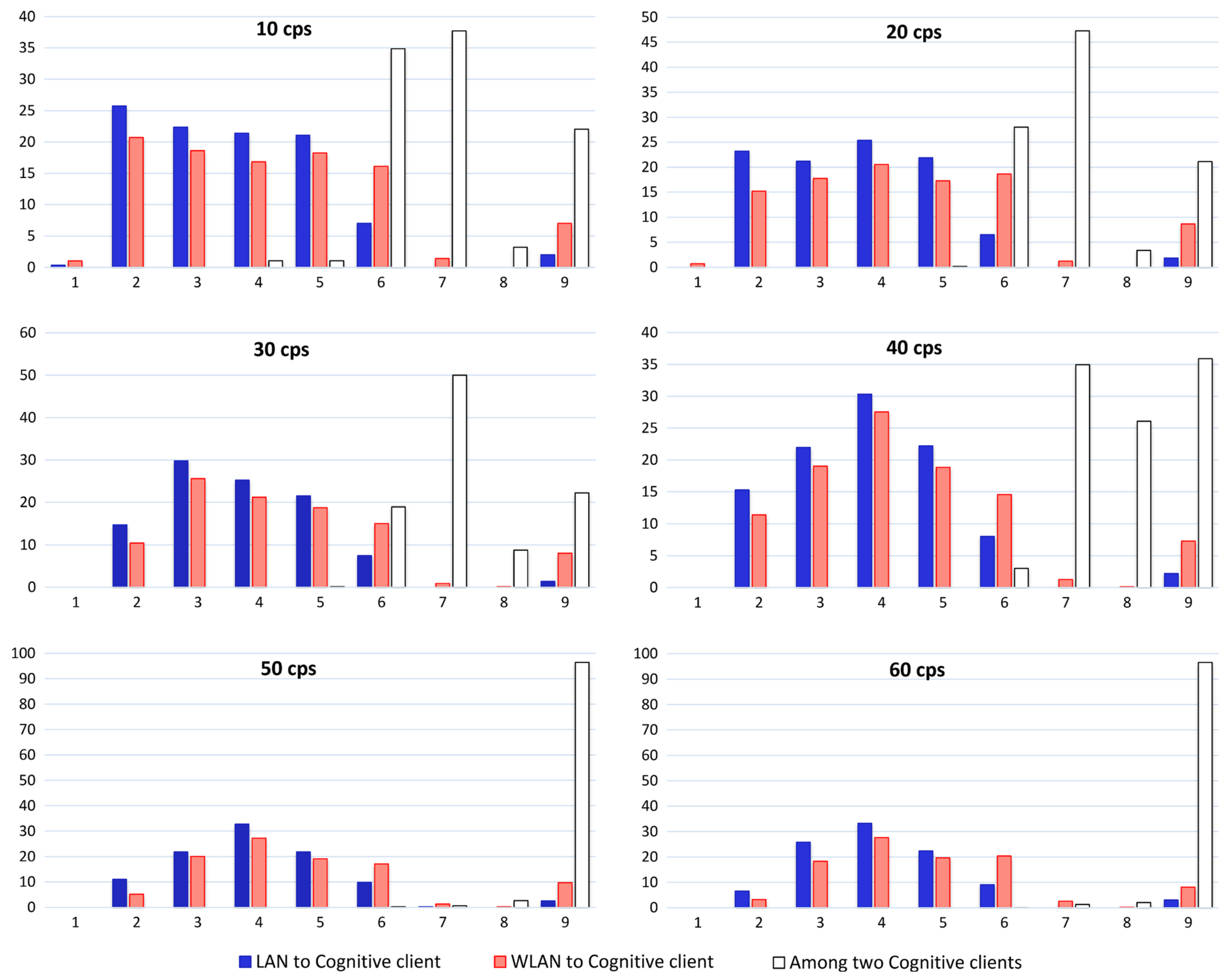


Fig. 13 Illustration of the call response time towards the cognitive network. *Vertical axis* presents number of calls as a percentage while the *horizontal axis* presents the nine different regions of response time, consecutively (1) response time <10 ms, (2) response time <20 ms, (3) response time <30 ms, (4) response time <40 ms, (5) response time <50 ms, (6) response time <100 ms, (7) response time <150 ms, (8) response time <200 ms, and (9) response time ≥200 ms

Table 2 Averaged response time between network segments

Call-rate (cps)	LAN-cognet (ms)	WLAN-cognet (ms)	Cognet-cognet (ms)
10	40.0	72.0	142.7
20	40.0	76.0	144.5
30	42.0	84.0	145.3
40	44.0	86.0	335.0
50	47.0	90.0	701.0
60	51.0	91.0	903.0
70	53.0	91.9	1018.0
80	59.0	93.2	1230.0
90	262.0	427.0	1460.0
100	348.0	695.0	1473.0

Fig. 13 specifically cover-up the VoIP characteristics of pure-cognitive communication, which in fact, highly affected by wireless propagation delay, cognitive processing delay, and queuing delay due to limited buffer in BSs and clients. However, it should be noted here that the call response time does not significantly change in any of the cases, before entering to the loop although it significantly changes after that.

7 Conclusion

In this article, we have proposed, validated, and evaluated the novel SDN&R framework that unifies radio resource management with SDN. This is the first SDN footprint for unified management of cognitive radio and core network resources/status. We have designed and implemented OpenFlow enabled cognitive BSs on top of WARP with OFDM reference design. Thereby, we made the BSs understand the OpenFlow protocol and perform radio resource allocation through an automated SDN application. By splitting the control and forwarding planes, we simplify the existing complexity in the cognitive networks and introduce an scalable framework that lower both CAPEX and OPEX. Below, we summarize our contributions in point form.

- As a proof-of-concept, we prototyped OpenFlow enabled cognitive BSs on WARP FPGA with OFDMA reference design,
- Proposed architecture, cognitive algorithms, and API agents to dynamically control the radio resources on BSs,
- Performance evaluation of SDN&R and SDN enabled cognitive base stations,
- Centralized control/management for cognitive, WLAN and core network though the integrated SDN controller and the applications developed on top of it.

Our prototype processes more than 2000 packets per second below 2^{13} flows on a BS, which means its forwarding capacity is around 30 MBps. For validation, we limit our experiments for VoIP due to the constraints enforced by the hardware platform. Below is the successful call rate in-between different domains in the network.

- 98 % of successful call rate between two cognitive clients in the same SDN-cognitive domain is achieved below 45 cps.
- Below 70 cps, the same level of performance is noticed between SDN-cognitive and OpenFlow Wi-Fi domains.
- Finally, the complaint successful call rate between SDN-cognitive network and core (LAN) is achieved below 85 cps.

This findings demonstrate the impact of wireless propagation delay and the nature of resource allocation through the SDN control APIs implemented on the cognitive BSs. The results in this paper successfully validate the SDN&R framework, and we believe this study will motivate researchers in the direction of SDN towards radio networks.

References

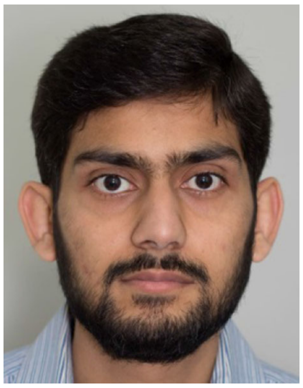
1. Akyildiz, I. F., Lee, W. Y., Vuran, M. C., & Mohanty, S. (2008). A survey on spectrum management in cognitive radio networks. *IEEE Communications Magazine*, *46*(4), 4048.
2. Anwer, M. B., Motiwala, M., Tariq, M., & Feamster, N. (2011). Switchblade: A platform for rapid deployment of network protocols on programmable hardware. *ACM SIGCOMM Computer Communication Review*, *41*(4), 183194.
3. Bansal, M., Mehlman, J., Katti, S., & Levis, P. (2012). Openradio: A programmable wireless dataplane. In *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, pp. 109114.
4. Benton, K., Camp, L. J., & Small, C. (2013). Openflow vulnerability assessment. In *Proceedings of the second ACM SIGCOMM workshop on hot topics in software defined networking*. ACM, pp. 151152.

5. Berl, A., Gelenbe, E., Di Girolamo, M., Giuliani, G., De Meer, H., Dang, M. Q., et al. (2010). Energy-efficient cloud computing. *The Computer Journal*, *53*(7), 10451051.
6. Chowdhury, N. M. K., & Boutaba, R. (2009). Network virtualization: State of the art and research challenges. *IEEE Communications Magazine*, *47*(7), 2026.
7. Cho, H.-H., Lai, C.-F., Shih, T. K., & Chao, H.-C. (2014). Integration of SDR and SDN for 5G. *IEEE Access*, *2*, 1196–1204.
8. Desmond, P. (2012). NTT Com announces first carrier-based OpenFlow service. June 13, 2012, http://www.nttcom.tv/2012/06/13/ntt-com-announces-first-carrier-based-openflow-service/sthash.4zGqn7tx.dpuf.
9. Ding, L., & Goubran, R.A. (2003). Speech quality prediction in VoIP using the extended E-model. In *Proceedings of IEEE global telecommunications conference (GLOBECOM'03)*. IEEE, vol. 7, pp. 39743978.
10. Dutta, A., Saha, D., Grunwald, D., & Sicker, D. (2010). An architecture for software defined cognitive radio. In *Proceedings of ACM/IEEE symposium on architectures for networking and communications systems (ANCS)*, pp. 112.
11. El-Gendy, M. A., Bose, A., & Shin, K. G. (2003). Evolution of the Internet QoS and support for soft real-time applications. *Proceedings of the IEEE*, *91*(7), 10861104.
12. Feng, T., Bi, J., & Hu, H. (2012). TUNOS: A novel SDN-oriented networking operating system. In *Proceedings of 20th IEEE international conference on network protocols (ICNP)*. IEEE, pp. 12.
13. Fortino, G., Di Fatta, G., Pathan, M., & Vasilakos, A. V. (2014). Cloud-assisted body area networks: State-of-the-art and future challenges. *Wireless Networks*, *20*(7), 1–14.
14. Garbin, D. A. (1998). Toward a national data network: Architectural issues and the role of government. The unpredictable certainty: White papers, p. 217.
15. Gudipati, A., Perry, D., Li, L.E., & Katti, S. (2013). SoftRAN: Software defined radio access network. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*. ACM, pp. 25–30.
16. Hoelzle, U. (2012). Openflow@google. 17th Open Networking Summit, pp. 1–10.
17. ITU-T. (2003). Series G: Transmission systems and media—one way transmission time, Digital systems and networks, International telephone connections and circuits—General Recommendations on the transmission quality for an entire international telephone connection—ITU-T Recommendation G. 114, p. 1.
18. Jararweh, Y., Ayyoub, M. A., Doulat, A., Aziz, A., Salameh, H. A. B., & Khreishah, A. A. (2014). SD-CRN: Software defined cognitive radio network framework. In *Proceedings of IEEE international conference on cloud engineering (IC2E)*. IEEE, pp. 592–597.
19. Jokinen, M., & Tuomivaara, H. (2011). LE-WARP: Linux enriched design for wireless open-access research platform. In *Proceedings of the 4th international conference on cognitive radio and advanced spectrum management*. ACM, p. 16.
20. Khan, A., Kellerer, W., Kozu, K., & Yabusaki, M. (2011). Network sharing in the next mobile network: TCO reduction, management flexibility, and operational independence. *IEEE Communications Magazine, 49*(10), 134–142.
21. Kim, S., Kang, J. M., Seo, S., & Hong, J. W. K. (2013). A cognitive model-based approach for autonomic fault management in OpenFlow networks. *International Journal of Network Management*, *23*(6), 383–401.
22. Kobayashi, M., Seetharaman, S., Parulkar, G., Appenzeller, G., Little, J., Van Reijendam, J., et al. (2014). Maturing of OpenFlow and software-defined networking through deployments. *Computer Networks*, *61*, 151–175.
23. Kondareddy, Y. R., & Agrawal, P. (2008). Synchronized MAC protocol for multi-hop cognitive radio networks. In *IEEE International Conference on Communications (ICC'08)*. IEEE, pp. 3198–3202.
24. Koponen, T., Casado, M., Gude, N., Stribling, J., Poutievski, L., Zhu, M., et al. (2010). Onix: A distributed control platform for large-scale production networks. In: *OSDI* (Vol. 10, ppp. 1–6).
25. Kwon, T. T., Gerla, M., & Das, S. (2002). Mobility management for VoIP service: Mobile IP vs SIP. *IEEE Wireless Communications*, *9*(5), 66–75.
26. Levy, S. (2012). Going with the flow: Google's secret switch to the next wave of networking. Wired, April 17, 2012, http://www.wired.com/wiredenterprise/2012/04/going-with-the-flow-google/.
27. Li, L. E., Mao, Z. M., & Rexford, J. (2012). CellSDN: Software-defined cellular networks. Technical report, Princeton University.
28. Lockwood, J. W., McKeown, N., Watson, G., Gibb, G., Hartke, P., Naous, J., et al. (2007). NetFPGA-An open platform for gigabit-rate network switching and routing. In *IEEE International Conference on Microelectronic Systems Education (MSE'07)*. IEEE, pp. 160–161.
29. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., et al. (2008). OpenFlow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, *38*(2), 69–74.
30. Mendonca, M., Astuto, B. N., Obraczka, K., Turletti, T., et al. (2013). Software defined networking for heterogeneous networks. *IEEE MMTC E-Letters*, *8*(3), 36–39.
31. Mendonca, M., Obraczka, K., & Turletti, T. (2012). The case for software-defined networking in heterogeneous networked environments. In *Proceedings of the 2012 ACM conference on CoNEXT student workshop*. ACM, pp. 59–60.
32. Mitola, J. (2000). *Cognitive radio—An integrated agent architecture for software defined radio* (pp. 1–2). Royal Institute of Technology (KTH).
33. Mitola, J., & Maguire, G. Q, Jr. (1999). Cognitive radio: Making software radios more personal. *IEEE Personal Communications*, *6*(4), 13–18.
34. Murphy, P., Sabharwal, A., & Aazhang, B. (2006). Design of WARP: A wireless open-access research platform. In *Proceedings of European signal processing conference*, pp. 1804–1824.
35. Nahum, E. M., Tracey, J., & Wright, C. P. (2007). Evaluating SIP server performance. In *Proceedings of ACM SIGMETRICS performance evaluation review*. ACM, Vol. 35, pp. 349–350.
36. Namal, S., Ahmad, I., Gurtov, A., & Ylianttila, M. (2013). Enabling secure mobility with openflow. In *Proceedings of IEEE SDN for future networks and services (SDN4FNS)*. IEEE, pp. 1–5.
37. Namal, S., Ahmad, I., Jokinen, M., Gurtov, A., & Ylianttila, M. (2014). SDN core for mobility between cognitive radio and 802.11 networks. In: *Proceedings of 8th international conference on next generation mobile apps, services and technologies (NGMAST)*. IEEE, pp. 272–281.
38. Pavon, J. (1996). Towards integration of service and network management in TINA. *Journal of Network and Systems Management*, *4*(3), 299–318.
39. Porras, P., Shin, S., Yegneswaran, V., Fong, M., Tyson, M., & Gu, G. (2012). A security enforcement kernel for openflow networks. In *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, pp. 121–126.
40. Ramaswamy, R., Weng, N., & Wolf, T. (2004). Characterizing network processing delay. In: *Proceedings of IEEE global telecommunications conference GLOBECOM'04*. IEEE, Vol. 3, pp. 1629–1634.
41. Rice University WARP Project: http://warpproject.org/.
42. Rodriguez, A., Kostic, D., & Vahdat, A. (2004). Scalability in adaptive multi-metric overlays. In *Proceedings of 24th international conference on distributed computing systems*. IEEE, pp. 112–121.

43. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., et al. (2002). SIP: Session initiation protocol. Tech. rep., RFC 3261, Internet Engineering Task Force.
44. Sezer, S., Scott-Hayward, S., Chouhan, P. K., Fraser, B., Lake, D., Finnegan, J., et al. (2013). Are we ready for SDN? Implementation challenges for software-defined networks. *IEEE Communications Magazine*, *51*(7), 36–43.
45. Sun, G., Liu, G., & Wang, Y. (2014). SDN architecture for cognitive radio networks. In *Proceedings of 1st international workshop on cognitive cellular systems (CCS)*, pp. 1–5.
46. Sun, G., Liu, G., Zhang, H., & Tan, W. (2013). Architecture on mobility management in OpenFlow-based radio access networks. In Proceedings of IEEE global high tech congress on electronics (GHTCE). IEEE, pp. 88–92.
47. Sutton, R. S., & Barto, A. G. (1981). Toward a modern theory of adaptive networks: Expectation and prediction. *Psychological Review*, *88*(2), 135.
48. Yap, K. K., Huang, T. Y., Kobayashi, M., Chan, M., Sherwood, R., Parulkar, G., et al. (2009). Lossless handover with n-casting between WiFi–WiMAX on OpenRoads. *ACM Mobicom (Demo)*, *12*(3), 40–52.
49. Zander, J. (1997). Radio resource management in future wireless networks: Requirements and limitations. *IEEE Communications Magazine*, *35*(8), 30–36.

Suneth Namal received the B.Sc. degree in computer engineering from the University of Peradeniya, Peradeniya, Sri Lanka in 2007. He completed M.E. degree in Information and Communication Technologies from the Asian Institute of Technology, Bangkok, Thailand, in 2010 and M.Sc. degree in Communication Network and Services from Telecom Sud-Paris, Paris, France, in 2010. Currently he is a Doctoral student with the Department of Communication Engineering, and a project manager at University of Oulu, Oulu, Finland. His research interest includes Software Defined Networks (SDN), OpenFlow, network security, mobile femtocells, fast initial authentication, load balancing and mobility management, Wi-Fi protocols. Currently, he is a senior lecturer at the Department of Computer Engineering, University of Peradeniya, Sri Lanka.

Ijaz Ahmad received his B.Sc. degree in Computer Systems Engineering from University of Engineering and Technology (UET), Peshawar, Pakistan in 2008. After working in the telecommunication industry, he moved to Finland to obtain his M.Sc. (Technology) degree of Wireless Communications Engineering with major in Telecommunication Engineering from University of Oulu, Finland in 2012. Currently he is a Doctoral student with the Department of Communications Engineering, University of Oulu, and a research scientist in Centre for Wireless Communications (CWC), Oulu, Finland. His research interest includes network architectures, software defined mobile networks, network security, and network load balancing.



Saad Saud received his Master's Degree in Electrical Engineering from University of Oulu in 2014. Currently he is a Doctoral Student in Department of Communications Engineering, University of Oulu.

Markku Jokinen graduated as M.Sc. in Electronics from the University of Oulu, Finland in 2010 and he is working as a Research Scientist at Centre for Wireless Communications and is working on his doctoral studies at the Department of Communications Engineering. His research focus is on the field of wireless communications, especially on the practical implementations of algorithms and protocols. He has gained experience on the design, implementation, and demonstrations of the various algorithms and protocols of PHY/MAC layers on wireless test beds.

Andrei Gurtov Andrei Gurtov received his M.Sc (2000) and Ph.D. (2004) degrees in Computer Science from the University of Helsinki, Finland. He is presently a Principal Scientist at the Helsinki Institute for Information Technology HIIT. He is also adjunct professor at Aalto University, University of Helsinki and University of Oulu. He was a Professor at University of Oulu in the area of Wireless Internet in 2010-12. Previously, he worked at TeliaSonera, Ericsson NomadicLab, and University of Helsinki. He was a visiting scholar at the International Computer Science Institute (ICSI), Berkeley in 2003, 2005 and 2013. Dr. Gurtov is a co-author of over 150 publications including three books, research papers, patents, and five IETF RFCs. He is a senior member of IEEE.