A SURVEY ON WIRELESS SOFTWARE DEFINED NETWORKS - WSDN

Postgraduate programme " Technoloogy Education & Digital Systems"

Digital Communications and networks

Mourtikas Theofanis, University of Piraeus

Supervisor : Prof. Panagiotis Demestichas

ACKNOWLEDGMENTS

I would like to thank my advisor, Konstantinos Tsagkaris for guiding and supporting me as well as Mr. Prof. Panagiotis Demestichas for all the support that he committed to me.

I would like especially to thanks my family for all the support, inspiration and confidence that all this time gave to me. I could not forget my friend Patras Xaris, who encouraging me to complete my goals .

ABSTRACT

In this thesis, we present and analyze the software defined network architecture as a new concept that addresses the emerging need of new network architecture derived from diversification of traffic patterns, emergence of personal IT devices, cloud computing, and big data. We present the limitations of existing networks against these new challenges and provide detailed information about the Openflow protocol that may address these limitations. Based on the detailed examination of the Openflow protocol we examine the literature relative to advancements in application of Openflow to wireless networks. Wireless Software Defined Networks (Wireless SDN) use cases are examined relative to wireless network type, namely WLAN, WiMAX, cellular networks, mesh and wireless sensor networks. Each network type is thoroughly examined in terms of current limitations and the opportunities presented by SDN in wireless topologies are addressed. Next several proposals are presented, according to the literature, relative to the future of wireless network in the era of SDN.

CONTENTS

1 INTRODUCTION

Software Defined Networks (SDN) are a new approach to network architecture, which employs separate and directly programmable network control, undependable of data forwarding procedure. Therefore, the derived network architecture is highly dynamic, manageable, cost-effective and workable, which gives operators the option of scheduling, automation and control (Azodolmolky, 2013).

The idea of Software Defined Networks has even been introduced as a way to facilitate their development. Since control is now detached from the operation of data forwarding and has been transferred to central controllers, the networking infrastructure can be differentiated in relation to the application. Moreover, the architecture of Software Defined Networks provides a set of Application Programming Interfaces (APIs), which greatly simplify the integration of common web applications such as, routing, multicasting, security, access control, spectrum management, traffic engineering, QoS - Quality of Service, and various energy efficiency policy management schemes. In Software Defined Networks, computing is logically centralized to controllers software based at the control level and also the networking devices are converted to simple packet forwarding devices on the data layer) which can be programmed through an open interface (Zhang & Zhao, 2014).

Wireless communication is the transfer of information between two or more points that are not connected to an electrical conductor (Ilcev, 2006). The most common wireless technologies use radio spectrum. The radio distances can be small, from a few meters for television or thousands or even millions of kilometers for radio communications in space. It includes various types of fixed, mobile and portable applications, including two-way radios, mobile phones, personal digital assistants (PDA), and wireless networking. Other examples of wireless radio technology applications include GPS units, garage door openers, wireless computer mice, keyboards and headsets, earphones, radio receivers, satellite television, television broadcasting and wireless phones (Tse and Viswanath, 2005). Wireless functions allow services such as long range communications that are impossible or impractical to implement using cables. The term is commonly used in the telecommunications industry and relates to telecommunications systems (e.g., radio transmitters and receivers, transmitters, etc.), which use some form of energy (e.g., radio waves, acoustic energy, etc.) to transfer information without the use of cables. The transfer of information is thus both small and large distances (Tse and Viswanath, 2005).

As presented above, mobile and wireless networks still exhibit a huge amount of diversification and development. While, wireless becomes the first or maybe sole access technique for a lot of individuals, mobile operators should support higher volumes of traffic and more sophisticated services. Specifically, video-based services account for a bigger part of the traffic, requiring important demands for new network functions like media transcoding and content caching. Moreover, mobile networks should at the same time support many generations of mobile services (i.e., 3G and 4G) beside a spread of user services such as VoIP, streaming media, and messaging, leading to wide varied traffic properties. They as well do this by a cost-efficient manner in an environment of quickly increasing capability demands that way exceed the expansion in revenues—and within the budgets needed to handle the new demands. Increase in traffic has made more and more cells in the radio access network (RAN) necessary to provide the mobile subscribers with an access to wireless networks. Thus, mobile operators turn to technology of small cells in order to grow capacity by reusing frequencies mainly in high dense populated areas. Limited distance between the physical cells is one of the factors that increase the interference between cells, reinforcing higher bandwidth in 4G LTE services (ONF, 2013).

Adding up to the challenges, there is an increasing number of wireless technologies that are utilized at the same time. Typical devices today support 3G, 4G, as well as WiFi connectivity and Bluetooth. Such a variety of wireless technologies require mobile operators to maintain and operate remote access and backhaul and core networks, increasing both costs and management complexities. In addition, providers need flexible implementation options to migrate from older to newer technologies without affecting the customer experience. Another important change is the need to provide services across technologies, since mobile providers should strengthen policies of high complexity to secure proper access to the right service and transfer control between the access types. Also, the ever-changing business needs require quick implementation of new mobile services and quick adoption of new technologies (ONF, 2013).

It is argued that SDN may provide increasing benefits to wireless networks addressing many of its limitations due to the suitability of flow structure for end to end communication, the logical centralization of control to address interference, optimization of path management and network virtualization (ONF, 2013).

2 SOFTWARE DEFINED NETWORKING TECHNOLOGY

2.1 THE EMERGING NEED FOR A NEW NETWORK ARCHITECTURE

The blast of cell phones, server virtualization, and the new approach of cloud service provisioning are among the patterns that drive networking industry to reconsider customary system designs. Numerous traditional networking systems operate in a hierarchical manner, developed with levels of Ethernet switches orchestrated in a tree structure. Although this configuration was sufficient when client-server architecture was predominant, this exact static engineering is not suitable to the dynamic operation and capacity needs of the current business data centers, universities, and carrier situations. Following, an analysis is presented of the key factors that drive the requirement for another networking approach (ONF, 2012):

2.1.1 Diversification of traffic patterns

Due to server virtualization challenges that have emerged in recent years, the traffic patterns inside business data centers have changed fundamentally. The traditional focus for client-server applications, realized the majority of the traffic between one client and one server, yet today's applications access diverse databases and servers, realizing a significant amount of "east-west" machine-to-machine activity before returning information to the end client by "north-south" traffic design. As presented in Figure 2-1, a significant IT evolution of today’s networks is the shift from fat client to thin client. Fat client computing is the typical client-server communication, where transaction of data is realized among user and server application in customer and public networks, evoking north-south traffic as depicted by the relative arrow. Thin client computing allows interaction of traffic data among applications servers and a single request of data transaction triggers large communication traffic between servers. The fat to thin client shift is realized through the shift from traditional desktop computers to smartphones and tables or other smart devices, hence east-west communication is increasing, presenting tremendous surges. The shift to east-west traffic is driven by cloud computing incorporating pay as you go IT resources, big data that are collected by sensors in machine to machine designs, mobile and distributed users, virtualization, converged infrastructure and integrated systems (ONF, 2012).

There are various server-to-server utilities and applications, which empower and drive the colossal increment in east-west transactions. One such application is storage replication, a basic part of most organization's recuperation procedures in case of damage or disaster. Shockingly, disk array and NAS filer volumes are extending quickly and have achieved the point where sizes of 500 TB to 1 PB are turning out to be progressively normal. Storage replication activity is described by a moderately little number of flows, yet high throughput per flow (Karasiewicz, 2014).

Virtual machine (VM) migration is the one application that drives east-west activity increase, which is turning out to be progressively regular due to the far reaching selection of virtualization and distributed computing. In the process of VM migrationrunning in sizes if 10 Gigabytes or more of dynamic memory, the execution condition of a virtual machine is transmitted over a fast system starting with one physical server then onto the next. In VM migration among datacenters, an asynchronous replication or WAN access of the virtual machine disk space may take place. However, another methodology is to utilize synchronous replication among data centers, which permits the information to dwell at both areas and to be effectively accessed by VMs at both destinations (Karasiewicz, 2014).

Hence, clients and network users are changing transaction designs as they push for access to business content and applications from any smart device, at any time, from any place. The concept of cloud services is emerging, driving data center managers to take into account a utility computing model that may incorporate a private cloud, public cloud, or some blend of both, bringing about extra activity over the wide area network (ONF, 2012).

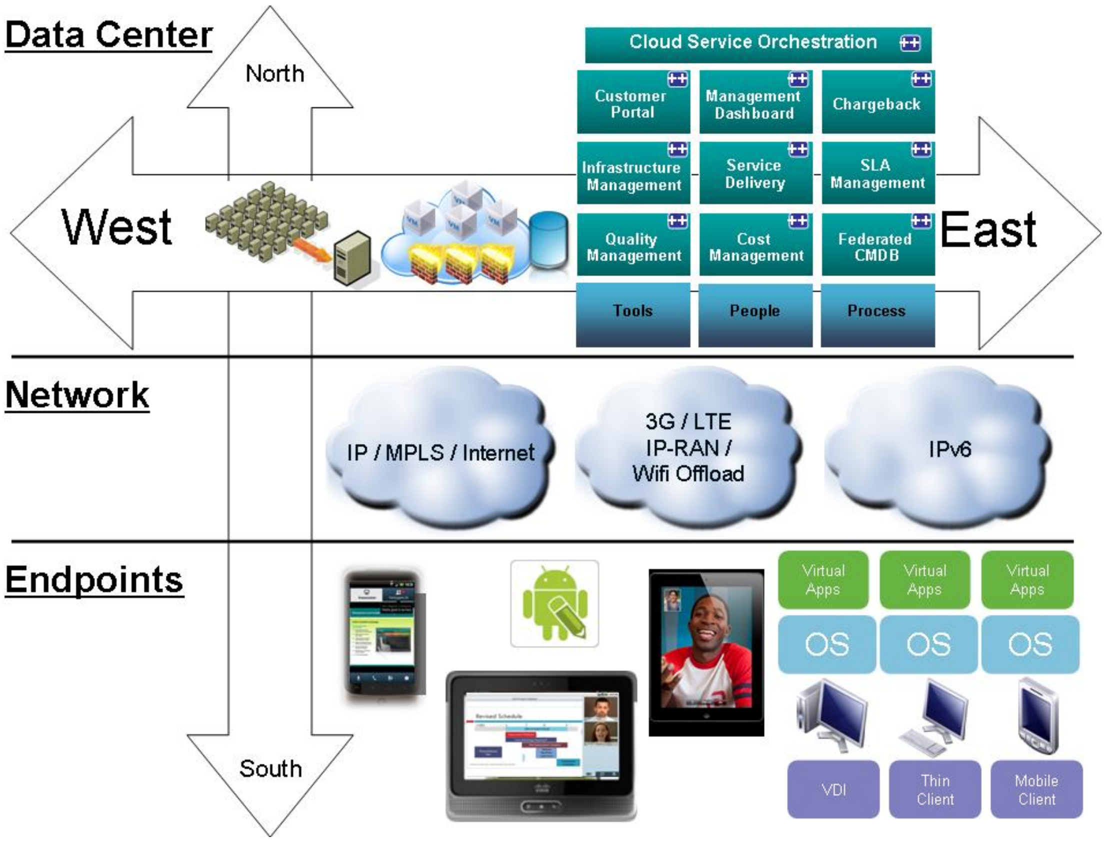


Figure 2-1 East-West and North-South data traffic in Data Centers (Karasiewicz, 2014)

2.1.2 Emergence of personal IT devices

The annual Samsung Techonomic Indexthat was announced in 2015, was derived from a thorough analysis relative to the ways that Europeans interact with technology. The study involved 18 European countries, revealing usage habits and expenditure in many electronic devices.The new EU-wide study reveals that the home of 2015 has been transformed in a working node, for study and entertainment, where people connect with friends and family. IT technology enhances this experience as the average Greek household appliances correspond to 19, indicating the path to the era of smart homes (ONF, 2012).

In a European level, 17% of households are paying in regularly for access to educational content, five times more than in 2014 (3%). One in four Europeans (25%)

consume educational content on a weekly basis - an increase of 3% compared to 2014. Hence, technology enables Europeans to open their horizons in education. Households spend more on educational materials (€ 12 / month) than in mobile games (€ 10) and recreational materials such as books and magazines (€ 11), movies (€ 10) and music (€ 9). In a typical day, 85% of European adults spend at least one hour for consumption of educational content. Meanwhile, adherence to mobile games, in contrast, is being limited. Only 33% of Europeans admit they play a mobile game on at least a weekly basis - a significant decrease compared with 42% in 2014. Almost 24% of households in Europe have paid to stream content (movies and TV shows) last month, four times more than just 5% in 2014. One in five Europeans (21%) pays for digital versions - twice as many as in 2014 (9%). 75% of Europeans now participate in social networking activities - increase from 70% in 2014 (Samsung, 2015).

Over the past 12 months there has been a significant increase in the use of 4G mobile technology and an increase in the use of hardware, which is connected to the internet, such as Smart TVs and tablets. The increasing use of fastest internet facilitates the expansion of knowledge, new skills learning, exploration of art in different cultures, all from the comfort of our home (Samsung, 2015).

Households across Europe are searching online to meet their desire for learning and discovering new things. More than half of Europeans (54%) make regular video streaming, an increase of 50% compared to last year. Digital books and other publications remain popular. Also, the continued dominance of social networks indicates that the life within the household is more socially networked than before. The possibilities of Internet of Things will increase as the use of hardware devices connected to the internet continues to evolve (Samsung, 2015).

Hence, the smart devices, such as mobile smartphones, tablets and notebooks access corporate networks, pressuring for accommodation, protection of data and intellectual property as well as facilitation of compliance to guidelines and standards (ONF, 2012).

2.1.3 The era of cloud computing

Relative to the scientific community, there is no single definition of cloud computing. The term “cloud computing”has been aspired by computer network diagrams that represent the Internet as a cloud. The main players of the IT market have issued manuals that attempt to determine the meaning of cloud computing, approaching to a common definition, which covers the commonly agreed aspects of cloud computing. The NIST working definition summarizes the cloud, as: “a model that allows for easy, on-demand web access to a shared set of regulated computing resources (eg, networks, servers, storage, applications and services) and may be delivered quickly and rendered with minimal management effort or interaction with the service provider” (Mell, 2009). Another definition given by Armbrust et al, (2009), describes cloud computing as a model that includes applications distributed as web services, and hardware and software of systemic datacenters that provide those services. The cloud is the set of resources allocated as a follow demand. Cloud computing proposes new ways to deliver services. Cloud computing makes it possible to use virtual resources through the Internet, on demand.However, the technology of cloud computing is still at an early stage and as recognized by Mell (2009), the definition of NIST and each definition is likely to evolve in the future as new developments are explored in the cloud. The definition of NIST (Mell, 2009) describes the cloud, through five key characteristics, three service models, and four development models. The key features are:

- On-demand self-service: the computational resources can be acquired and used at any time without the need for human interaction with cloud services providers. The computational resources include the processing power, storage, virtual machines, etc.
- Broad network access: the above resources can be accessed over a network using heterogeneous devices, such as laptops or mobile phones.
- Concentration of resources: the cloud services providers pool their resources, which are then shared by many users. This is referred to as multi-tenancy,

where for example, a physical server can host many virtual machines that belong to different users.

- Rapid elasticity: the user can quickly obtain more resources from the cloud through expansion. The resources can be brought back to the original, with the release of additional resources once they are no longer needed.
- Measurable services: the use of resources measured using appropriate metrics as usage of storage space monitoring, CPU hours, bandwidth usage etc.

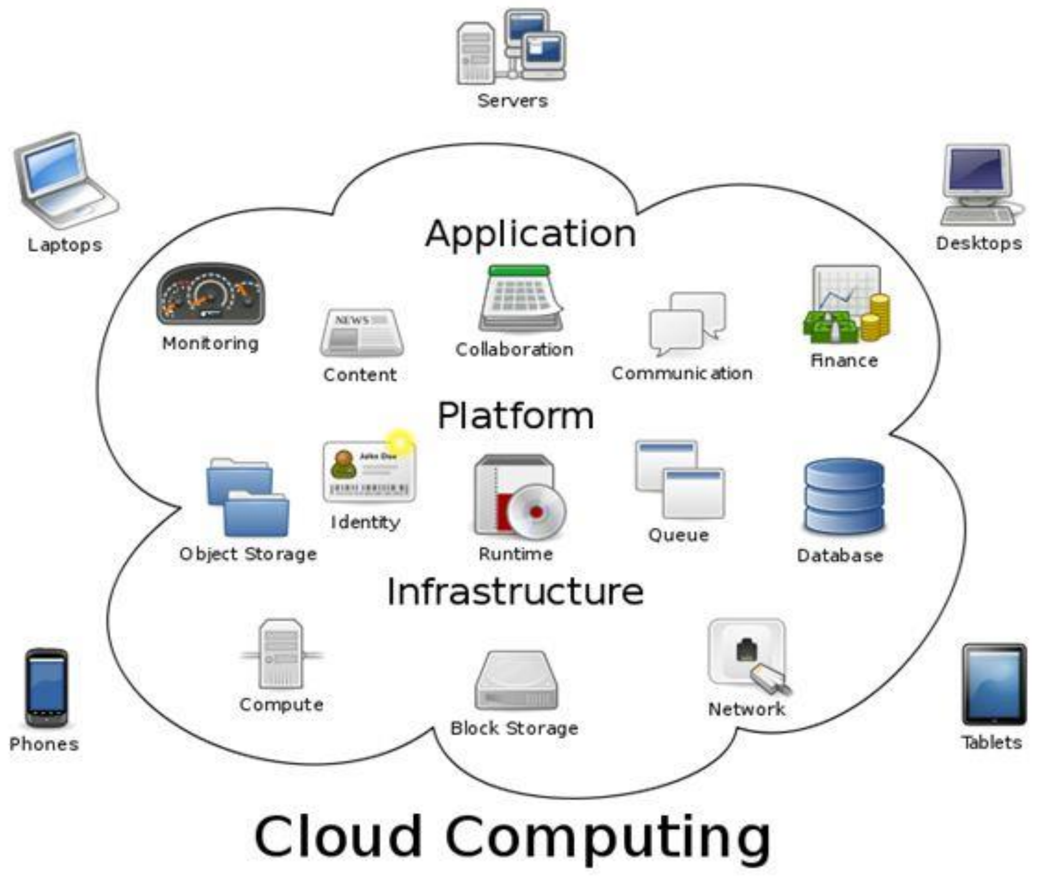


Figure 2-2 Cloud computing architecture (Scudder, 2011)

The dynamic features of cloud computing have been widely accepted by enterprises that utilize private and public clouds with the growing demand of having access to applications and resources on demand in a pay as you go scheme. In addition, security and privacy issues, as well as compliance with various technologies, business requirements etc. can enhance the system’s complexity. The provision of self-service mode in either private or public clouds means that elastic scaling of features such as storage and other network resources is demanded, with a common tool suite.

2.1.4 The rise of “big data” concept

The emergenceof digital sensors, telecommunications, computation and storage has created very large datasets that are created from data gathered from multiple sources of human activity such as homes, enterprises, governments, educational and research institutes. Enterprises such asGoogle, Yahoo! and Microsoft have taken this opportunity to create new business by collecting information that is freely available on the Web and provide services to various users or clients. The amount of data that is collected is so large and the service’s range. Examples include satellite imagery and navigation services, but the whole range of services presents many benefits, affecting the way information is gathered and utilized (ONF, 2012).

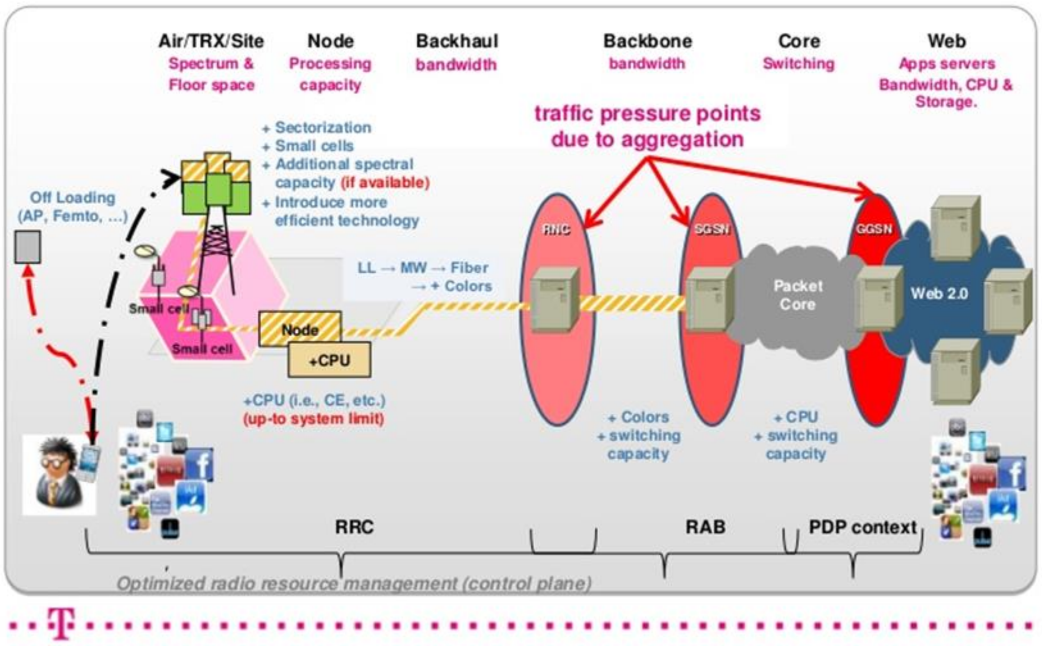
The term Big Data has been introduced since 2005 for the purpose of defining the“great amount of data that traditional data management techniques cannot manage and process due to the complexity and size of this data” (Ularu et al., 2012).

Big data are defined by IBM as having four aspects (Ularu et al., 2012):

- Volume, referring to the quantity of data gathered by a company, utilized to gain important knowledge,
- Velocity: this aspect takes into account the processing time of Big Data. Processing time is differentiated relative to activities that are very important, requiring immediate response and maximum efficiency and others that can be processed slower,
- Variety, as an aspect, includes the different types of data that can be included in the Big Data concept, structured or unstructured.

- Veracity, referring to decision making based on the information derived from Big Data correlations and processing. It accounts for the level of trust of decision makers on Big Data information.

The management of Big Data today (or mega datasets as otherwise referred to) requires massive parallel processing from multiple servers, which need to be directly connected to one another (Figure 2-3). Hence, Big Data enhances the demand of increased network capacity of data centers and therefore hyperscale data center operators are faced with the task of capacity scaling to extreme levels without allowing any connections among entities to break (Larsen, 2012).



Air/TRX/Site
Spectrum & Floor space
Node
Processing capacity
Backhaul
bandwidth
Backbone
bandwidth
Core
Switching
Web
Apps servers Bandwidth, CPU & Storage.
traffic pressure points due to aggregation
+ Sectorization
+ Small cells
+ Additional spectral capacity (if available)
+ Introduce more efficient technology
Off Loading (AP, Femto, ...)
LL → MW → Fiber → + Colors
RNC
SGSN
GGSN
Packet Core
Web 2.0
Small cell
Small cell
Node
+CPU
+CPU (i.e., CE, etc.) (up-to system limit)
+ Colors + switching capacity
+ CPU + switching capacity
RRC
RAB
PDP context
Optimized radio resource management (control plane)

Figure 2-3 Today's bit pipe and bottlenecks (Larsen, 2012).

2.2 Existing network technologies and their limitations

The inclusion of today’s market requirements in every day IT practice seems impossible for traditional network architectures. Business IT sector has decreased budgets and therefore they try to gain the most out of current architectures by utilizing management at device-level and other manual processes. The demand of increased bandwidth and mobility is emerging and therefore carriers are faced with several challenges that, combined with limited profits, lead to reduced revenues. The fact is that existing network technologies were not designed to comply with today’s trends and the consequent requirements of business, users, and carriers. In detail the main limitations of existing network technologies are as follows (ONF, 2012):

Network complexity leading to stasis: Several discrete sets of protocols are utilized today for connecting every host over arbitrary distances, speeds and topologies with reliability. The protocols of existing network architectures have been evolved in the past decade in order to accommodate the requirements of businesses and technology and offer high reliability, performance, connectivity and security. As mentioned above, protocols are discrete and solve a specific issue, defined in isolation from the other topologies that it need to connect the hosts. Therefore, the number one limitation in today’s networks is complexity, derived from the fact that a multitude of discrete protocols need to be communicating with each other in an environment that performance and reliability are very significant. This is evident by the fact that in order to move or add any device the operator has to pass a multitude of switches, routers and firewalls, authentication portals in order to update mechanisms that are based on protocols by utilizing tools of device management. In order to do so, the operator has also to consider the topology of the network, switch models, and versions of software. This has resulted in the stasis of existing networks in order to minimize the risk of service disruption.

Server virtualization that has been projected to the new service environment, is on contrast to the static nature of existing network architectures. This is due to the fact that server virtualization has added up to the number of hosts that need to connect to the network as well as the physical location of hosts. Before virtualization all applications were located on a single server and communicated with specific clients. Nos, application can be distributed in several virtual machines, exchanging data among them. The reason for virtual machine migration is the optimization of server workloads, hence the physical location of endpoints are changing and challenge existing networks in a multitude of aspects, such as addressing schemes and namespaces.

Additionally, the combination of virtualization with IP converged networks for data, voice and video ads up to complexity. Quality of Service is very important in any network topology and existing architectures have the ability to provide diversified QoS levels relative to the application, but in a manual way of resource provision. This means that the operator needs to adjust QoS parameters by hand relative to vendor's equipment and that the network is not able of dynamic adaptation to variations is application, user demands and traffic.

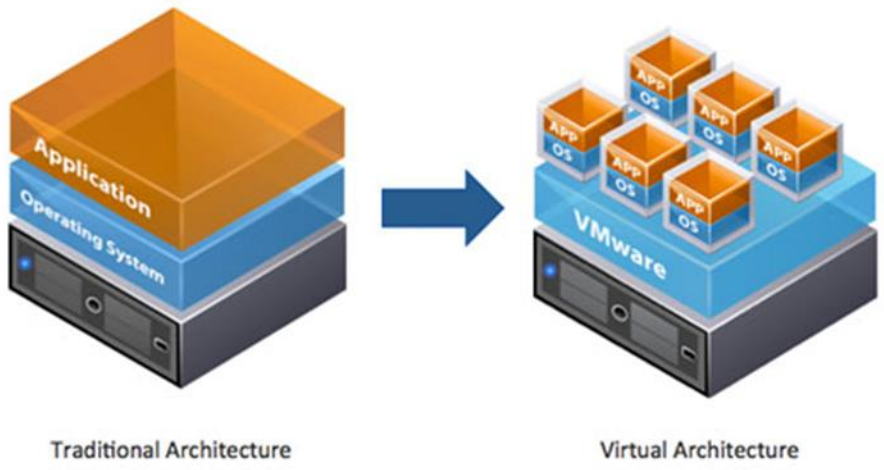


Figure 2-4 The definition of virtualization in schematic form (Cryptum, 2016)

Policies inconsistency: The operator has to reconfigure thousands of devices and mechanisms to apply a policy to the whole network. Having introduced virtualization, each time a new virtual machine appears, reconfiguration of access control lists across the entire network may take hours or even days to implement. It is very difficult with existing networks to apply consistent policies for security and quality of service that are very important in allowing access to mobile users without increasing the risks of breaches and non-compliance.

Scaling inability: Network development must follow demands on datacenters, nevertheless, network complexity has increased with the addition of a multitude of devices that need to be managed and configures, relying on linkoversubscription to scale the network, based on predictable traffic patterns. Virtualization has transformed traffic patterns to a more dynamic and unpredictable phenomenon. Enterprises such as Google, Yahoo!, and Facebook, are against major scaling challenges due to the employment of large - scalealgorithms and datasets that are processed in parallel across theirentire computing pool. Due to the ongoing increase of user application demands such as entire web crawling and indexing to provide user search results, computing elements are expanding dramatically while data traffic may reach petabytes.Hyperscale networks have been employed but such enterprises in order to provide forhigh-performance, low-cost connectivity among the multitude of physical servers. It is evident that the level of such scaling is not applicable by manually configurations of the network. The operation is further complicated by adding the demand of multi-tenancy, in the frame of providing services to users with various applications and QoS needs. Many important operations, that in the new era are key to the dynamic nature of the new environment, for example traffic flow steering for customized performance control or on-demand delivery, may seem very complex to existing network architectures, since they need special devices that raise the cost for the vendor and the time to commercialization.

Vendor dependence:The ever changing needs in a business and end user aspect are the goal of business and carriers to respond to, by employing the new network capabilities and services. Nevertheless, the lack of standardization for vendor’s equipment, lack of open interfaces and the fact that the life cycle of this equipment can range to 3 years or more, makes these applications vendor dependent, limiting the ability of network operators to make tailored network configurations.The mismatch that has occurred among the requirements of customers and enterprises and the capability of existing network architectures has challenged the industry, which in response, developedthe architecture of Software-Defined Networking (SDN) and the relevant standards.

2.3 THE SDN CONCEPT

As mentioned above, cloud computing and virtualization as well as high data rate video services, are the main factors that lead to the requirement of high bandwidth, for the applications for data centers. These new environment consists of mainly ultra-high wavelength applications at speeds higher than 100 Gbps. Software defined networks using the OpenFlow protocol, operating as a centralized control architecture, are being increasingly popular due to the accommodation of software defined network functions and protocols. Software defined network functions provide enhanced flexibility to users and create a unified control over various network resources to optimize operations and services (Zhang& Zhao, 2014).

2.4 SDN PRINCIPLES

The basic SDN principles are the following (Rajasri et al., 2011):

- The Control and Data planes are separate from each other
- The Control plane software may be executed or run on general purpose hardware, utilizing commodity servers and decoupled form specific networking hardware.
- Data planes are programmable, and can be managed, controlled and programmed form a central entity.

- The architecture is adequately developed to be able to control not just a networking device but the whole network.

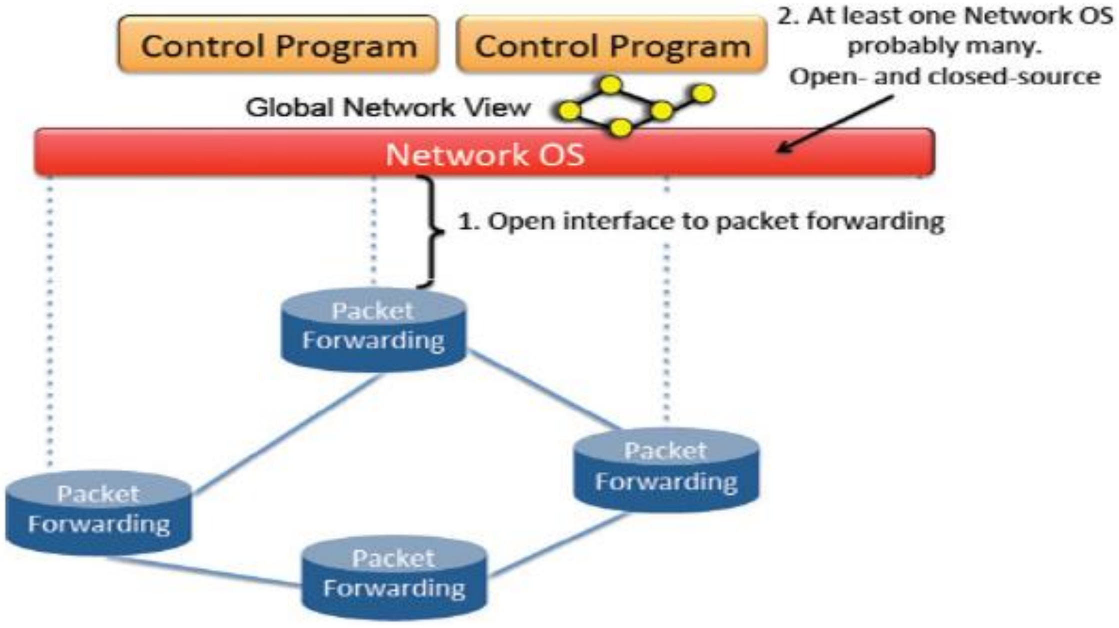


Figure 2-5 Software defined networks basic principles (Rajasri et al., 2011).

As is described in Figure 2-5, in Software defined Networks, there should be at least one Network OS (operating system), while there are probably going to be many different operating systems. The packet forwarding is realized through an open interface.

2.5 SDN ARCHITECTURE

SDN architecture is greatly diversified from the architecture of existing conventional networks. As depicted in Figure 2-6, SDN architecture consists of three layers, the Infrastructure layer, the Control layer and the Application layer. The infrastructure layeris the layer where all network devices communicate with each other, such as switches. The supervision of this communication is employed by the control layer, which communicates with the infrastructure layer through a control-data interface, such as OpenFlow (Figure 2-6). Finally, at the higher layer, the one called application layer, the implementation on all the utilized applications takes place. The application layer communicates with the control layer via APIs(Rajasri et al., 2011).

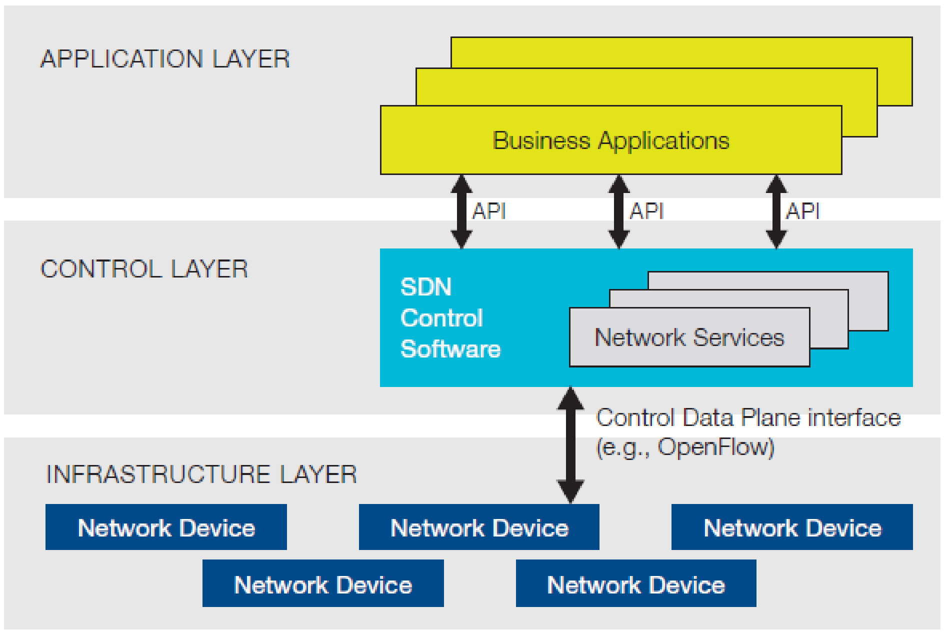


Figure 2-6 Software defined network architecture (ONF, 2012).

A logical structure of SDN architecture is also presented by Stallings (2012), as depicted here in Figure 2-7. In the presentation of Stallings (2012), there are three planes: data plane, control plane and application plane. The data plane incorporated all the different devices that may be employed in the network, such as routers, LAN switches, packet switches and other network devices. The data plane is respective to the infrastructure layer presented in Figure 2-6. The control plane, which communicates with the data plane via Openflow, incorporates functions such as routing, traffic engineering and mobility. The control plane communicates via APIs with the application plane which facilitates SDN applications, business applications and cloud orchestration.

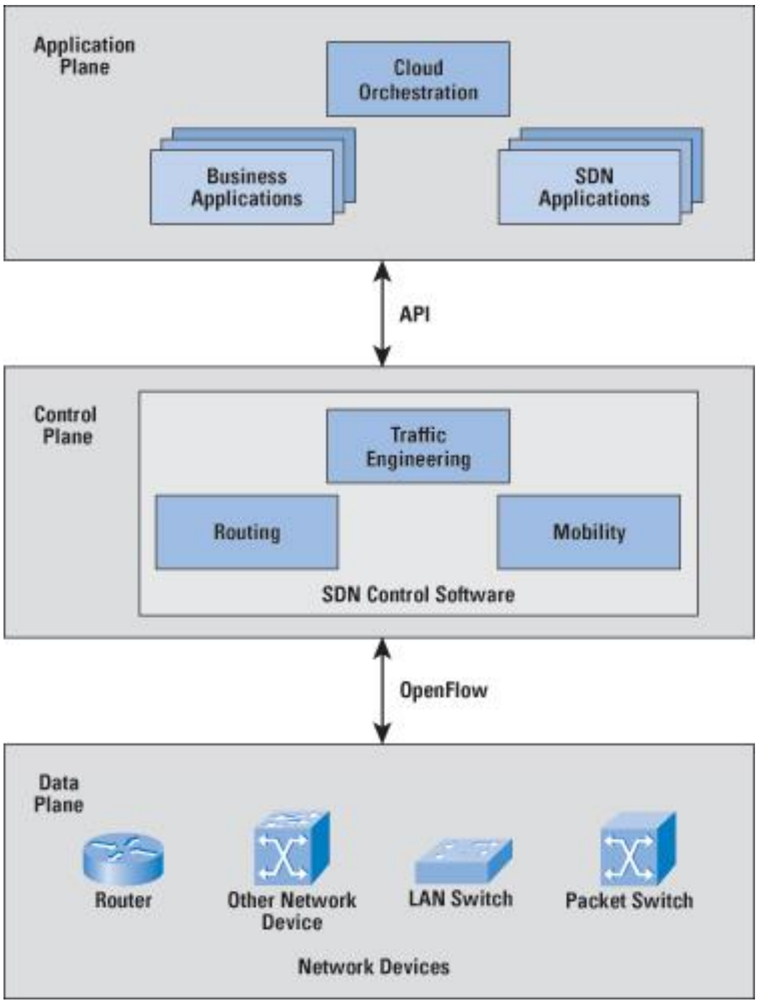


Figure 2-7 SDN logical structure (Stallings, 2012)

According to Stallings (2012), in the SDN architecture all complex functions of the control plane are performed by a central controller. The control plane may consist of one or more SDN servers. The controller of the SDN architecture has the ability to define the data flows that are created in the data plane. The controller has the task of giving permission to each data flow in order to verify that the communication is permissible under the network policy. In the case of the controller allowing the data flow, a dedicated route is computed and entries are added to all switches for that respective flow. Therefore, all the complex functions are realized by the controller, and the switches have mainly the task to manage flow tables, created by the controller. Also, networks devices are simplified since they no longer need to manage and comprehend the multitude of various protocols, but only take instructions from the controller (ONF, 2012). This is referred to as network abstraction (ONF, 2012).

This architecture is very flexible, since it has the ability to operate with various types of switches and protocol layers. The basic principle of SDN architecture is packet forwarding based on some kind of flow definition. According to Stallings (2012), controllers and switches may be developed for Ethernet switches, internet routers, transport switching or switching and routing of the application layer. Due to the architecture of SDNs, network operators can apply various configurations programmatically, without having to manually code each path of communication from and towards the various devices. Network behavior can be altered in real time, incorporating new services, applications and devices in a matter of hours. Resource optimization is also centralized by the nature of controller operations. Since operators can write tailored programs to load to the controller, they are not any more dependent on the vendor's hardware or closed software environments (Rajasri et al., 2011).

In addition to network abstraction, a set of APIs is supported in order to realize common service such as bandwidth management, access control and security. Hence, the operator can define and enforce consistent policies to the entire network (ONF, 2012). Open APIs, studied by ONF (2012), are said to "to promote multi-vendor management,which opens the door for on-demand resource allocation, self-serviceprovisioning, truly virtualized networking, and secure cloud services" (ONF, 2012). In this way, the applications may be operated on an abstracted network, without the need to being tied in implementation details, while efficiently maximize the benefits from network services and capabilities (Rajasri et al., 2011).

According to Stallings (2012), a switch in SDN logical structure may perform the following tasks:

- Encapsulation and forwarding the first packet of a data flow to the controller in order to be decided by the controller if the flow is permissible or not. If the flow is permissible, then it is added to a flow table included to the switch.

- Forwarding of incoming packets to the suitable port, according to flow table. The controller may also have included priority information to the flow table located to the switch.
- According to the decisions of the controller, the switch drops packet for a while or permanently. The operation is called packet dropping and is linked to security protocols, Denial-of-Service (DoS) attacks or traffic management requirements.

All the switches forwarding state is managed by the controller via vendor-neutral APIs that enable compliance with a multitude of operator requirements, without the need to alter lower-level network structure or topology (Rajasri et al., 2011).

Due to the decoupling among data and control, software defined networks allow for applications "to deal with a single abstracted network device without concern for the details of how the device operates" (Stallings, 2012). Hence, what applications "see" is only a single API towards the controller, nevertheless creation and deployment of new applications is possible in order to manage traffic flow according to demands for security or performance.

2.6 SDN DOMAINS

As depicted in Figure 2-8, the concept of a single controller to manage all network devices, in large-scale networks, is not good practice. For the future wide adoption of SDNs the most possible scenario is to implement several sub networks according to SDN topology by breaking down the entire network into smaller non-overlapping parts. This would be implemented by the operator in order to be able to manage the network properly. This smaller network parts are called SDN domains (Stallings, 2012).

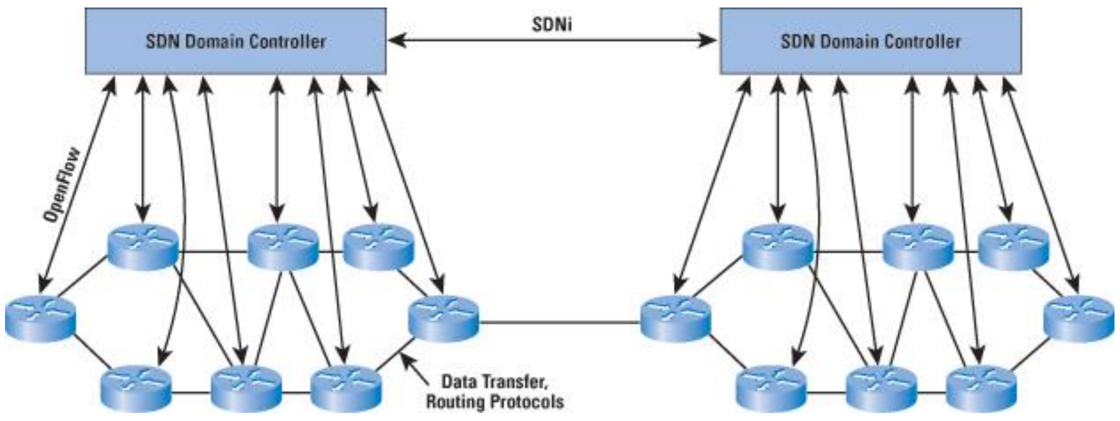


Figure 2-8 SDN domain structure (Stallings, 2012)

There are various reasons why a network operator may utilize SDN domains (Stallings, 2012):

- Scalability: There are limits to the number of physical devices that an SDN controller can manage while maintaining performance and availability standards. Hence, larger networks may need to employ several controllers connected to different groups of devices.
- Privacy: Various privacy policies may be implemented in different SDN domains. As a paradigm, dedication of an SDN domain to customers deploying tailored privacy and security policies, having as a requirement to not share the same domain with other entities.
- Incremental arrangement: The system of a network operator system may comprise of parts of conventional and newer devices. Separating the system into different, independently managable SDN areas takes into account adaptable incremental deployment.

In the presence of numerous domains, the necessity of communication among individual controllers by means of standardized protocol relative to routing data, is emerged. Currently the SDNi protocol is being developed, for the purpose of interfacing SDN domain controllers (Stallings, 2012).

The functions included in the SDNi protocol comprise mainly the following:

- Coordination of flow setup that came from applications containing data, for example, QoS and path requirement through various SDN domains.
- Allowing for routing among SDN domains by exchanging reachability information, by permitting to a single data flow access various SDN domains and each controller decides for the best path, in the case of multiple available paths.

According to Stallings, (2012), there the following message types included in the SDNi tentatively:

- Reachability update
- Flow setup/tear-down/update request
- Capability update

3 OPENFLOW PROTOCOL

3.1 THE PROTOCOL

OpenFlow protocol plays an important part in SDN architecture as presented in the previous chapter, by enabling the separation of the data plane (infrastructure layer) from the control plane. Through OpenFlow, the exploitation of SDN characteristics, such as the fact that most Ethernet switches incorporate flow tables required to implement services, for example, Network Address Translation (NAT), Quality of Service (QoS) and Firewall, can be implemented. Flow tables are developed using multiple Ternary Content Addressable Memories (TCAMs), containing access-lists to filter packets based on the MAC address, and QoS access-lists for the priority of network traffic. The OpenFlow protocol provides the ability to program these flow-tables, via the management of each OpenFlow switch by the network operator through an Openflowcontroller. A key feature of the Controller is the fact that it can add or remove data flows (flows) in the flow-table of the OpenFlow switch. By utilizing FlowVisor isolated network resources (slices) can be created, each controlled by a single controller (Kerner, 2012).

The requirements of common logical structure among all devices and secure network standard for switch-controller communication, are addressed via the OpenFlow protocol, which has the role of both a protocol for controller and devices and a specification addressing the logical structure of operations of the network switches (Limoncelli, 2012). OpenFlow is defined by the Open Networking Foundation (ONF, 2012), which is a“consortium of software providers, content delivery networks, and networking equipment vendors whose purpose is to promote software-defined networking” (Stallings, 2012).

3.2 OpenFlow switch

An OpenFlow switch consists of a flow-table, which is used for mapping and packet forwarding, and a secure communication channel to a Controller. The OpenFlow switch is handled by the Openflow controller through the secure channel using the OpenFlow protocol (ONF, 2012).

The flow-table consists of entries for flows, counters and actions for each record. Each packet that arrives the OpenFlow switch is checked against the records of the flow-table. In the case it matches with any record, the proper functions are implemented, else the packet is forwarded on the Controller through the secure channel. The Controller is now authoritative to reach a decision for the route that the packet will follow, adding or removing entries from the flow-table of the switch (Limoncelli, 2012).

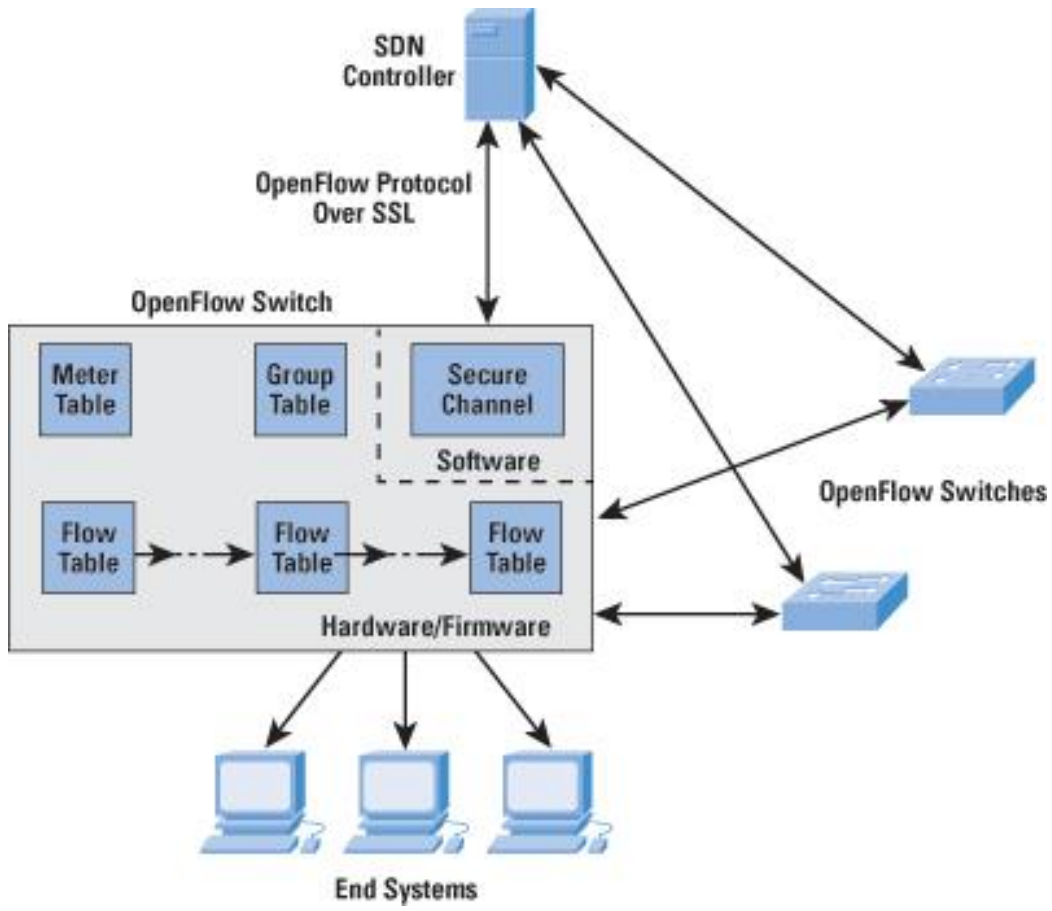


Figure 3-1 The OpenFlow switch logical architecture (Stallings, 2012).

Three table types are defined by the OpenFlow specification in the logical switch architecture (Figure 3-1). The flow table addresses the matching of packets to particular flows and choose the functions that are to be performed on the packets. Flow tables are not singular to a switch, hence multiple flow tables may be assigned a pipeline arrangement. A group table, has the job of triggering an action to more than one flow. The meter table triggers plenty operations that have to do with performance of the data flow (Stallings, 2012).

Going further to the analysis, the term “flow” must be standardized since it is not defined in the specifications of OpenFlow. Normally, as supported by Stallings (2012), “a flow is a sequence of packets traversing a network that share a set of header field values. For instance, a flow could involve all packets with the same source and destination IP addresses, or all packets with the same VLAN identifier”.

3.2.1 Flow tables

According to Openflow Switch Specifications v 1.5.0 (ONF, 2014), the flow entries that correspond to any flow table are the ones that are shown in Table 3-1.

Table 3-1 Main components of a flow entry in a flow table.

Match fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

Many such entries are listed in a table called flow table. The routing of packets from the Openflow protocol is based on these entries. Specifically, the structure of a flow entry is:

- Match fields: These are the fields that are matched against received packets. These contain the input port (ingress port), the headers and packet ports (headers, ports) and optionally any metadata.
- Counters: Counters on the received packets, for example how many matched successfully or how many were rejected.
- Instructions: Instructions for performance or modification of some actions taken during packet processing.
- Priority: Precedency of the matching of the flow entry.
- Timeouts: Expiration time or idle time, of the flow by the switch.
- Cookie: It is the opaque data value that is selected by the controller. Cookies can be utilized by the controller to filter flow entries affected by flow statistics, flow modification and flow deletion requests. Cookies are not used whenprocessing packets.
- Flags: The flagscan change the way of flow entry management, for example the flag OFPFF_SEND_FLOW_REMtriggers flow removed messages for that flow entry.

For the sequential comparison to the flow tables, the Openflow protocol follows the procedure of pipeline, thus creating a constant flow of packets through the flow tables, as can be seen from the following scheme:

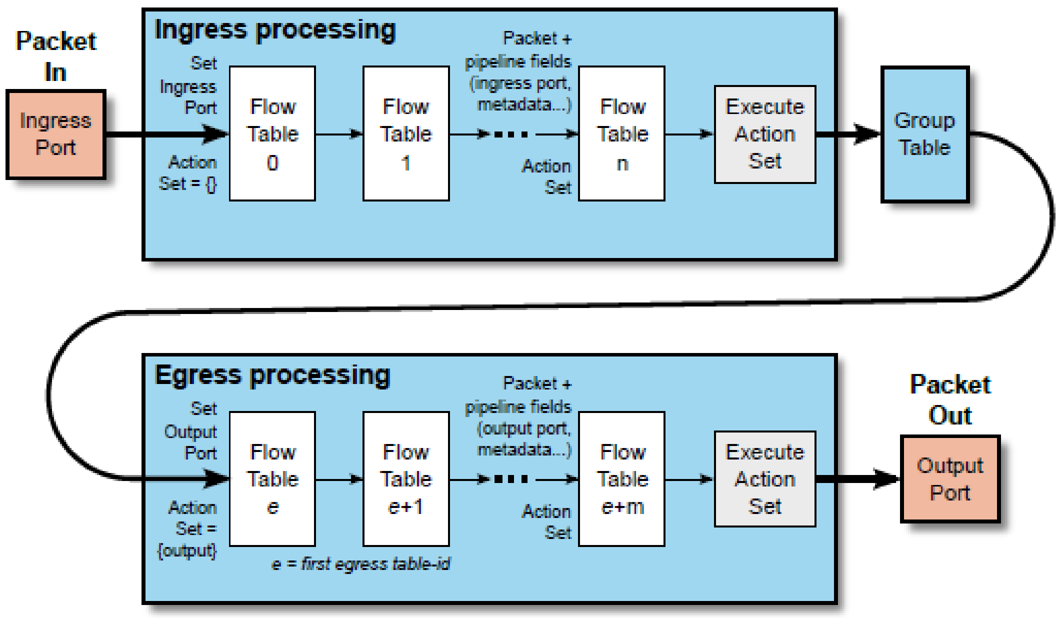


Figure 3-2Packet flow through the processing pipeline (ONF, 2014)

A unique flow entry in a specific flow table, is identified by match and priority fileds. A table miss flow entry is a flow entry with all fields omitted and priority equal to 0. The instruction field of the flow entry contains the actions that are to be performed on the packet at some point of the pipeline. For example, the action set-field specifies the rewriting of header fields. The different flow tables of a switch may not support the same set of match fields, instructions and actions. A request of table features may enable the controller to find out the supporting sets of each flow table.

As understood from the shape of the flow charts of OpenFlow router are numbered sequentially, starting with 0. The processing begins sequentially. The process of channeling always starts from the first flow table. To make the match when receiving a packet, a comparison is implemented of the header data of the packet, the packet input port and any metadata sequentially with the first flow entry of the first flow table and gradually all following entries in the other flow tables (as shown in Figure 3-3). The mapping is done in order of priority, namely the first successful matching flow entry is followed by the next entries in the remaining flow tables to be used according to the instructions of the matched entry.

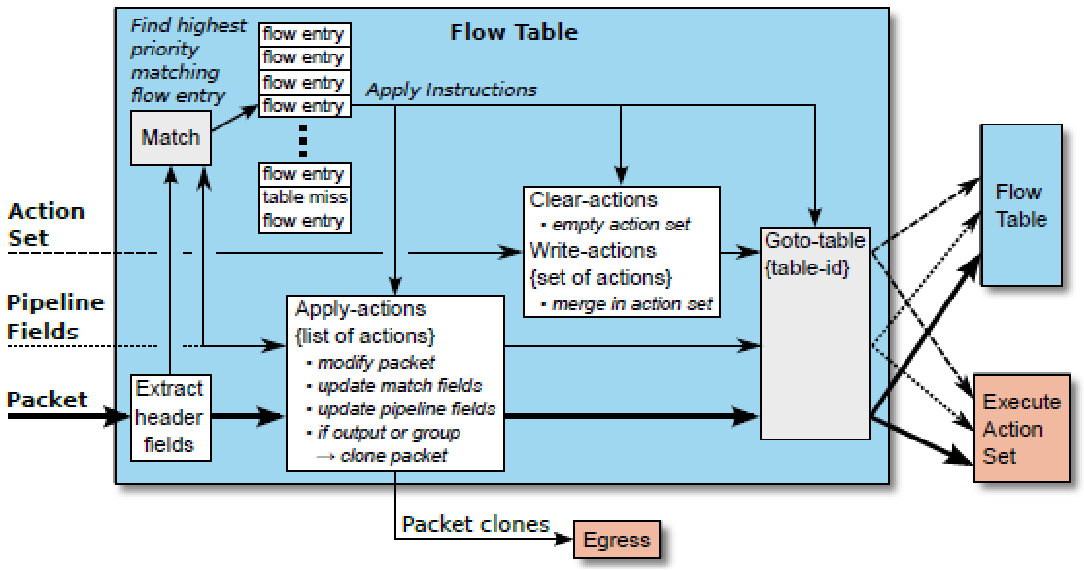


Figure 3-3Matching and Instruction execution in a flow table (ONF, 2014)

When a packet is matched with a flow entry of a flow table, the correspondin Instructions set is executed, that contains the entry. These instructions can redirect the packet to another flow table where the promotion process is repeated (Goto commands). Noteworthy is the fact that a flow entry can direct the packet to a flow table with a number greater than the current, as the diversion process works to promote the packet forward and not backwards. It is obvious that the flow entries of the last flow table may not contain instructions concerning the promotion process (Goto commands).

When the packet reaches the end of pipeline and there is no other forwarding command, then the packet is processed by the flow entry instructions and promoted in the analogue output port.

If a packet is not assigned to a flow entry of a flow table, then we havea table miss. The procedure followed in such cases depends on the policy of the OpenFlow network. By default in this case the corresponding packets are forwarded through the control channel to the controller. In some cases the process of packet drop is followed.

The instructions set contained in each flow entry is executed when a packet matches the entry. Therefore, the packet, action set or pipeline process may change (Figure 3-3). All instruction types are not required to be supported by a switch, onle the ones marked as “Required Instruction”, meaning Goto-Table, Write-Actions, and next-table-id. There are also optional instructions, such as Apply-Actions, Clear-Actions and Write-Metadata. One instruction is the minimum to be contained in each flow entry. The inability of executing an instruction set is a reason for the switch to reject a flow entry.

3.2.2 Counters

Counters are assigned to each flow table, flow entry, port, queue, group, group bucket, meter andmeter band. Counters that are OpenFlow-compliant may be implemented in software and maintained by pollinghardware counters with more limited ranges. The set of counters that are defined by ONF (2014) are presented in Table 3-2. Only the required counters of Table 3-2, are mandatory to be supported by all counters.

Second precision is required in tracking the duration, or time amount that the flow entry, port, group, etc. is installed to the switch. The field of “Receive Errors” maintains the sum of all errors (receive or collision) defined in Table 3-2.

Packet related counters, are assigned to count every packet that uses an Openflow object, regardless of the effect that the object has on the packet or if the packet is dropped. As an example, an Openflow switch must have packet related counters of the below items (ONF, 2014):

- A flow entry with only a goto-table instruction and without actions
- A group outputing to a non-existent port

- A flow entry triggering a TTL exception
- A port which is down

According to ONF (2012), “counters are unsigned and wrap around with no overflow indicator. If a specific numeric counter is notavailable in the switch, its value must be set to the maximum field value (the unsigned equivalent of-1)”.

Table 3-2 List of counters (ONF, 2014).

Counter	***Bits***	
Per flow table		
Reference count	32	*Required*
Packet lookups	64	*Optional*
Packet matches	64	*Optional*
Per flow entry		
Received packets	64	*Optional*
Received bytes	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per port		
Received packets	64	*Required*
Transmitted packets	64	*Required*
Received bytes	64	*Optional*
Transmitted bytes	64	*Optional*
Received drops	64	*Optional*
Transmitted drops	64	*Optional*
Received errors	64	*Optional*
Transmitted errors	64	*Optional*
Receive frame alignment errors	64	*Optional*
Receive overrun errors	64	*Optional*
Receive CRC errors	64	*Optional*
Collisions	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per queue		
Transmit packets	64	*Required*
Transmit bytes	64	*Optional*
Transmit overrun errors	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per group		
Reference count (flow entries)	32	*Optional*

Packet count	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per group bucket		
Packet count	64	*Optional*
Byte count	64	*Optional*
Per meter		
Flow count	32	*Optional*
Input packet count	64	*Optional*
Input byte count	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per meter band		
In band packet count	64	*Optional*
In band byte count	64	*Optional*

3.2.3 Group table

A group entry of a group table has the structure that is presented in Table 3-3 below.

Table 3-3 Main components of a group entry in the group table (ONF, 2014)

Group identifier	Group type	Counters	Action buckets

Each component of Table 3-3 is presented in the following:

- group identifier: group ID, which is an unsigned integer of 32bit and serves to uniquely identify the packet group on this entry of the Openflow switch.
- group type: the type of the group, showing the commands from the command group (action bucket) that will be performed for the packets belonging to the entry.
- counters: counters, used for various information, such as number of packets belonging to the group entry and are updated when the packets are processed by a group.

- action buckets: An ordered list of instruction where each instruction contains a set of actions and the relevant parameters to apply to the processing packet.

As for the group types, there are four categories:

- all: Performs all the commands in the command groups, with the packet to be cloned in copies and to process each command in the action bucket. This category is usually used for broadcasting and multicasting.
- select: Performs only a command of the entire command team, based on packet parameters and of the device selection algorithms.
- indirect: Perform only a predetermined command to the group table, which allows multiple packet streams to indicate a group identifier and supports faster and more effective promotion of packet groups to a specific destination.
- fast failover: Perform the first active command set, namely the set of commands that are associated with an active output port. With this method, each script in the action bucket is associated with an output port to enable the routing device in case of connection loss from an output port to relaunch packets in spare ports without having to communicate again with the controller.

3.3 SECURE CHANNEL

Secure Channel is an interface connecting each OpenFlow switch to a controller. Through this interface, the Controller regulates and manages the switch, gets informed of events via the switch and sends packets through it. The interface may vary depending on the implementation of OpenFlow switch, but each message destined to be sent through the secure-channel must be standardized with OpenFlow protocol.

3.3.1 Messages of controller to switch type

Messages of controller-to-switch type start from the controller to the switch, without being required to send a message in response. These messages are divided into the following subcategories:

- Features: Since the start of the session between the controller and the switch through Transport Layer Security (TLS), the controller sends a message to the switch requesting information on features (features request), and expects a relative answer (features reply).
- Configuration: The controller can configure the switch settings it controls, or request information about them. In this case the switch is required to respond with a note.
- Modify-State: These messages are used by the controller mainly for addition, deletion, or modification of the flow table that exists in the switch, or to adjust the properties of the ports.
- Read-State: Used by the controller to gather statistics on the flow tables, the ports, as well as for each record flow separately.
- Send-Packet: The send-packet messages are used to indicate the Controller to switch via which specific port to promote a package.
- Barrier: Barrier request / reply messages are used to confirm to the controller that the requirements for a particular message apply. They are even used to ascertain the controller for the completion of a process.

3.3.2 Asynchronous messages

The asynchronous messages are sent by the switch, without first having been requested by the Controller. Their purpose is to inform the controller of packet arrivals, changes in the state of switch, or an error that has occurred.

The four main asynchronous messaging sub types are:

- Packet-in: Any new packet that enters the switch and is not mapped to any existing flow entry, causes the creation and sending of a Packet-in message to the controller (packet-in event). If the switch has enough available memory to cache (buffer) this packet, then the message that will be sent will contain 128 bytes with the necessary information for the controller. The information relates to the values of the header of the packet entered, and one identification value (buffer ID) of the packet. If the switch does not support packet caching, or does not have enough available memory, then the message sent on the controller will include the entire original packet.
- Flow-removed: When a record is added to the switch flow from the controller via a flow-modify message, the switch is dictated about the idle time to erase that entry. It is even dictated when to shut off in general, regardless of the activity associated with this entry. Simultaneously this message type dictates the switch if it should notify the Controller after such a deletion, which is a flow-removed messagetype.
- Port-status: The switch uses these messages in cases of state change of a port, for example, if a user disables a specific switch port. Additionally, it is used in cases of change in state of a port as defined by Protocol 802.1D.
- Controller role status: A message that informs the controller, when its role is changed by the switch, and not directly by the controller via a OFPT_ROLE_REQUEST message.
- Table status: The OFPT_TABLE_STATUS message informs the controller aout a change in table state status.

- Request – forward: The OFPT_REQUESTFORWARD message forwards the request to the controller, which modified the state of groups and meters.
- Controller status: The switch sends a message to all connected controllers when the status of one controller changes.

3.3.3 Symmetric messages

Symmetric messages may be sent either by a switch, or a controller, without the other party having requested such an action, and are separated into the following three categories:

- Hello: Messages of this type are exchanged between the switch and the controller the moment the connection between them is established.
- Echo: Messages of type echo request / reply can be sent either sides and are used for measurements of delay (latency) or frequency range (bandwidth). They are also used to verify whether the connection between them is active.
- Error: With these messages, the switch can inform the controller for problems or errors that may arise.

3.3.4 Read state messages

Messages of type Read-State, are used by the controller to gather statistics from the router. In order to find the desired path in the routing algorithm, it is necessary to collect statistical data on various network elements and store these data into a database that will be updated frequently.

The OpenFlow protocol enables such data collection through Read-State. More specifically, the controller sends at regular intervals messages of type OFPT_STATS_REQUEST to the network devices. Routers respond with one or more messages of type OFPT_STATS_REPLY.

The only value specified as flag in a reply message is the value 0x0001, and is defined only if more than one replies are followed. To facilitate implementation, the routers may send replies without additional entries. However, they should always be sent after a message containing flag field values. The identities of the transactions (xid) replies must always match the request. Both in requests and replies, the field type determines the type of information provided and determines how the field is interpreted.In all types of statistics, if a counter value is not available on the router, its value is set to -1.

3.4 CONNECTION OF SWITCH-CONTROLLER

The switch should have the capability to establish communication with the Controller, via an IP address and a port, which are set by the user of the switch and remain stable. The movement through the secure channel is not checked against entries in the table of flows, and for this reason, the switch must recognize the rest of network traffic as local, in order to compare with recordings of the flow-table.

The communication between the switch and the Controller is via a connection TLS. Linking this initiates the switch to the Controller, who usually expects this to TCP port 6633. Then the authentication takes place via exchange certificates with private key. For this reason, each switch must have a certificate certifying the Controller and another for his certification to the Controller.

If after some time the connection is lost, the switch attempts to connect with him, or spare controllers that have been defined. If this connection fails after a specified number of attempts, the switch automatically enters a safety mode (emergency state) and returns a list of flows to its original state (reset). From that moment, and until it again reaches its connection to a Controller, the mapping and forward process of packets entering the switch, dictated by a flow table security (emergency), which will be determined by the user. So, once recovered the connection to a Controller is possible to maintain or abolish these safety records.

3.5 CONTROLLERS

3.5.1 The NOX controller

The first open source control platform for the first applications of the Openflow technology was NOX. NOX has enabled researchers to develop web applications using programming techniques with Python, and the programming language C ++. Over the years created the need to develop a platform that could cover more needs in applications but also provide higher yields (Voukelatou, 2015).

NOX was first introduce in 2009, developed by Nicira Networks and now owned by VMware, along with OpenFlow. Later, different NOX controllers were developed (Rao, 2014a):

- NOX classic is a version available under the GPL since 2009.
- NOX has support only for C++ and limited applications compared to the classics, but is faster with better codebase.
- POX support the Python language and can be regarded as NOX sibling.

When we refer to NOX we almost always mean the new NOX, no the NOX classic. The following table summarizes the differences among the classic and new NOX.

Table 3-4 Differences among NOX and NOX classic (Rao, 2014a).

	NOX	**NOX classic**
Core apps	OpenFlow, Switch	Messenger, SNMP, switch
Network Apps	—	Discovery, Topology, Authenticator, Routing, Monitoring
Web Apps	—	Webservice,Webserver,WebServiceClient
Language Support	C++ Only	C++ and Python
GUI	NO	YES

Figure 3-4presents the architecture of NOX. Helper methods are provisioned by NOX core, i.e. network packet process, threading and event engine, in addition to OpenFlow APIs for interacting with OpenFlow switches, and I/O operations support.

Core, Net and Web applications are at the top. Core appplicaitons are only two, namely OpenFlow and switch, while there are no network and web applications. In the middle, are the in-built components of NOX. These are connection manager, event dispatcher and OpenFlow manager and the dynamic shared object (DSO) deployer which checks the directory structure for any components being implemented as DSOs.

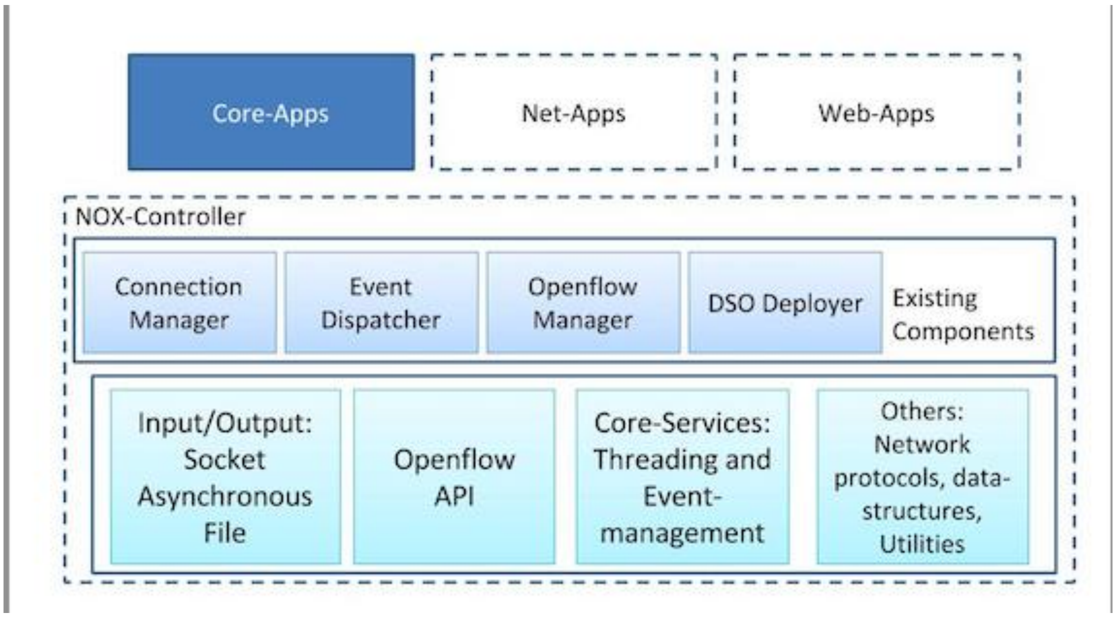

Core-Apps
Net-Apps
Web-Apps
NOX-Controller
Connection Manager
Event Dispatcher
Openflow Manager
DSO Deployer
Existing Components
Input/Output: Socket Asynchronous File
Openflow API
Core-Services: Threading and Event-management
Others: Network protocols, data-structures, Utilities

Figure 3-4 NOX Architecture (Rao, 2014a)

All applications can be regarded as components because they all inherit from the component class. Various different components that cooperate with each other compose the NOX applications to provide the required functionality. Hence, the functionalities of every component are made available to NOX.

The event system of NOX is very important and may be addressed by low and high level event into the network system that provide information which is processed by the handlers. The events report something that happens in the network and may be important to the NOX controller. Due to the fact that the event information are processed by handlers, it can be said that every event corresponds to an execution in NOX. Events are divided into core events and application events. NOX core events are presented in Table 3-5.

Since NOX has limited applications, NOX classic incorporates events that are linked to applications, such as host_event, flow_in_event and link_event.

Table 3-5 NOX core events (Rao, 2014a)

OpenFlow-Events	**Description**
Datapath_join_event	When a new switch is detected.
Datapath_leave_event	When a switch leaves the network.
Packet_in_event	Called for each new packet received.
Flow_mod_event	When a flow has been added or modified.
Flow_removed_event	When a flow in the network expires/removed.
Port_status_event	Indicates a change in port status.
Port_stats_in	When a port statistics message is received.

3.5.2 The POX controller

The POX provides a platform for developing and prototyping network control software, using the Python language. In fact, POX is the transfer of NOX (another highly prevalent Controller) in Python. POX acts as the general background on which all communication with OpenFlow switches is realized and is one of the most popular tools in the SDN sector. It is under constant updating and an ideal solution for familiarization with the environment of the Software Defined Networks.

Some characteristics of POX are (Rao, 2015):

- "Pythonic" OpenFlow interface.
- Reuse of various elements (path selection, topology discovery).
- "Runs anywhere" – Runs in any system, specifically targeting Linux, Mac OS and Windows.
- It supports the same graphical user interface and visual tools with the relative controller NOX.

- It has good performance compared with the NOX applications written in Python.

The execution of the POX is through pox.py file. But performing exclusively pox.py file does not have a particular result. The main functionality of the POX is provided by components (components). The POX has some components (however mainly intended for an audience that aims to develop its own components).

When talking about components, we mean files that can be added to the instruction with which we invoke the POX. POX has features of construction components, some that provide basic functionality, a number of convenient features and some are just examples. Some indicative available components are (Pavlidis, 2015):

- Py: Start an interactive Python interpreter. Use for debugging and experimentation.
- Forwarding.hub: Simple installation FLOOD rules in each switch, transforming them into hubs.
- Forwarding.l2_learning: Similar function to an L2 learning switch. Although the component focuses on the various layer 2 addresses, the rules are installed as specific (the fewest possible wildcards) as possible.
- Forwarding.l2_pairs: Similar to l2_learning operation however the rules are mainly based on MAC addresses.
- forwarding.l3_learning: This component is not just a router, but also not a L2 switch. It learns and records IP addresses, without giving special attention to concepts such as different subnets. In order to align with host operation, it is possible during loading to define fake gateways.
- forwarding.l2_multi: It behaves like a learning switch. The significant difference of the corresponding components is that it uses the openflow.discovery to find out the whole network topology. So when a switch registers a MAC address, all share this entry. Note that it also requires the use of openflow.discovery component.

- forwarding.topo_proactive: Installation rules in advance, based on an IP address that is relevant to the topology. The switching is based on forwarding.l2_multi mentioned earlier.
- openflow.spanning_tree: Uses openflow.discovery to create an overview of the network topology, it creates a spanning tree, and then disables the flooding for ports that are not in the spanning tree.
- openflow.webservice: It provides a web service of JSON-RPC type for interacting with the OpenFlow protocol. Derived from of_service messaging service and requires the use of webcore component.
- web.webcore: Start a web server in the process of POX.
- Messenger: It offers an interface for the POX to communicate with external processes through interactive messages based on JSON.
- openflow.discovery: Send tailored LLDP message to the controller to discover the network topology.
- openflow.debug: Create pcap files with OpenFlow messages to be exchanged.
- misc.full_payload: The default behavior when a packet can not be mapped to any record of the flow table is sending a portion of the packet (the first 128 bytes) in the controller. This component suitably amends the switches connected to the controller to send the whole packet.

3.5.3 The Floodlight controller

Floodlight Controller is based on Java, but has a different architecture and operating mode. The controller comprises an autonomous collection of modules which perform the main functions of Floodlight as OpenFlow controller and applications are developed so as to overlap (on-top) the base unit controller (REST applications) or to operate together with the controller (Module applications), as shown in the following scheme (Izard, 2016).

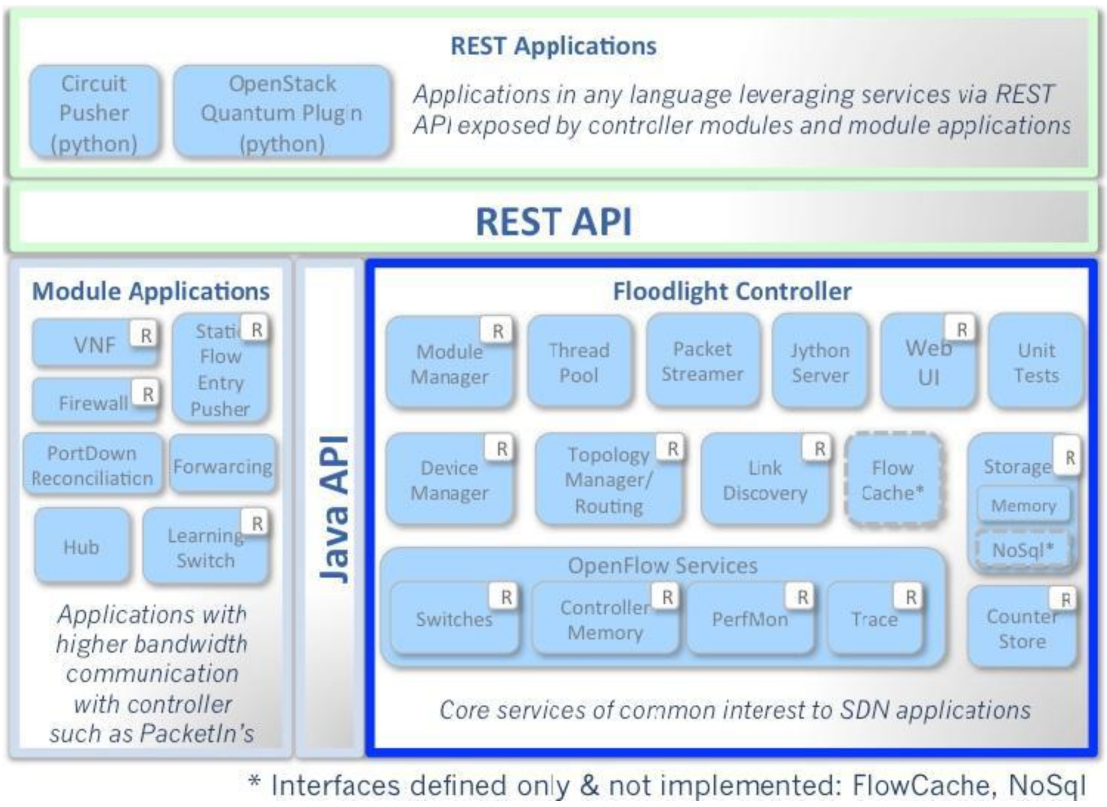


Figure 3-5Floodlight controller and REST applications (Izard, 2016)

As shown above, Floodlight consists of functional topology and link modules (Topology Manager, Link Discovery), control devices and units (Device Manager, Module Manager), OpenFlow services (OpenFlow Services), unit storage and handling (Storage, Counter Store, Flow Cache, Packet Streamer, Thread Pool).

Applications that use Floodlight as an underlying layer are developed as independent Java modules and use the programming REST interface of the controller (REST API - application programming interface). Through its REST API applications Floodlight communicates with the controller and can use the network for any function (Izard, 2016).

Also applications can be developed and adjusted to the level of Floodlight controller, enough to develop and implement the controller modules as Java (Java modules). This way, although more demanding and difficult than the use of the REST API's, has significant benefits as regards the execution speed of the application and greater communication bandwidth offer with Floodlight, since the coupling of application - controller becomes more directly to the use of Java API's (Izard, 2016).

3.5.4 The RYU controller

Controller RYU (in Japanese stands for dragon) is an implementation of OpenFlow in Python and is considered flexible and agile for normal network operations. It consists of collection of subprograms and libraries as Netconf, OF-conf and supports the OpenFlow protocol 1.0, 1.2, 1.3, 1.4.

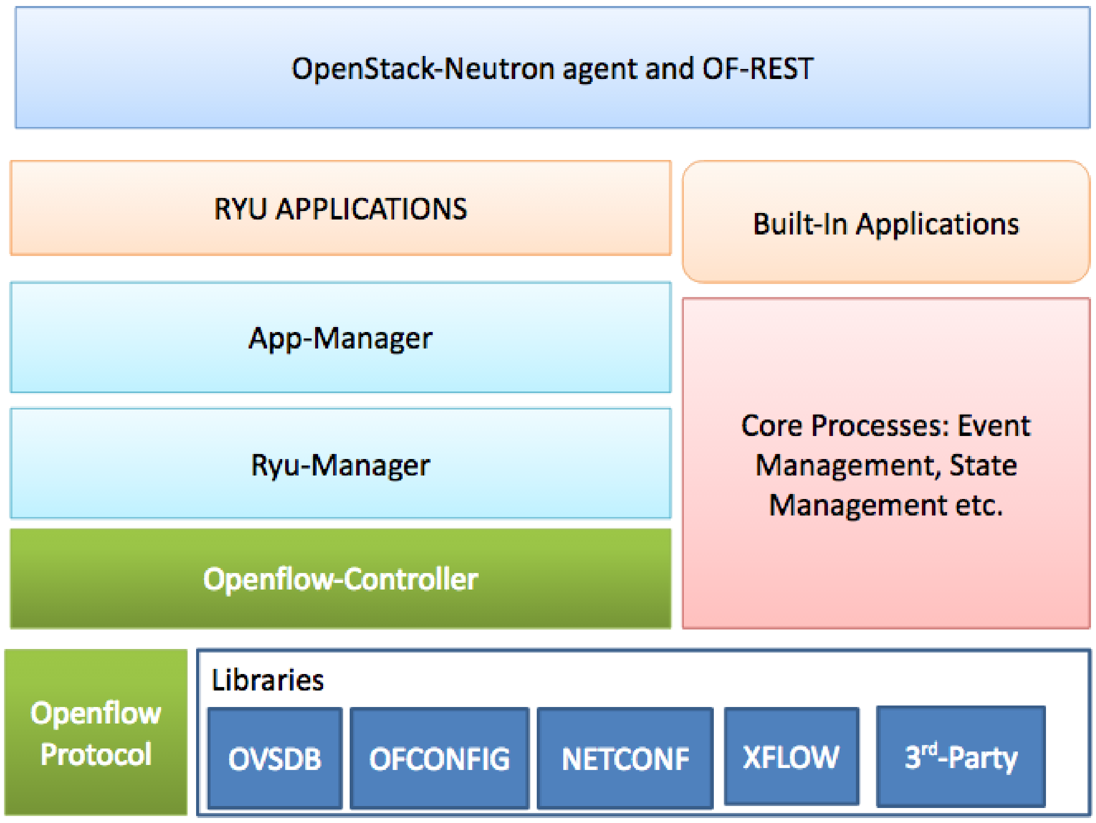


Figure 3-6 The RYU controller architecture (Rao, 2014b)

The collection of Ryu libraries is impressive, incorporating support for multiple southbound protocols and different packet processing operations. Regarding southbound protocols, OF-Config, Open vSwitch Database Management Protocol

(OVSDB), NETCONF, XFlow (Netflow and Sflow) and other third-party protocols ar supported by Ryu. Cisco supports Netflow while it is IP specific. Netflow protocol as well as Sflow, provides packet sampling and aggregation, operations that are utilized mainly for network traffic measurement. Open vSwitch Python binding, the Oslo configuration library and a Python library for the NETCONF client are third-party (Rao, 2014b).

The latest versions of OpenFlow protocol are supported by Ryu,including a library for encoding and decoding OpenFlow. The controller is a main component of the Ryu application responsible for managing the OpenFlow switches for flow configuration and event management. Events are internally generated mainly by the OpenFlow controller in the Ryu architecture (Rao, 2014b).

Table 3-6 Ryu OpenFlow protocol messages, structures and corresponding API (Rao, 2014b)

Controller to Switch Messages	**Asynchronous Messages**	**Symmetric Messages**
Handshake, switch-config, flow-table-config, modify/read state, queue-config, packet-out, barrier, role-request	Packet-in, flow-removed, port-satus, and Error.	Hello, Echo-Request & Reply, Error, experimenter
send_msg API and packet builder APIs	set_ev_cls API and packet parser APIs	Both Send and Event APIs

The distribution of Ryu is made with multiple applications i.e simple_switch, router, isolation, firewall, GRE tunnel, topology, VLANthat are single-threaded entities, providing different functionalities. Asynchronous events are sent by Ryu applications among each other.

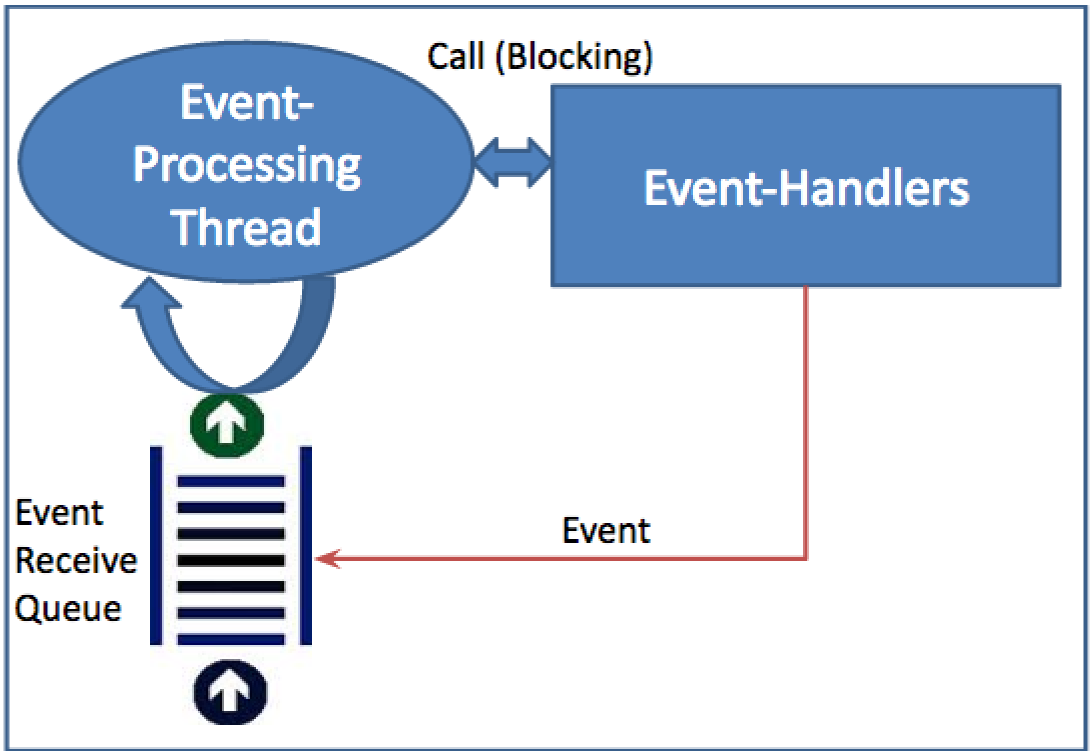
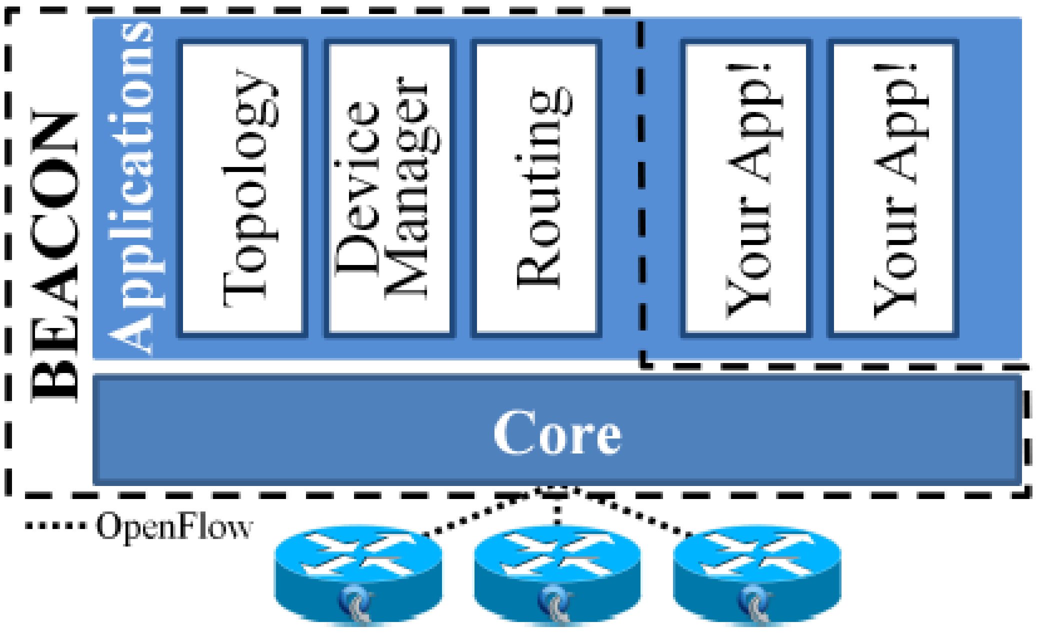


Figure 3-7 Functional Architecture of Ryu Application (Rao, 2014b)

3.5.5 The Beacon controller

The Beacon is one of the most commonly used open-source controllers in OpenFlow. It was created at Stanford University in 2010 and is developed in Java. It is used for teaching and research purposes. The programming language and development environment have a significant impact on the productivity of developers, and can also be a limiting factor in application performance. There are manyfriendly languages, but their performance when used in an OpenFlow controller is unknown (Christoforou, 2014).

Before the development of Beacon Controller, the main controllers' programming languages was C and C ++. These languages are used to produce very high performance applications, but also affect the performance. With the use of these languages,problems were usually presented such as a large compilation time (> 10 minutes), errors in the build process, memory management problems that lead to segmentation faults, memory leaks etc. (Erickson, 2014).



Εικόνα3-1 Overview of Beacon (Erickson, 2014)

Although efforts have been made to minimize such problems, they need cumbersome techniques and the use of peripheral devices, which requires devices with increased capabilities. This resulted in the need to develop a platform based on friendly programming language and provide high yields that can be installed in basic hardware, in which the CPU and the RAM can be easily expanded with the lowest possible cost.

The main features taken into account for the programming language to be selected for the development of this platform were: memory management, cross-platform, and high performance.The automatic memory management (called garbage collection) can eliminate most programming errors related to memory. Languages with this ability usually have little or no preparation, eliminating the waiting time for compiling the program. Also such languages provide error report indicating the exact point of error. The languages that meet these abilities is C #, Java and Python (Erickson, 2014).

Although the Beacon platform is designed to run on Linux operating system, the developers wanted to be executed in the other two most important operating systems, Windows and Mac OSX, without significant porting effort. The concept of high performance is subjective since a feature of the definition is the ability to increase efficiency according to the use and the number of processing cores. The lack of multi-threading ability by the official Python compiler eliminates this language as a candidate. The remaining basic choice is Java language (Erickson, 2014).

The Beacon controller uses multiple libraries in an effort to maximize the reuse of code and to lighten the burden of the controller itself and the applications that use it. The most important library is Spring. The Spring, is a lightweight Java / J2EE application framework. The Application Programming Interface (API) for Beacon is designed to be simple and effective without restrictions. Developers are free to use any available Java constructor, as Threads, Timers, Sockets, etc. The API includes information on the interaction with OpenFlow routers. The Beacon uses the model of the observer where listeners are correlated to respond to specific events (Christoforou, 2014).

The Beacon also includes applications based on functions related to the core, adding additional APIs (Erickson, 2014):

- Device Manager. Monitors devices connected to the network, including: address (Ethernet and IP), last login date and router door and the device connected last. The API Device Manager provides an interface (IDeviceManager) designed to search known devices and update feature for various events such as when new devices are added, that when updating, or removing devices.
- Topology. Identifies the links between routers connected OpenFlow. ITopology interface retrieves a list of such links, and records events such as the connection or removal of a link.

- Routing. It offers the shortest routing path (1-data link layer) between network devices. This application uses the IRoutingEngine environment, allowing changes to settings that control routing. The applications of this type using all possible shortest path calculation. The Routing API depends on both the corresponding Topology and Device Manager.
- Web. It provides a UI (User Interface-User Interface) through LAN (localhost: 8080) for the Beacon Controller. The web application that uses the IWebManageable interface, allows the addition of new elements in the UI.

4 WIRELESS AND MOBILE NETWORKS

4.1 HISTORY OF WIRELESS NETWORKS

The first successful wireless networking attempt took place in 1901, with the wireless telegraph of G. Marconi. To transmit messages, Marconi used the Morse code, which can be considered a precursor of the binary system of modern digital systems (Kallipos, 2015).

More recently, wireless networks developed in the form of TCP / IP radio networks. The first packet switching techniques were developed around 1964, while the concept of the packet was introduced in Great Britain. The technology of wireless packet transmission networks began to grow in the period 1970-1980, although the high growth coincides with the spread of microcomputers in 1980-1990. An important step in the development of integrated local wireless networks, i.e. LAN (Local Area Network) was the ALOHA network in the University of Hawaii.

Today, a number of new devices and wireless communication products are available based on new technologies and new standards. In recent years, various types of mobile devices (notebook, laptopand tablet) are adopted by the general public, since they offer cost effective, computing power and quality services compared to their desktop computers. A key point on the development of the mobile market has been the introduction of smart phones. The use of smart phones by consumers has led to the sharp increase in traffic of network data and a multitude of new services and applications. All the above, results in continuous research to develop new standards to support wireless communications (Kallipos, 2015).

In recent years there has been rapid improvement in the quality of wireless networks with speeds which can exceed those of wired networks and the adoption of new standards by organizations and alliances. Bluetooth, GPRS (General Packet Radio

Service), HIPERLAN (High - Performance European Radio LAN) by ETSI organization (European Telecommunications Standard Institute), 802.11 WLAN (Wireless Local Area Network) and 802.16 by IEEE organization (Institute of Electrical and Electronics Engineers), as well as the UMTS (Universal Mobile Telecommunication System), and LTE (Long Term Evolution) by 3GPP organization, are the most representative examples. All these standards cover the requirements for the physical layer and the MAC substrate (Medium Access Control). Furthermore, the third (3G) and fourth (4G) cellular generation network systems developed by the 3GPP alliance (3rd Generation Partnership Project) offer data rates, which were previously achieved only through a wired network (Kallipos, 2015).

4.2 TYPES OF WIRELESS NETWORKS

Wireless networks are categorized as follows (Kallipos, 2015):

- Personal Area Networks - PAN
- Local Area Networks - LAN.
- Wide Area Network - WAN

The PAN interface refers to the interface of the computer components using short-range radio or devices which are placed on the body or in close proximity to the user. The popular PAN protocol is Bluetooth, which facilitates interconnection and file transfer between end user devices.

In the wireless local area networks (Wireless LANs, WLANs), each device has a wireless modem and an antenna, via which it can communicate with other systems. The WLAN in turn can be connected to a wired LAN or to help create a new network. The area covered by the WLAN depends on the transmission power, the antenna pattern and the characteristics of the network region, such as walls and obstacles. In its simplest form, the radiation pattern of the WLAN antenna is isotropic, that covers an approximately circular area. The network devices (laptops, tablets, and smartphones) can move into the WLAN, without being disconnected from the network. The communication between devices within the wireless network coverage area is coordinated by the access point (Access Point, AP). The AP can connect various WLAN to another and can also be connected to the WLAN cells to a wired LAN cable to an output of the wired LAN. The third type is the wireless WAN with its main application in cellular mobile networks. Now located on the fourth generation, cellular networks can provide data speeds comparable to the LAN with a radius coverage of tens or hundreds of meters (Kallipos, 2015).

4.3 WLAN

A wireless local area network (WLAN) is a network which allows communication via radio waves between different computer systems (stations), with the use of a central node (access point) within a service area of that node. The range can vary from a few dozen to a few hundred meters. A wireless local area network, can operate autonomously providing interconnection to a group of computers within the network range, or it can also function as an extension of a wired network to a unified network (Stathopoulos, 2011).

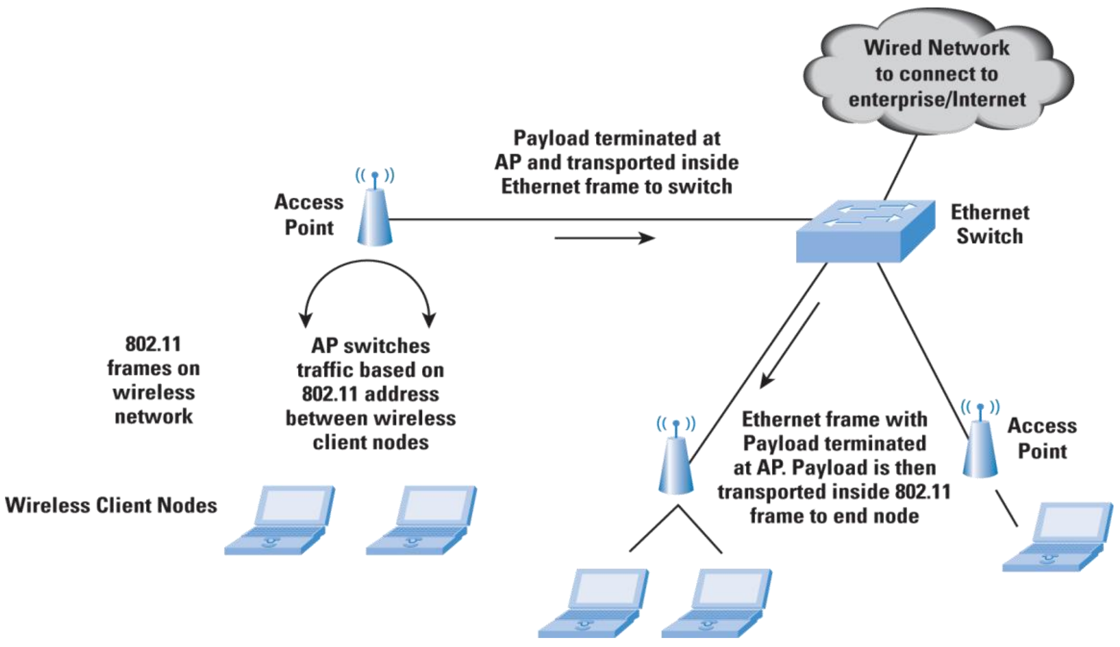

Wired Network to connect to enterprise/Internet
Payload terminated at AP and transported inside Ethernet frame to switch
Access Point
Ethernet Switch
802.11 frames on wireless network
AP switches traffic based on 802.11 address between wireless client nodes
Wireless Client Nodes
Ethernet frame with Payload terminated at AP. Payload is then transported inside 802.11 frame to end node
Access Point

Figure 4-1 WLAN network with Ethernet connectivity (Sridhar, 2008).

Today, the use of wireless local area computer networks, is very widespread, finding application in commercial, academic and personal computer networks around the world. The prevalent communication standard for implementing wireless local area computer networks, as has been established so far is the IEEE 802.11, which, from its original creation in 1997 by the Institute of Electrical and Electronics Engineers (IEEE), has received a large number of renewals for better operation and addition of new features. The various versions of 802.11 are presented in Table 4-1 (Kallipos, 2015).

The WLAN protocol implements a set of features that guarantee access and transmission security (authentication, encrypted transmission) and also features offered for roaming services, where a LAN subscriber can be connected to another WLAN (the case of WLAN that have been implemented in airports). Recently wireless access platforms were developed, where WLAN owners allow free access to their network, providing access to other AP, whose owners are members of such platforms.

Table 4-1 IEEE 802.11 versions (Kallipos, 2015).

Version	Features

IEEE 802.11	Original specification at 2.4GHz with rates up to 2Mbps
IEEE 802.11a	Physical Layer Extension for 5,2GHz (rates up to 54Mbps). Created after the 802.11b and is based on OFDM technology.
IEEE 802.11b	Physical Layer Extension in the 2.4GHz (rates up to 11Mbps)
IEEE 802.11c	Extended support for wireless bridging function in the MAC (link wireless LANs or different segments of the same LAN)
IEEE 802.11d	Add operation control process in different countries
IEEE 802.11i	Improvements in security
IEEE 802.11j	Amendment of 802.11a for 4.9-5.0GHz
IEEE 802.11k	Technical corrections and clarifications (Documentation of specifications).
IEEE 802.11m	Amendments to high rates (MIMO)
IEEE 802.11n	Radio resource management system
IEEE 802.11p	Amendment of 802.11a communication vehicle-to-vehicle
IEEE 802.11s	Provision Protocol autoregulation of mesh networks
IEEE 802.11w	Providing data integrity and authentication
IEEE 802.11ac	Amendments to high rates (5GHz, 80-160MHz bandwidth, 500Mbps).
IEEE 802.11ad	Amendments to high rates (60GHz, WiGig, 7Gbps)

WLAN network architectures can be categorized in three types (Sridhar, 2008):

- Autonomous Architecture
- Centralized Architecture
- Distributed Architecture

As part of the autonomous architecture, the 802.11 function is implemented and terminated by the Wireless Termination Points in order for frames on the wired LAN to be 802.3 frames. EveryWireless Termination Pointhas the capability of autonomous management as a separate entity.

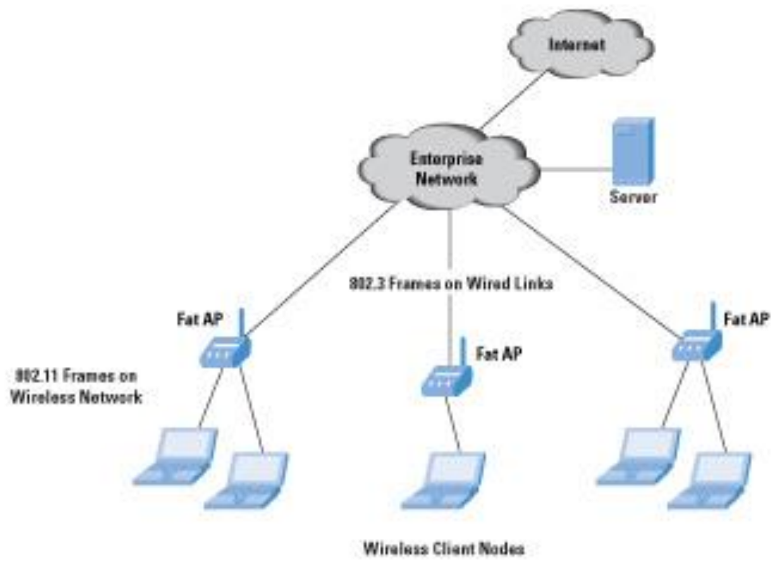


Figure 4-2 FAT APs in Autonomous WLAN Network Architecture (Sridhar, 2008).

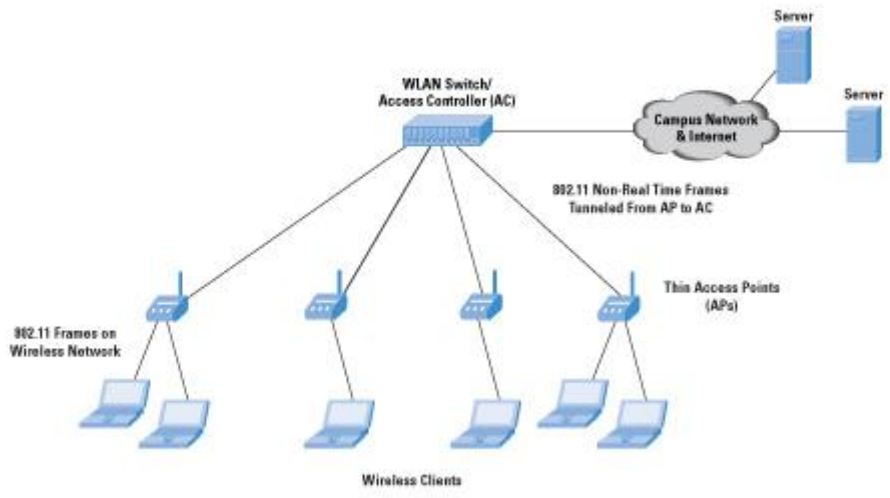

Server
WLAN Switch/ Access Controller (AC)
Server
Campus Network & Internet
802.11 Non-Real Time Frames Tunneled From AP to AC
Thin Access Points (APs)
802.11 Frames on Wireless Network
Wireless Clients

Figure 4-3 Thin APs in Centralized WLAN Network Architecture (Sridhar, 2008).

According to the centralized architecture, a hierarchy is implemented with the WLAN controller to undertake configuration, control, and management of several Wireless Termination Points. This controller is also referred to as the Access Controller. Both wireless termination points and the access controller perform the 802.11 function.

The major disadvantages of WLANs are (Kallipos, 2015):

- Power consumption: To take advantage of the mobility offered by the wireless network must use portable devices. If appropriate measures are not taken to their charge or reduce the transmission power, the autonomy of the device can be decreased dramatically.
- Transmission rate: Although WLAN offer data rates comparable to wired networks, the wireless medium is not always possible to achieve the theoretical transmission rate.
- Interference: As in any wireless system, so in the case of wireless networks, the interference can result in a reduction of network performance. Interference can come from other network stations in an attempt to transmit at the same time or other devices that use the same spectral range, particularly when using the ISM band.
- Security: Data presented in a wireless network is easily intercepted by anyone, provided it has the appropriate receiver and access the network coverage area. Therefore, encryption should be used, thus increasing the cost and reducing the performance of the network.
- User Safety: The safety of users during the use of radio is a subject that is constantly being studied. In wireless networks one of the grounds for limiting transmitted power aims to reduce users' exposure to non-ionizing radiation.

4.4 WiMAX

WiMAX (Worldwide Interoperability for Microwave Access) is defined under the IEEE 802.16 working group, while there are two standards, namely 802.16d-2004 and 802.16e-2005 for fixed access and mobile stations respectively (Sridhar, 2008).The 802.16e standard and its successor 802.16m, have been selected by industry for the manufacture of devices and equipment of WiMAX networks. From a user's perspective, there is a plethora of devices to support both fixed as well as mobile access network (Andrews et al. 2007). These specifications converge in the radio interface between the mobile terminal and the base station. The standards define the technical parameters and the basic procedures for wireless users to the access network, the physical level and the level of control of wireless access. The functions that have been standardized is the discovery and selection of network entry and exit of users in the network, service quality, safety, handovers, and various power management techniques (Kallipos, 2015).

The WiMAX base stations are built by networks and mobile terminals. Each base station provides wireless connection to multiple users who are located within the station's coverage area. The interconnection of base stations to the core network are not determined by all available embodiments, such as a wired DSL network, fiber optic network standard IEEE 802.16x and therefore can be used also wirelessly via microwave point to point links. The latter option essentially appears to be the most popular, as a base station can be used for wireless support by one or more base stations, and operates in the microwave frequency band. It becomes immediately apparent that by this way, point-to-multipoint connections can be supported (PMP) for the same network backbone. Thereby achieve the optimization of the network and the best development, aiming at maximum coverage. In addition, very important advantage of WiMAX is that it allows providers who do not have wired backbone network and want to reduce the cost of line rental from others to adopt this technology for the development of their networks at reduced cost (Kallipos, 2015).

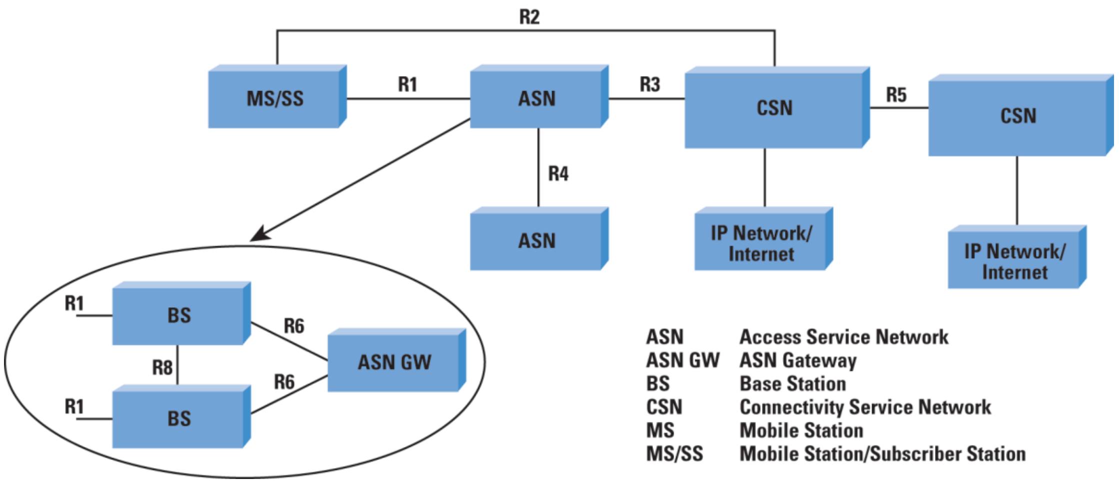


Figure 4-4 WiMAX Forum Network Architecture Functional Blocks and Interface Points (Sridhar, 2008).

4.5 CELLULAR NETWORKS

An equally important class of wireless networks are cellular networks or mobile networks. In recent years they have made significant advancements with respect to increasing the capacity of these networks, combining broadband coverage with large coverage radius. The mass adoption of smartphones makes cellular systems as the main representative of wireless networking, giving incentive to researchers and companies for their further development. Until today, there is a continuous development of cellular systems from second up to fourth generation networks, as shown in Figure 4-5 (Holma & Toskala, 2010).

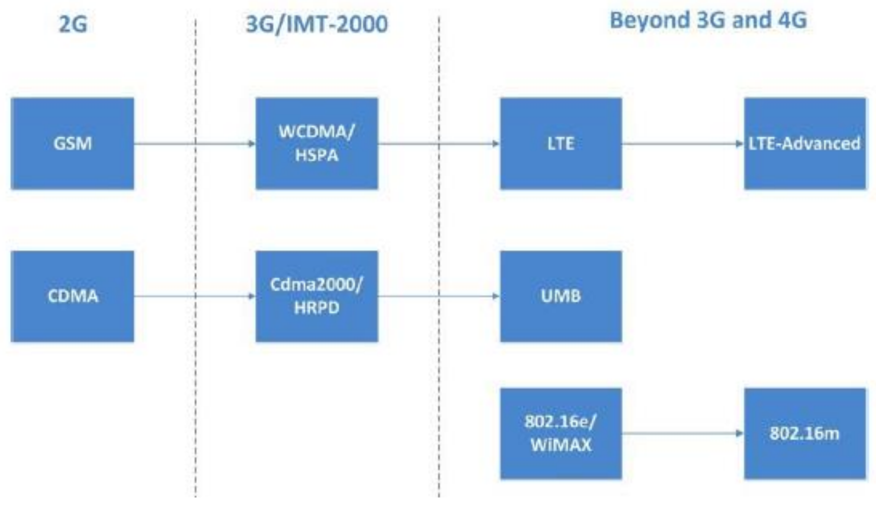


Figure 4-5 Advancements in cellular networks (Holma & Toskala, 2010).

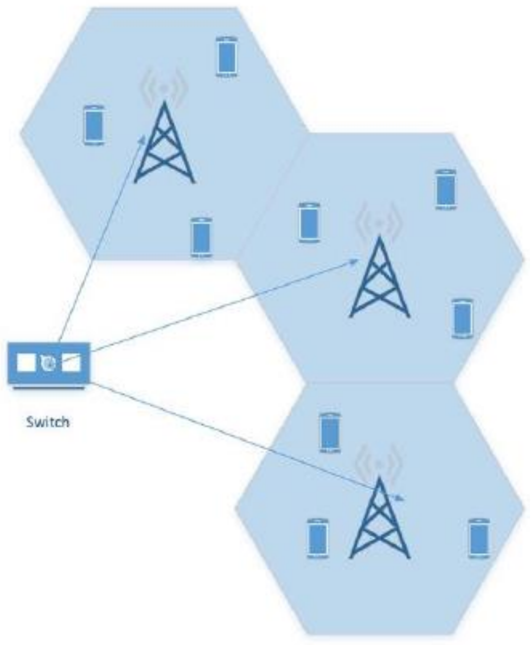


Figure 4-6 Cellular network structure (Tse & Viswanath, 2005).

Generally, in cellular networks, when the user wants to make a call, the phone communicates with a base station, requesting to be connected to a given telephone number or by a network node with a specific IP address. If the base station has sufficient resources to satisfy the call, a switching center will undertake to rout the user's data. In this way, the call takes up a wireless channel which is assigned exclusively to the user to complete. The basic structure is shown in Figure 4-6 (Tse & Viswanath, 2005).

4.6 WIRELESS SENSOR NETWORKS

Wireless sensor networks consist of a large number of physical devices that are geographically close to one another, developed in a monitored area, and connected by a wireless multi-hop manner. These devices have the same goal: the wireless infrastructure aims to identify the events that occur in environmental monitoring, and to transfer the sensory results in a small number of special gateways (i.e. sinks), which eventually send aggregated data in remote management units(Tse & Viswanath, 2005).

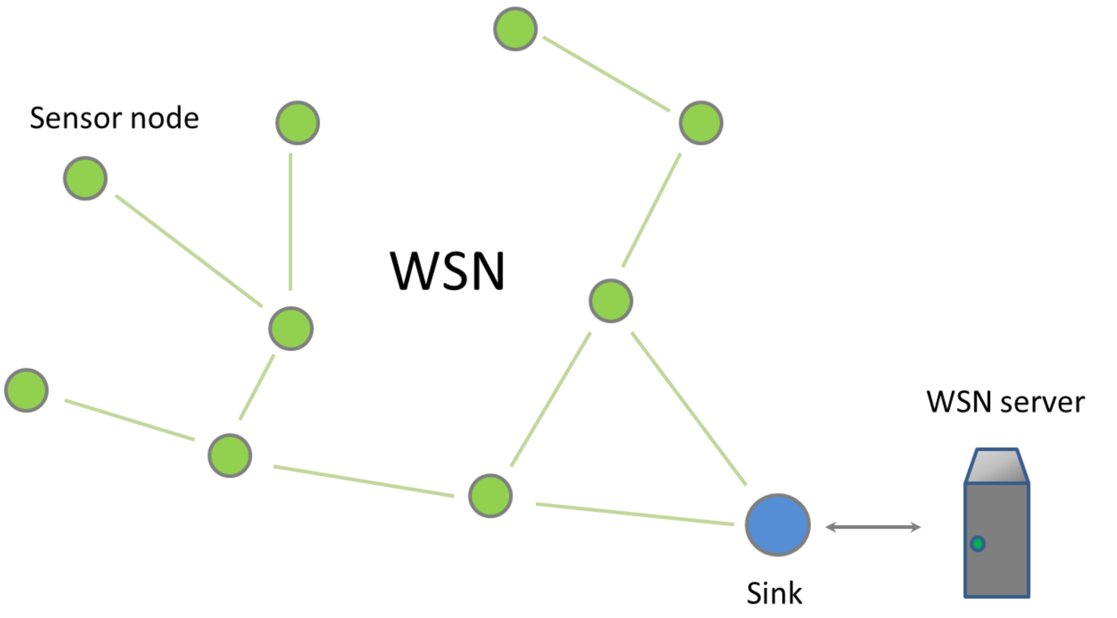


Figure 4-7An example of a WSN (ITU, 2014)

Wireless sensor networks are widely applied in a great number of applications, such as monitoring the military, environment, health, home and industrial applications. Usually the sensor nodes consist of small and inexpensive devices powered using batteries, and therefore their terms of energy supply capacities and information is very limited. Consequently, the optimization of the energy consumption was the main driver in the field of sensor network research. Proposed protocols and applications that have been designed keeping in mind energy efficiency (IEC Whitepaper, 2014).

More recently, energy harvesting technologies have been proposed for the same purpose of maximizing the lifetime of wireless sensor networks: a node sensor may be able to draw energy from the environment and adds to the battery, so long as it has a collecting circuit and a nearby energy sources such as light, wind or vibrations.The machine-to-machine (M2M)environment, largely an extension of the sensor network model represents a more advanced type of network that the data communication between physical devices without human intervention. The M2M networks inherit limited resources, non-guarded and mass sensor networks, and developed through the integrated intelligence and self-organization (ITU, 2014).

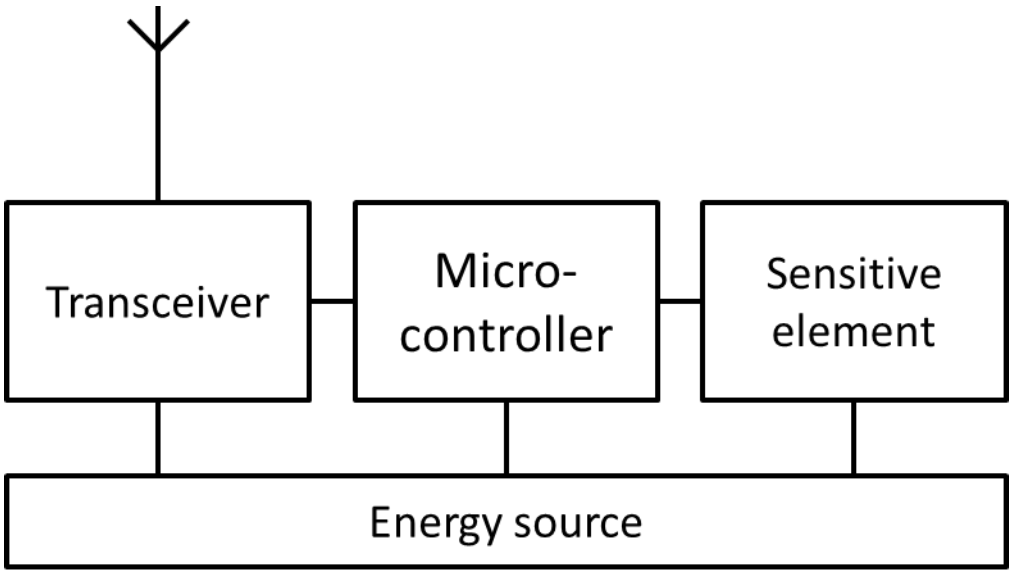


Figure 4-8Sensor node inner structure (ITU, 2014).

M2M architecture has three main features. First, it is a highly diversified tank components, ranging from low resources as sensors and powerful servers, these components can be distributed across a wide geographical area. Second, it emphasizes the greater autonomy in relation to the legacy Internet. M2M systems are designed to provide decentralization and minimizing the requirement for human intervention, more advanced systems can even apply functions situational awareness, self-organization or cognition. At last, M2M systems have adopted a communication modelthat is distributed, which can create the relationship between any two nodes, provided that one offers the service, or the resources which are necessary to the other node. For this aspect, the M2M systems broke the logical and topological simplicity of sensor networks (IEC Whitepaper, 2014).

Unlike what happens in wireless sensor networks, the communication path between two nodes must not follow a hierarchical path, e.g. from sensor to sink, and sink in remote management units. A sensor in an M2M environment will likely have direct communication with other sensors, regardless of the distance, the role and capabilities, provided that these relationships are desirable under M2M scenario. This new example, where nodes communicate with a large set of heterogeneous entities through a decentralized form, leads to situations where asymmetric resource capabilities between the two communicating entities (ITU, 2014).

4.7 Challenges in Wireless and Mobile Networks

In detail, the current wireless networks present challenges in terms of (ONF, 2013):

- Scaling: While the traffic from video continues to grow, static networks cannot cope with the challenges arising from the additional bandwidth required to meet this requirement.
- Management. At present, mobile networks are based on operational support systems (OSS) and management systems, demanding a high amount of expertise and resources for network operation. Due to the fact that these systems require manual intervention, the networks are prone to errors of configuration and delays in the delivery and resolving errors.
- Flexibility. Today's networks require weeks or even months to introduce new services due to manual procedures to enable service delivery and assurance. To multitenancy and isolation of traffic limited to such structures as virtual networks (VLANS), with limited policy management mechanisms.

- Cost: by inefficient and not as flexible use of network bandwidth and the growing complexity incurred staff operations, capital expenditure and especially functional, increases rapidly.

The challenges described above demonstrate the need for a new network architecture that overcomes limitations and addresses challenges of the current schemes.

4.8 Benefits of SDN in wireless networks

Networks defined by software, based on SDN architecture,provides a number of advantages for mobile networks, including wireless access segments, mobile backhaul networks and backbone networks (core) (ONF, 2013):

- SDN flow structure can be particularly suitable for the provision of communication from end to end across many different technologies such as 3G, 4G Wi-Fi etc. Scalar policies may be attributed to flows for effective traffic isolation and service quality management.
- The logically centralized control enables efficient coordination of the base station, which is particularly useful for addressing the interference between the cells as described in the following use cases.
- The SDN enables to optimize the path management based on individual needs for service and not routing configuration. This is especially important in Mobile environment where end users constantly change their position, the demands on bandwidth vary considerably based on the type of content sent and basic wireless coverage is not uniform.
- Network virtualization removes network services from their underlying physical resources. Multitenancy gives the capability to each isolated network resource section to have a separate policy, unrelated to the entity that

controls it. Such services can immediately be temporarily offered such as video feeds events on the news.

5 SDN FOR WIRELESS

5.1 OPPORTUNITIES

The limits on capacity is more difficult to overcome in the area of wireless networks and users often experience restrictions in data rate or service availability by using for example the cellular access network. The bandwidth for a given technology is a limited resource and is controlled by national regulations. Consequently, the capacity of a wireless technology is generally a level below the respective wire. Worldwide, wireless spectrum is not fully exploited, even though the groups of frequencies reserved for current or future use or for possible tactical communications.

The software-defined radio (cognitive radio) could allow the IP traffic to use unused or already allocated frequencies opportunist groups. However, the legal owner of the channel will have absolute priority and access and the secondary drive will need to be prevented when the main motion is detected.The spectral efficiency of several technologies is high. LTE reaches almost 80% of the Shannon limit. However, as the radio channels are shared among concurrent users, the increasing capacity in wireless users is still possible. Generally,the reduction of interference between cells through optimized network distribution channel, is realized by reducing the transmission range to form smaller cells. Both strategies are well known and efficient algorithms exist to optimize reuse and coverage in a network.

However, these techniques do not work anymore when many providers need to share the same channel. The lack of coordination among nearby access points limit optimization possibilities, as an access point selects the parameters based on the partial information transmitted from the operator or the communication between nearby access points. The situation where the terminals experiencing collisions between packets sent by access points that are both out of range remains unresolved.From the optimization point controller is the perfect location to be decided by the access points.

5.2 Applications of SDN in WLAN

According toJagadeesan & Krishnamachari (2014) the research in wireless SDN has focused in WLAN networks that operate based on the IEEE 802.11 protocol. The reason for this is that WLAN network devices "are not as closely tied tothe architecture as it is in the case of cellular networks" (Jagadeesan & Krishnamachari, 2014). However, with the omnipresence of high definition video content, and the demand of high quality of service, cellular companies are interested in offloading data traffic to WiFi networks whenever possible. Hence, research on WiFi SND and cellular SND may not be entirely independent. Network Protocol IEEE 802.16 (WiMax) has also been researched relative to the handover between WiFi and WiMAX on Openflow Wireless.

Following we present the major efforts on SDN for WLAN networks.

5.2.1 Openflow Wireless or OpenRoads

Yap et al. (2010) introduced Openflow Wireless as a vision of a world in which the users can change freely the wireless infrastructure they use and at the same time provide payment to the owners of that infrastructure in order to encourage continued investments. The authors argue that the best way of providing free movement across various infrastructures is to separate the network service from the physical infrastructure in order to allow for the implementation of innovative service from different stakeholders, such as researchers, operators, vendors and third parties. In this frame Yap et al. (2010) proposed an open, backward compatible, wireless network infrastructure to be implemented in college campuses. The implementation utilizes virtualization schemes to allow for experimentation on new services directly on the network. Openflow Wireless is a step towards enhanced openness of the mobile world ecosystem that allows for the user to choose among a wide range of networks and services.

The work of Yap et al. (2010) has been influenced by the structural barriers towards openness that are imposed by current implementations, i.e. the inaccessible and closed wireless capacity and the network infrastructure that is closed to innovation. The authors argue that even though there are multiple cellular and WiFi networks around us (especially in cities) they are all inaccessible due to restrictions from cellular companies and the demand of authentication of private WiFi networks. Also the use of IP that is very popular in today's mobile internet is not suitable for the future networks since it cannot support security and mobility, has a fixed architecture and does not allow for innovation.

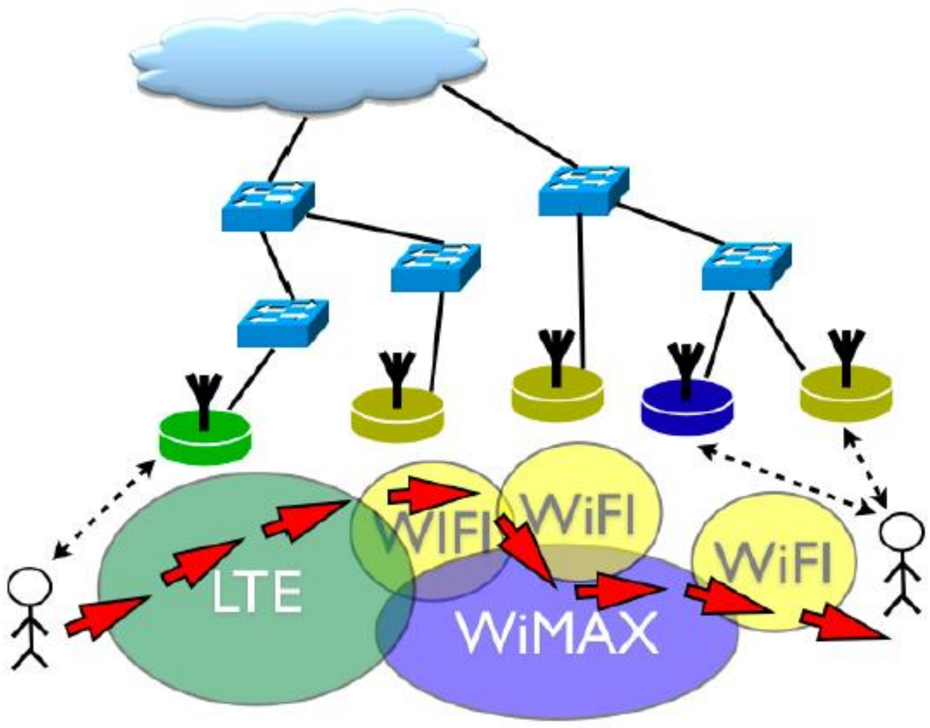


Figure 5-1Vision for FutureWireless Mobile Internet:Choice of providers, networks and technologies (Yap et al. 2010).

Figure 5-1 demonstrates the vision of openness of Yap et al. (2010). To support their vision the authors present Openflow Wireless, as a blueprint for an open network architecture. “OpenFlow Wireless uses OpenFlowto separate control from the datapath through an open API,FlowVisor to create network slices and provide isolation among them, hence allowing multiple experiments torun simultaneously in production wireless network, and SN-MPVisor to mediate device conguration access among experiments. These components which virtualize the underlying infrastructure relate directly back to our vision for futurewireless Internet design, in terms of decoupling mobility fromphysical network (OpenFlow), and allowing multiple serviceproviders to concurrently control (FlowVisor) and configure(SNMPVisor) the underlying infrastructure” (Yap et al., 2010). The architecture of Openflow Wireless (or otherwise called, OpenRoads) is depicted in Figure 5-2.

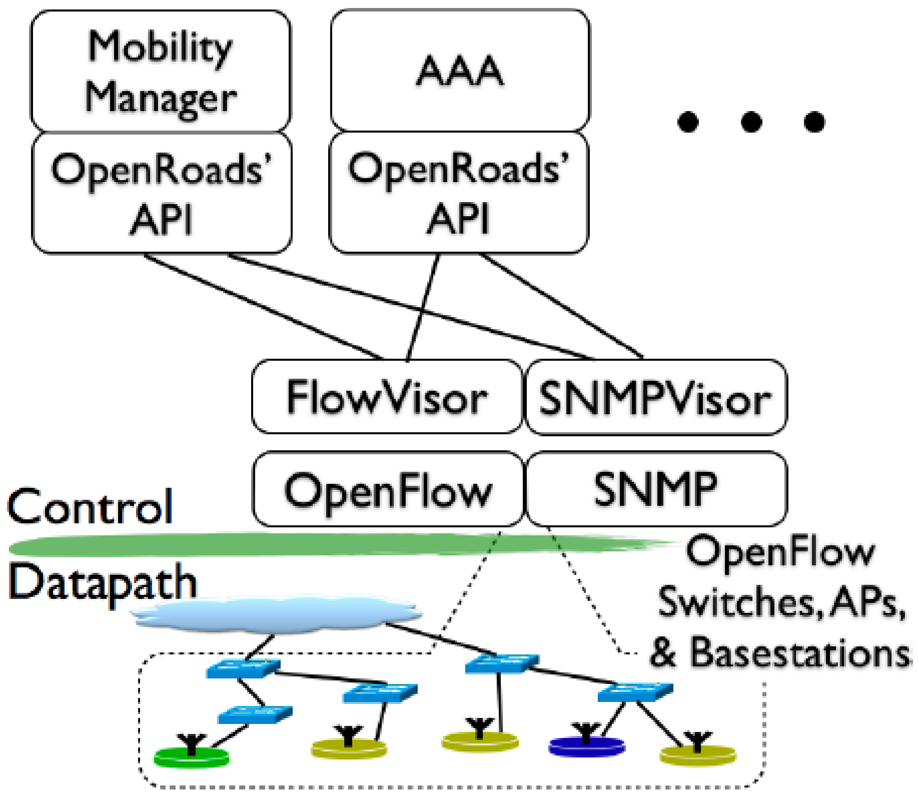


Figure 5-2OpenFlow Wireless (a.k.a. OpenRoads)Architecture (Yap et al. 2010).

In the Openflow Wireless environment, Openflow is added to WiFi Aps and WiMAX base stations by modifying their software, proposing that the same thing can be implemented in LTE and other cellular technologies. Remote controllers, one or multiple, running on a PC are responsible for controlling the network datapath. The freely available open source NOX controller is used but according to Yap et al. (2010), any controller is possible in principle. The Openflow Wireless architecture includes three different layers, namely flow, network slicing and controller. Openflow Wireless also implements various wireless technologies such as WLAN and WiMAX.

Application and plugins are hosted to the NOX controller in order to implement the new mobility manager. The mobility manager is notified about the movement of the user, picking the route and deciding whether to re-route the flow with vertical handoff among various radio networks. The additional layer of FlowVisor can slice up the network and give control to different controllers. In order to control the Openflow Wireless data path, a command line interface, SNMP or NetConf, is utilized. According to Yap et al. (2010) "We slice datapath configuration using a "SNMPVisor", thatruns alongside the FlowVisor, to allow an experimenter toconfigure his slice. FlowVisor slices the datapath, and SN-MPVisor slices the configuration by watching SNMP controlmessages, and sending them (and possibly modifying them)to the correct datapath element. Similar to FlowVisor, SNM-PVisor acts as a transparent SNMP proxy between the datapath and controllers, providing the same features of versioningand delegation".

5.2.2 Optimized Openflow Wireless for enhanced mobility

According to Sabbagh et al. (2013), the mobility among subnets and access points, while realized in Openflow Wireless, it is not optimized, since when a user moves to a different access point and base station, a new subnet and a new IP address are required, hence all flows must be updated. This function adds unwanted delays and reduces the quality of user experience. Sabbagh et al. (2013) propose a mobility enhancement control (new mobility management) to optimize the Openflow controller and provide control flexibility, virtualization and high level abstraction flow based routing and forwarding capabilities for mobile users.

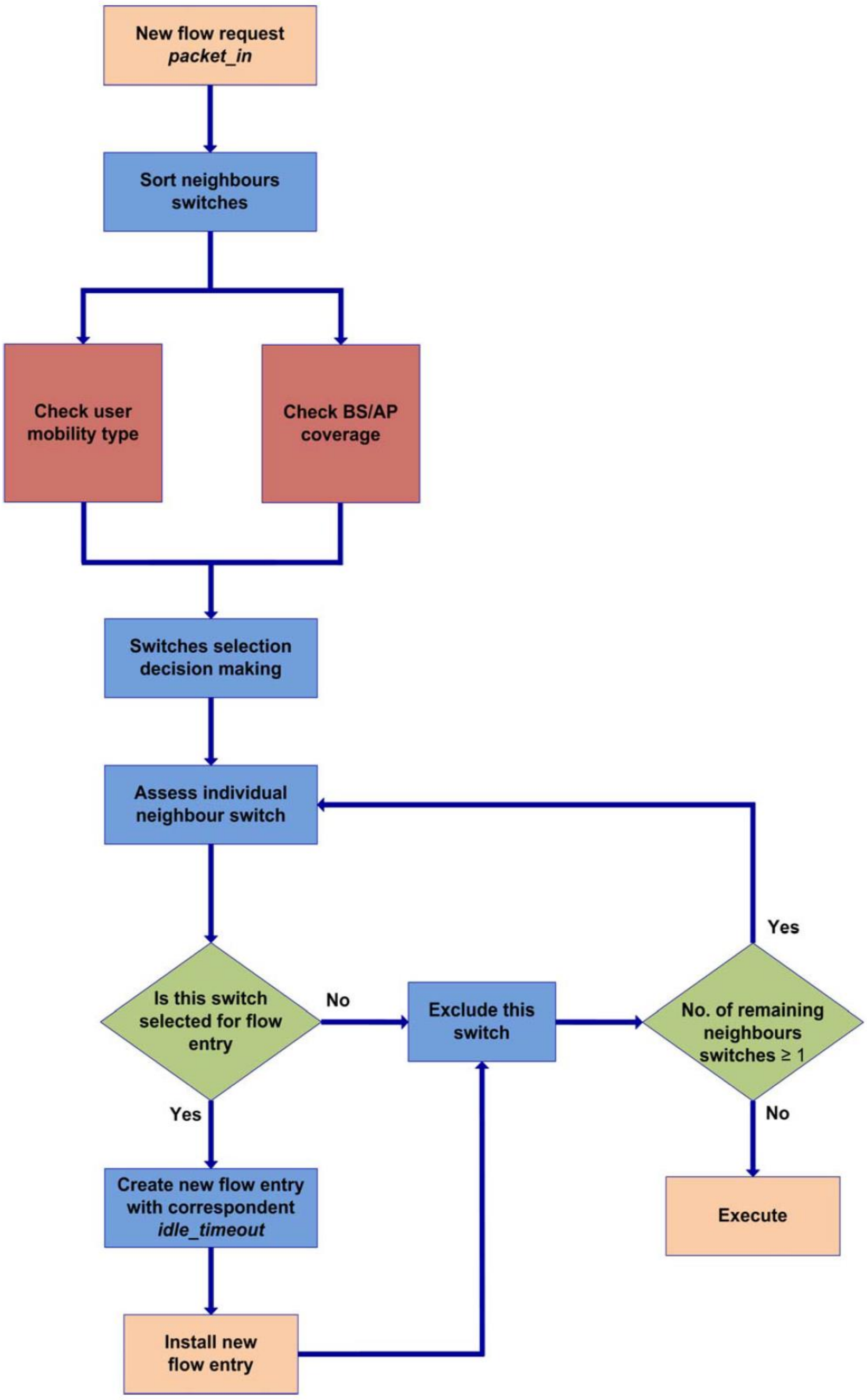

New flow request
packet_in
Sort neighbours switches
Check user mobility type
Check BS/AP coverage
Switches selection decision making
Assess individual neighbour switch
Is this switch selected for flow entry
No
Yes
Exclude this switch
No. of remaining neighbours switches ≥ 1
Yes
No
Create new flow entry with correspondent *idle_timeout*
Install new flow entry
Execute

Figure 5-3Proposed intelligent algorithm for enhancing mobility (Sabbagh et al., 2013).

The proposed algorithm for enhancing mobility, according to Sabbagh et al. (2013), is presented in Figure 5-3. As mentioned in Chapter 3, several switches and one controller are accommodated in an Openflow network. The controller undertakes the management of entries within these switches, by determining rules and apply actions on the way that the packets are handled, whenever a new flow request has beenreceived from the OpenFlow switch. Addition, deletion, modification and installation of flows to the appropriate switches are also responsibilities of the controller.

The algorithm presented in Figure 5-3, proposes that the flow is also installed in selected neighbor switches (not only to associated ones), to accommodate mobile terminals (MTs), and “reduce processing time for theneighbor switches in case of a MT has been moved to any ofthe access points or base station within these switches” (Sabbagh et al., 2013). The authors developed also amobility manager application is developed, in order to undertake decision making referring to the selection of neighbor switches for the installation of newflow entries.

According to the flowchart of Figure 3-5, upon receive of a new flow request, the controller willundertake evaluation and the creation and installation of flow entries to selected path switches. At the next step a list of of available neighbor switches will be developed by the controller and the mobility manager application will undertake information collection (velocity type, coverage range) relative to the neighbor switches. Next, the controller will decide upon the neighbor switches to create and install flow entries and allocate the appropriate idle-timeout that needs to becreated for each flow entry. As a next step,different idle-timeout flow entries are created forselected neighbor switches. At last, installation of the created flow entries into the selected switches and packet forwarding are implemented by the transmitted switch (Sabbagh et al., 2013).

5.2.3 Software defined radios

According to Jagadeesan & Krishnamachari (2014), Openflow Wireless and any advancement based on it, focus on the network layer and above. On the other hand Bansal et al. (2012) proposed OpenRadio to provide programmability to the PHY and MAC layers with the manner of defining a software abstraction layer hiding hardware details from the programmer. In this way, protocols are broken down to processing blocks and decision logic. The analog waveform is processed by processing blocks, while decision logic undertakes path mapping to traverse different processing blocks at varioustimes. It is not yet clear if this simplification can be applied to all wireless protocols, but there are advancements in WiFi protocols. According to Jagadeesan & Krishnamachari (2014) "key advantages claimed by OpenRadioinclude modular programmability and the ability for real-time implementation oncommodity digital signal processors".

Other efforts that offer programmable PHY and MAC layers are WARP and CloudMAC. CloudMAC proposes a network architecture that results in a programmable MAC layer without software radios, using virtual access points that are managed by virtual machines. The Openflow switch is connected both to the controller and to that virtual machines. The wireless termination points, i.e. the physical access points, do not produce their own MAC frames, but only send and receive MAC frames to the Openflow switch (Jagadeesan & Krishnamachari, 2014). The fact that the MAC frames should travel to the virtual access point increases delay time by three times. Nevertheless, these efforts show the way to a future were the entire wireless software can be modified according to the needs of the network.

5.2.4 Odin

According to Suresh et al. (2012), Odin is proposed as an SDN framework that simplifies the realization of WLAN services at enterprise level, namely authentication, authorization and accounting (AAA services) by the introduction of light virtual access points, similar to CloudMAC. This abstraction provides programmers with a constant virtual link of users to the access point, although the actual access point may change according to local decisions. Therefore, one physical access point may host multiple light virtual access points that correspond to each client. Odin is implemented on top of Openflow and consists of a single master, i.e. an application running on a controller and agents that run on physical access points. The master connect to each agent through a TCP channel. According to Jagadeesan & Krishnamachari (2014) “the advantages of moving to a centralized architecture,such as easy load balancing and mitigating the hidden terminal problem, arepreserved. Moreover, it is easier to implement mobility managers since the BSSID atthe user does not change during a handover”. The architecture of Odin is presented in Figure 5-4.

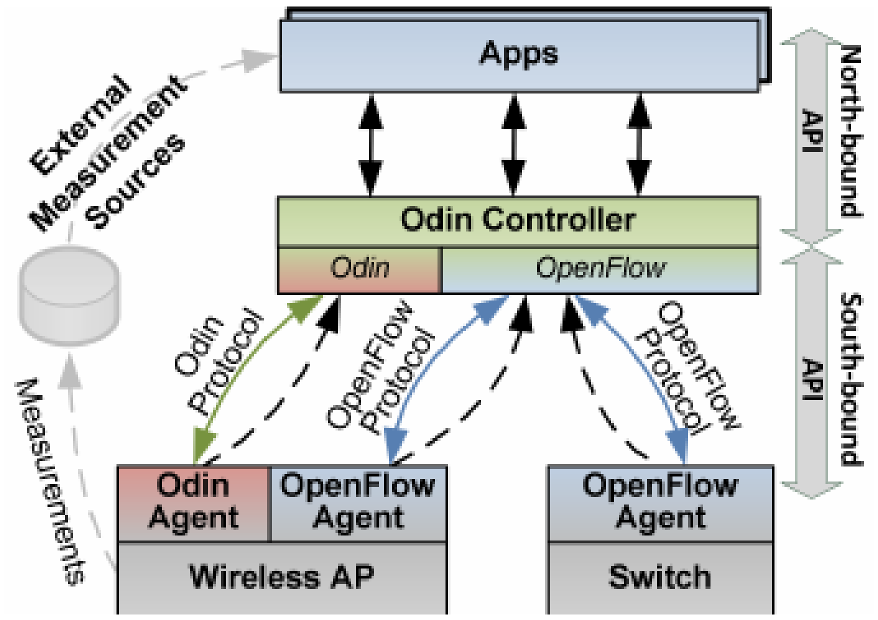


Figure 5-4High-level design of the Odin architecture (Schulz-Zander et al, 2014)

5.3 Applications of SDN in Cellular Networks

5.3.1 Inter-cell interference management

During the past years, there is a huge increase in telecommunications traffic with rapid adoption of smart mobile devices. The latest standard of wireless communications is LTE that gives the ability of high data communication speeds up to 150 Mbps and supports the growing need for broadband services. As LTE networks are growing and network traffic increases, the interference between the cells leads to a significant deterioration in the quality of service for mobile devices, as shown in Figure 5-5. Neighboring base stations, resulting in overlapping cells, have to synchronize the allocation of subcarriers to reduceunacceptable interference between users of mobile. The objective, as shown in Figure 5-5, is to reduce the signal to interference and noise ratio (SINR) by utilizing management techniques.

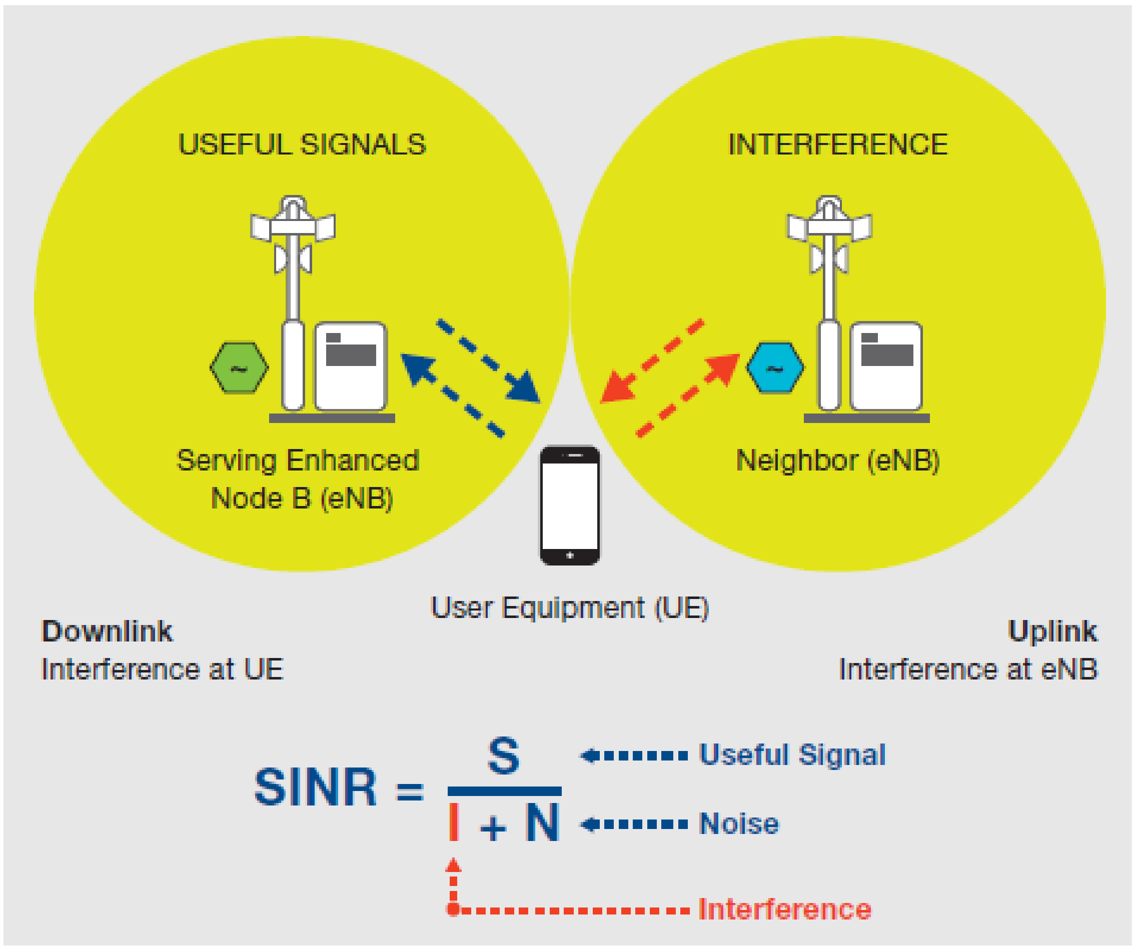


Figure 5-5 Reducing interference to increase SINR (ONF, 2013)

Multiple techniques are utilized in LTE networks to address the interference between cells, such as (ONF, 2013):

- Inter-cell interference coordination, which by selection decreases the power to sub-channels within the frequency range.
- Enhanced inter-cell interference coordination, where macrocells supplemented with picocells within their coverage area applying to public hotspots, i.e. cafes and airports.
- Coordinated multi point transmission / reception, where the interference is reduced for users at the ends of the cells, by accumulated programmingvarious cells with high levels of edge interference, or by accumulated transmission in order for the reception power and the experience of service users at the edge of the cell to be increased.

These interference coordination techniques present several limitations and drawbacks. Current LTE interference management between cells is implemented in a distributed manner. Interference coordination techniques such as coordinated multi point transmission / reception, are based on software, thus increasing the complexity and applyingng higher requirements on power and network resources in the radio access network (RAN). In addition, current interference between cells management algorithms are realized by employing distributed graph coloring algorithms, a solution which is complex and suboptimal.

The SDN LTE network provides the potential to overcome the limitations presented above, for interference management between cells, since the control layer is logically centralized and can make decisions for resource allocation for many base stations, offering global visibility. This implementation is more optimized compared to the management of distributed radio resource management (RRM), mobility management, and protocols/applications routing which are currently utilized (ONF, 2013). The centralization of network logic, enables the adjustment of distributed radio resource management (RRM) decisions based on dynamic power and subcarrier allocation in every base station. Upgrades in management can also be implemented unrelated to the base station equipment.

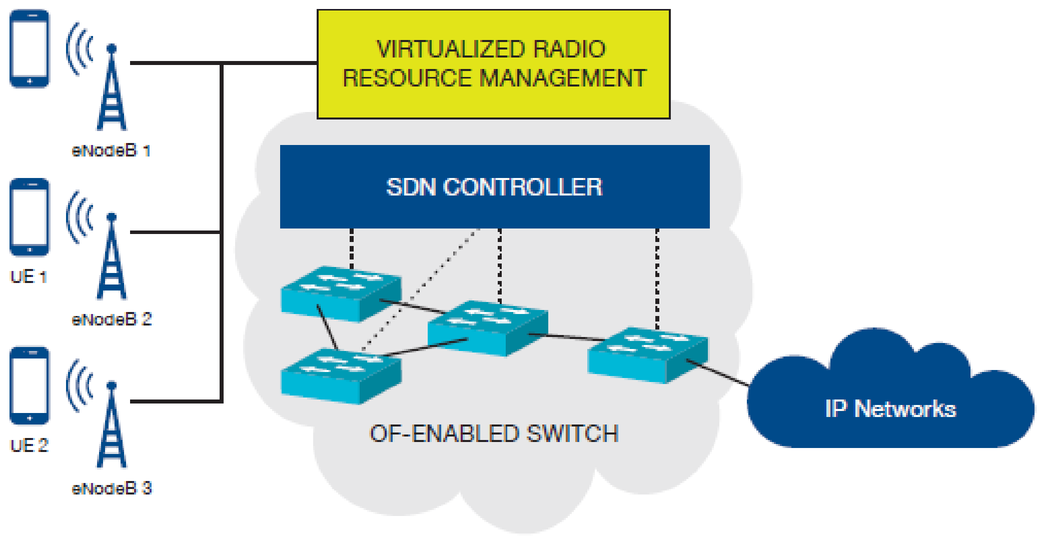


Figure 5-6OpenFlow-enabled centralized base station control for interference management (ONF, 2013).

5.3.2 Mobile traffic management

The techniques of traffic steering as well as path management, have attracted interest within the community of Openflow / SDN. According to ONF (2013), traffic steering applies to various areas and use cases include the balancing of load, filtering of content, control and implementation of policies, avoidance and recovery from disaster, particularly in anything that is related to rereouting and management of flows.

Relative to mobile and wireless networks there are actual applications, such as offloading and roaming of mobile traffic, adapting the content (including adaptive streaming), and optimizing mobile traffic, that could benefit greatly by Openflow implementation as SDN platform.

When mobile networks were mainly dominated by voice, the design of the RAN traffic was more predictable, but the transition to mobile data and the following rapid increase of sound and video have resulted in demands on bandwidth that are enormous and have risen with a greater speed than the budgets and revenue per user (average revenue per user, ARPU). Therefore, the area demands a more flexible approach to expand the capacity, to ensure the optimum use of RAN capacity, and to support the separation of services to maximize revenue.

Mobile traffic offloading, is the case where the SDN allows for placing and replacing traffic dynamically in mobile networks according to various criteria such as individual flow rate, cumulative flow rate or cumulative number of flows in a specific port / link, duration of flow, number of mobile users per base station, IP address, type of application and use of switches or ports (ONF, 2013). Criteria for dynamic positioning may also be defined by the user or network operator. These criteria may also be dynamically adjusted during monitoring form the network operator.

Offloading means that traffic is repositioned from the mobile network to a WiFi network, therefore it is also called WiFi roaming. In offloading the handover has to be very smooth, with minimum data loss and maintenance of IP connectivity, in order to achieve high levels of user experience. Offloading may also include the repositioning of traffic from a WiFi network with high congestion to a mobile network or other WiFi network. The SDN controller has to undertake monitoring and interaction functions to identify wireless networks in proximity to the mobile user for offloading, as presented in Figure 5-7. The selection of the roaming destination may be relative to a metricof quality of service (performance, signal strength, distance).

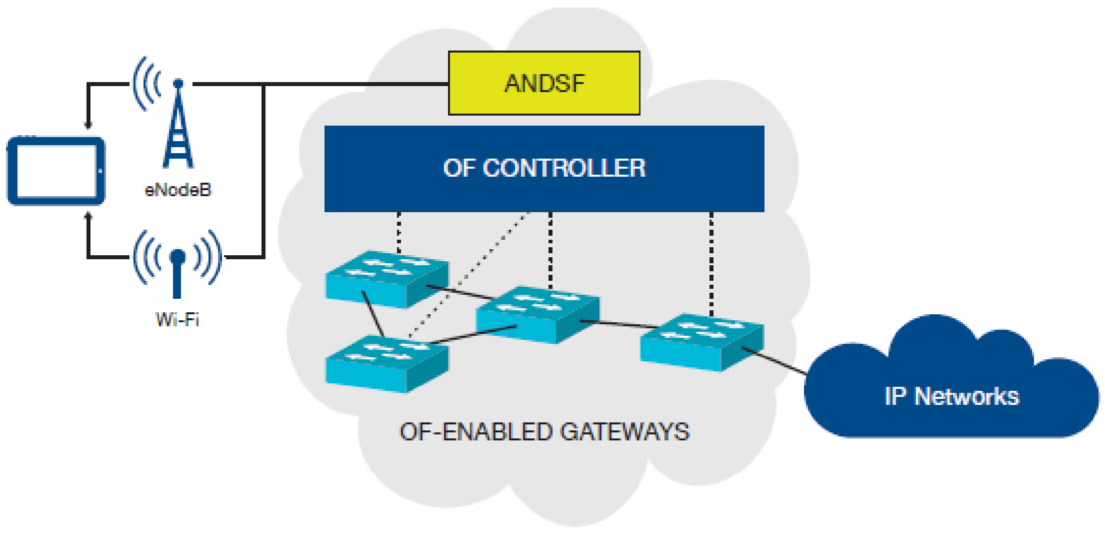


Figure 5-7Openflow-based mobile offload (ONF, 2013)

5.4 MULTI HOP WIRELESS NETWORKS AND SDN

5.4.1 Mesh networks

Mesh networks present frequent changes in topology, therefore the network nodes may be mobile or arrive and leave in high frequency. According to Dely et al. (2012), a new network architecture is proposed based on Openflow to enable mesh routers in accordance to the IEEE 802.11 standard. The same wireless interface is utilized for control and data traffic, while the functions are separated by using diversified SSIDs. The proposed architecture utilizes a NOX controller to manage and forward tables and to manage node mobility. Dely et al. (2012) conducted feasibility analysis and the results show that the increased complexity leads to deterioration of throughput. The complexity is increased because a complex rule is applied to match the MAC and IP address relative to a packet. This exact function resulted in throughput decrease by 15%.

5.4.2 Wireless sensor networks

Wireless sensor networks are multi-hop networks. The application of Openflow architecture may be beneficial to wireless sensor networks, as argued by Luo et al. (2012), since they present limitations that Openflow can overcome. These limitations range from low powered nodes that are utilized without care relative to the topology and are inflexible in policy changes, meaning that a change in policy may result to manual reconfiguration or reprogramming of the whole wireless sensor network. Also wireless sensor networks are hard to manage due to the intrinsic difficulty of management system development and the fact that up to now, management systems are implemented in already developed wireless sensor networks that need to be hacked in existing codes to implement the changes. Hence, each node of the wireless sensor network behaves as an independent entity performing networking operations to forward data and control the network. Therefore, Luo et al. (2012) argue that increased complexity and lack of abstraction in the existing wireless sensor networks create opportunities for the implementation of Openflow. Luo et al. (2012) propose this implementation claiming the following advantages:

- Versatility – the data plane supports basically every type of packet forwarding and the control plane can decouple applications to provide a global unified perspective of the network to the applications, hence sensors can be customizable in terms of applications and support them in a plug and play manner.
- Flexibility – centralization and granularity of network control in software defined networks supported by user customizable flow tables makes it easy to apply policy changes, which is very difficult in existing wireless sensor networks.
- Ease of management – the network management system can be implemented as yet another application on top of the WSN-SDN architecture's control layer. This is implemented with the use of open APIs without the need to change the existing code.

Luo et al. (2012) modified Openflow to support wireless sensor network use cases, by mainly accommodating sensor attributes to create flow tables. Nevertheless, control and data traffic are implemented in the same network resulting in increased average latency.

6 CONCLUSIONS

Software Defined Networks (SDN) are presenting new opportunities through centralized and automated control and separate and direct programming capability of network to provide dynamic, cost effective and manageable implementations. Due to the diversification of traffic patterns, cloud computing, big data and the rapid increase in personal IT devices, existing networks exhibit limitations such as complexity, policy inconsistency, scaling inability and vendor dependence. The SDN concept provides separation of control and data plane, control software that may be executed on general purpose hardware, programmable data planes and centralized control of the whole network. SDN architecture consists of the infrastructure (data), control and application layers and the Openflow protocol is responsible for enabling communication among data and control plane.

The new SDN architecture can offer better utilization of network resources, reducing costs for operators, and increasing the quality of service, in both wired and wireless networks. This is due to the fact that Openflow protocol enables the development and evaluation of networking techniques on top of the network architecture, meaning that already there is no interface with existing routing and security protocols. Hence, researchers and developers may test new network systems without changing the existing network infrastructure.

The application of Openflow in wireless networks may address challenges currently presented, such as limited scaling, difficult management, reduced flexibility and high costs. In this frame, adaptation of SDN architecture in wireless systems can be particularly advantageous in terms of flow structure to provide communication among users with different network technologies, efficient coordination of base stations, path management optimization and isolation of network services from physical resources in allow for separate policies and rules that can be scalable.

Up to now, many efforts have been made by the scientific community to implement Openflow architecture to wireless networks. WLAN networks have been the most popular in terms of Openflow adaptation, with the Openflow Wireless (OpenRoads) implementation, that aims to provide to the user free movement across various network infrastructures, i.e. LTE, WiFi, WiMAX etc. Based on this concept several mobility optimization techniques have been introduced to enhance quality of experience and reduce delays from the transition among different subnets and access points. On the other hand, OpenRadio provides programmability to the PHY and MAC layers with the manner of defining a software abstraction layer hiding hardware details from the programmer. Also, Odin is proposed as an SDN framework that simplifies the realization of WLAN services at enterprise level, namely authentication, authorization and accounting the introduction of light virtual access points. Application of SDN to cellular networks include inter-cell interference management and mobile traffic management basically by providing mobile traffic offloading techniques. Multi-hop wireless networks have also been studied as use cases for Openflow protocol, but increased complexity is so far a limitation for further deployment. The application of Openflow to wireless sensor networks may provide versatility, flexibility and ease of management by modifications in the actual Openflow protocol to take into account sensor attributes. So far the implementations present increased latency.

REFERENCES

Samsung (2015). The average Greek household has 19 devices, according to a survey by Samsung. Available online: http://www.kathimerini.gr/824755/article/texnologia/gadgets/to-meso-ellhniko-noikokyrio-dia8etei-19-syskeyes-symfwna-me-ereyna-ths-samsung [Retrieved June 1st, 2016]

Larsen, K.K. (2012). Capacity planning in mobile data networks experiencing exponential growth in demand. Available online: http://www.slideshare.net/KimKyllesbechLarsen/capacity-planning-in-mobile-data-networks [Retrieved June 1st, 2016]

Zhang, Y. Zhao, J. (2013). Software Defined Networking (SDN) Enabled Optical as a Service(OaaS) with Dynamic Network Provisioning. PIERS Proceedings, Guangzhou, China, 362-367.

Azodolmolky, S. (2013). Software Defined Networking with OpenFlow. Available online: http://file.allitebooks.com/20160314/Software%20Defined%20Networking%20with%20OpenFlow.pdf [Retrieved June 1st, 2016]

Stallings, W. (2012). Software-Defined Networks and OpenFlow. The Internet Protocol Journal, 16(1). Available online:http://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-59/161-sdn.html [Retrieved June 1st, 2016]

Karasiewicz, C. (2014). Network design approach for new trends in user computing. Available online: https://www.ibm.com/developerworks/community/blogs/mobileblog/entry/network_design_approach_for_new_trends_in_user_computing?lang=en [Retrieved June 1st, 2016]

Little, R.G. (2014). Will Meru's SDN wireless LAN replace the need for unified wireless? Available online: http://searchsdn.techtarget.com/news/2240223706/Will-Merus-SDN-wireless-LAN-replace-the-need-for-unified-wireless [Retrieved June 1st, 2016]

Armbrust M., Fox A., Griffith RJoseph, ., A. Katz, R. Konwinski, A. G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. (2009). Above The Clouds: A Berkeley View Of Cloud Computing.

Mell, P. and Grance, T. (2009). Draft NIST Working Definition of Cloud Computing.

Scudder, R. (2011). Visualizing the Workings of Cloud Computing With Diagrams. Available online: http://www.brighthub.com/environment/green-computing/articles/127086.aspx [Retrieved June 1st, 2016]

Cryptum (2016). Virtualization. Available online: http://cryptumlimited.com/services/service_specific/virtualization [Retrieved June 1st, 2016]

Rajasri, K. Srikanth, K. Kingston S. & Bhaskar, R. (2011). SDN and OpenFlow: A Tutorial. IPinfusion. Available online: https://www.clear.rice.edu/comp529/www/papers/tutorial_4.pdf [Retrieved June 1st, 2016]

Limoncelli, T.A. (2012). OpenFlow: A Radical New Idea in Networking. Communications of the ACM. Available online: http://queue.acm.org/detail.cfm?id=2305856 [Retrieved June 1st, 2016]

Kerner, S.M. (2012). OpenFlow Protocol 1.3.0 Approved. Enterprise Networking Planet. Available online: http://www.enterprisenetworkingplanet.com/nethub/openflow-protocol-1.3.0-approved.html [Retrieved June 1st, 2016]

Pavlidis, A. (2015). Design and development of data collection mechanisms in experimental platforms for the Internet of the future. Diploma Thesis. NETMODE - NETWORK MANAGEMENT & OPTIMAL DESIGN LABORATORY, NTUA.

Voukelatou, O.S. (2015). Openflow controller for Environmental AwarenessServices. Diploma Thesis. DEPARTMENT OF COMMUNICATIONS, ELECTRONICS AND INFORMATICS, NTUA.

Cristoforou, R. (2014). Software development using OpenFlow controller (OF controller) and implementation of packet routing mechanisms.Diploma Thesis. DEPARTMENT OF COMMUNICATIONS, ELECTRONICS AND INFORMATICS, NTUA.

Erickson, D. (2014). The Beacon OpenFlow Controller. Stanford University.

Kallipos, (2015). Modern wireless communication systems. Available online: https://repository.kallipos.gr/bitstream/11419/4588/1/02_chapter_07.pdf

Stathopoulos, C. (2011). Providing quality of service guarantees to 802.11 wireless networks. Diploma Thesis. DEPARTMENT OF COMMUNICATIONS, ELECTRONICS AND INFORMATICS, NTUA.

Andrews, J. G., Ghosh, A. & Muhamed, R. (2007). Fundamentals of WiMAX: Understanding Broadband Wireless Networking, USA: Prentice Hall.

Theologou, M. (2010). Networks Mobile and Personal Communications (2nd Edition). Thessaloniki: Ed. Tziola.

Kanatas, A., Constantinou, P. & Pantos, C. (2013). Mobile Communication Systems (2nd Edition). Athens: Ed. Papasotiriou.

Holma, H. & Toskala, A. (2010). WCDMA for UMTS: HSPA Evolution and LTE (5th Edition). USA: Wiley.

Tse, D. & Viswanath, P. (2005). Fundamentals of Wireless Communications. USA: Cambridge.

IEC Whitepaper (2014). Internet of Things:Wireless Sensor Networks. Available at: http://www.iec.ch/whitepaper/pdf/iecWP-internetofthings-LR-en.pdf

ITU (2014). SERIES Y.2000:NEXT GENERATION NETWORKS. Technical paper. International Telecommunication Union.

ONF (2013). OpenFlow™-Enabled Mobile and Wireless Networks. ONF Solution Brief, September 30, 2013.

ONF (2012). Software-Defined Networking:The New Norm for Networks. ONF White Paper, April 13, 2012.

Yap et al. (2010). Blueprint for Introducing Innovationinto Wireless Mobile Networks. VISA 2010, September 3, 2010, New Delhi, India. ACM 978-1-4503-0199-2/10/09.

Sabbagh et al. (2013). Optimization of the OpenFlow Controller in WirelessEnvironments for Enhancing Mobility. 13th Annual IEEE Workshop on WLAN.

Jagadeesan N.A. & Krishnamachari, B.(2014). Software Defined Networking Paradigmsin Wireless Networks: A Survey ACM Comput. Surv. 47, 2, Article 27 (December 2014), 12 pages.

Bansal, M. Mehlman, J. Katti, S. & Levis, P. (2012). OpenRadio: a programmable wirelessdataplane. In Proceedings of the first workshop on Hot topics in software defined networks (HotSDN '12).ACM, New York, NY, USA, 109–114. DOI:http://dx.doi.org/10.1145/2342441.2342464

Suresh, L. Schulz-Zander, J. Merz, R. Feldmann, A. & Vazao, T. (2012). Towardsprogrammable enterprise WLANS with Odin. In Proceedings of the first workshop onHot topics in software defined networks (HotSDN '12). ACM, New York, NY, USA, 115–120.DOI:http://dx.doi.org/10.1145/2342441.2342465

Schulz-Zander, J. Suresh, L. Sarrar, N. & Merz, R. (2014). Programmatic Orchestration of WiFi Networks. Conference Paper. Available at: https://www.researchgate.net/publication/263010363_Programmatic_Orchestration_of_WiFi_Networks

Dely, P. Kassler, A. & Bayer,N. (2011). OpenFlow for Wireless Mesh Networks. In Computer Communicationsand Networks (ICCCN), 2011 Proceedings of 20th International Conference on. 1–6.DOI:http://dx.doi.org/10.1109/ICCCN.2011.6006100

Luo, T. Tan, H.P. & Quek,T.Q.S.(2012). Sensor OpenFlow: Enabling Software-Define Wireless Sensor Networks. Communications Letters, IEEE 16, 11 (2012), 1896–1899.DOI:http://dx.doi.org/10.1109/LCOMM.2012.092812.121712

Rao, S. (2014a). SDN Series Part Three: NOX, the Original OpenFlow Controller. Available at: http://thenewstack.io/sdn-series-part-iii-nox-the-original-openflow-controller/

Izard, R. (2016). Floodlight Controller. Available at: https://floodlight.atlassian.net/wiki/display/floodlightcontroller/The+Controller

Rao, S. (2014b). SDN Series Part Four: Ryu, a Rich-Featured Open Source SDN Controller Supported by NTT Labs. Available at: http://thenewstack.io/sdn-series-part-iv-ryu-a-rich-featured-open-source-sdn-controller-supported-by-ntt-labs/

Sridhar, T. (2008). Wireless LAN Switches Functions and Deployment. The Internet Protocol Journal, 9(3). Available at: http://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-13/wireless-lan-switches.html