SDN: Layers, Architecture and Perspectives of a Key Technology for the Future Internet

Christian Tipantuña[1,2], Carla Parra[1], Jorge Carvajal[1], Ricardo Llugsi[1]
[1]Departamento de Electrónica, Telecomunicaciones y Redes de Información, Escuela Politécnica Nacional (EPN), Ladrón de Guevara, E11-253 Quito, Ecuador.
{christian.tipantuna, carla.parra, jorge.carvajal, ricardo.llugsi}@epn.edu.ec
[2]Department of Network Engineering, Universitat Politècnica de Catalunya (UPC), C. Jordi Girona 1-3, E-08034 Barcelona, Spain

***Abstract*—Software Defined Networking (SDN) refers to a networking technology that provides abstraction of physical resources and network programmability by separating the control plane and the data plane, enabling innovation and a simplified network management. This technology has revolutionized the world of the networked systems, has become an important networking research topic in academia and industry during the past couple of years, and nowadays is considered as a key technology for the deployment of current and future communication systems and as an enabler for the future Internet.**

Despite the growing popularity of SDN in the last decade and the constant scientific divulgation of the research work done, to get involved with this technology, to know its characteristics and capabilities, and in order to perform research in this interesting but extensive field, is necessary to know in advance some concepts, as well as the behavior of the different components that are part of this networking paradigm. In this insight, this paper addresses these requirements and more, providing a concise information about the concepts, fundamentals and perspectives of SDN. In this work is described the structure of traditional networks in order to contrast the changes, the benefits and the advantages that the SDN paradigm can offer. Further, is presented a chronological evolution of the technology, the levels of abstractions, the elements, the architectural framework as defined by the Internet Research Task Force (IRTF) in the RFC7426, and some fields of application. At the end, a list with the information of more than 130 SDN open-source projects is also provided.

***Index Terms*—Control plane, data plane, programmable networks, rfc7426, software defined networking.**

I. Introduction

The networking technology that is being used in our world is mostly built on the concepts introduced by the first networking research projects, such as ARPANET (Advanced Research Projects Agency Network) [1], and over the years, it is evident that the evolution of computer networks has been a great success, starting from a research project to exchange information between universities [1] and becoming a global communications infrastructure, which at every moment serves to more and more users and which increases in size and complexity.

Although the network technologies used in these decades (traditional technologies) have been very useful and have worked properly, the techniques, the requirements and demands, from both industry and users, as well as the hardware and the software, have evolved over time. Thus, in order for computer networks can fit to the current and futures needs, innovations have been developed mainly in the upper layers of the underlying network and in the end devices, but the network architecture has been the same over the years. Originally, computer networks were designed to provide interconnection between network devices considering principles such as: heterogeneity, interconnection and sharing, but other aspects such as: availability, mobility, scaling, security, quality of service (QoS), quality of experience (QoE) and QoX (where X stands for service, network economics, energy, etc.), among others, were left out, because those aspects were not important in the era that technology was born and because nobody thought the impact that would have the computer networks in our current society.

Networking solutions have been developed by vendors and operators, and although these solutions have allowed a global connectivity, they have certain features that have not given way to a more rapid technological evolution. The operation of a network device is closely linked to the design and degree of innovation of the vendor, namely it is a closed device (black box). Every vendor has its own software solution, which is included in the hardware, as well as specific network interfaces, protocols and management systems. Nowadays, managing a multi-vendor network is a very difficult and complex task, not to mention that there is a slow standardization of protocols and delay in introducing new features, also the networks should be able to transfer an increasingly large amount of information (big data) [2]. To cope with the challenges aforementioned, to meet the new services requirements and in order to promote innovation and open ecosystems (white-box devices), both academy and industry have considered to upgrade or redesign the network architecture. These changes have not been taken at random, on the contrary have been the result of decades of research and a collaborative effort of many institutions and organization [3].

A key point in the evolution of networked systems is the advent of virtualization technologies, because they changed the landscape in the compute domain. The idea of abstracting the software from the underlying hardware and the capacity to produce changes in the behavior of a system (virtual machine) based in changes in software components, were the starting

point to development of programmable networks, and later the birth of Software Defined Networking (SDN) technology.

At the moment SDN is an important topic in networked systems, and a large number of research works have been done in this area and in different fields. SDN covers a huge number of aspects that range from the hardware level to the software level, for this reason and as a basis for other research works in this paper are presented the most important concepts, definitions and fundamentals. However, in [4] is presented a more detailed state-of-the-art.

The rest of the paper is organized as follows. The description of the operation of a traditional networks is provided in Section II. The abstraction levels, the elements and the SDN architecture are presented in Section III, followed by the perspectives and some application fields in Section IV. Finally, the conclusions are presented in Section V.

II. Traditional Networks

A traditional network device or a network element has three well-defined planes, which are: data, control and management planes [5], as show in Fig.1. The description of these planes is given below.

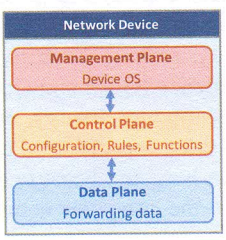


Fig. 1. Data, control and management planes in a non-SDN network device

- **Data Plane:** The data plane is also called the data path or the forwarding plane. The data plane processes traffic or data streams (data packets), and it is responsible for handling and applying actions to them, based on rules that were programmed into lookup-tables (forwarding tables). Examples of protocols in this plane are: Ethernet and MPLS.
- **Control Plane:** The control plane is connected to the data plane to update the logic that should be implemented on the data streams, this plane is tasked with calculating and programming actions for the data plane, namely how and where to send the network traffic. The control plane is where the forwarding decisions are made and where other functions such as: QoS, Virtual Local Area Networks (VLANs), etc. are implemented. Examples of protocols in this plane are: OSPF, ISIS and BGP.
- **Management Plane:** The management plane is where it is possible to configure and monitor the network device (switch or router), through a command prompt (shell) or a web interface. The management plane usually runs on the same processor as the control plane.

In a traditional communications network as shown in Fig.2, composed of network devices and physical and logical connections, every network element contains the management, control and data planes. The software is tied to the hardware where it runs, and the control and the modification of the data, which is forwarding across the network, works within a proprietary framework, which means that the user is directly and indirectly linked to the solutions provided by the vendor, and to adopt a new solution, protocol or feature can be difficult, expensive or it can take a long time.

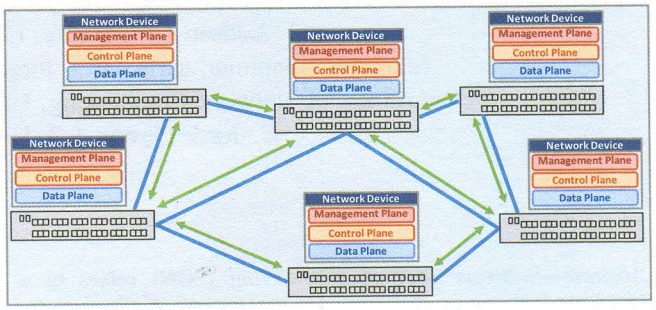


Fig. 2. Traditional network topology

The control plane within each device fulfills functions such as: network device configuration, management and exchange of routing information tables, and collection of information about network topologies and network status. The information used by the control plane, to take decisions about where to send the traffic, can be learned through the use of static routing or through the use of distributed routing protocols (e.g. OSPF, ISIS, BGP). Routing techniques allows to any router to exchange information with each other device within the network (distributed algorithms running between neighbors), in order to build routing tables that will be used by the data plane. Based on the status of the network topology, the information of all devices is updated in a distributed manner, and any device can take decisions or perform actions depending the requirements or goals to be achieved. But since all devices have the same level, it is difficult to understand the state of the network, its history and to decide which decision or action is best and who should execute it, in the traditional network approach, the distributed routing protocols in many cases do not take optimal decisions, because all devices have the same perspective and no device has a complete view of the network, this lack of information does not allow to take end-to-end optimal decisions. In order for solving aforementioned drawbacks, SDN technology has emerged.

III. Software Defined Networking (SDN)

SDN technology has been and continues to be a strong research point in the field of computer networks. An example of the growing and constant interest in the research area can be seen in Fig. 3 which, taken from Google trends, represents the percentage of searches about SDN topics since 2004.

The SDN paradigm as it is known today, has been the result of several years of research and the joint effort of industry and academia. Some of the research projects which led to the conception of a programmable network infrastructure, were: RCP (Routing Control Platform), 4D architecture, Ethane and OpenFlow [7]. A chronological evolution of SDN is presented

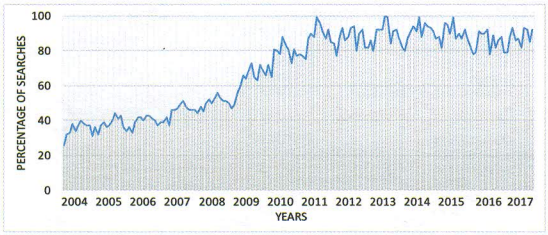


Fig. 3. Search interest about SDN, data taken from [6]

in Fig. 4, where are shown the most representative projects and initiatives, which led to the emergence and growth of SDN. Moreover, a detailed historical perspective is available in [3].

SDN decouples control and forwarding functions separating the control and data plane [4]. The separation of planes allows that the control plane becomes directly programmable and accessible by an application layer through an application program interface (API), and as a result underlying network layer can be abstracted from application and network services. SDN technology changes the paradigm of networks, unifying the behavior of network devices (routers and switches), and turning them into generic forwarding devices (SDN switches or white boxes), which can be implemented using open standards and vendor-neutral solutions. The functionality of the entire network can be programmed instead of programming individual devices, because the application is independent of the underlying hardware. The programmability offered by the SDN technology makes it possible to make changes in the network, which can range from tasks related to the routing, or even a complete change in the behavior of the network. All kinds of network applications can be now programmed, for instance: change configuration of network devices, give priorities to network traffic flow, give access or permission to the network properties, etc. Definitely, the programmability of the network introduced by a SDN enables innovation and accelerates the introduction of new features and services.

SDN transforms the network perspective from specialized hardware with protocols and applications implemented for each network device (switch, router) to an open ecosystem which promotes an independent evolution of hardware and software solutions.

A. Abstractions in SDN

One of the most important characteristics of SDN technology is the abstraction. The application does not send codes directly to the network devices, instead it communicates with the controller, and thanks to the abstraction, the network infrastructure is transparent to the application. The main levels of abstractions in a SDN network are listed below.

- **Network abstractions:** A global view of the network is available from controller. The behavior of the entire network based on the requirements of the applications and the decisions and actions are taken by the controller.
- **Control plane abstractions:** The controller must be compatible with any hardware and software solution that is part of the infrastructure layer. The controller makes decisions based on requests by the applications, and taking into account the network status and topology.
- **Data plane abstractions:** The network infrastructure is independent of software and hardware solutions. The configuration of the network devices can be based in terms of flows, as it is proposed by the OpenFlow protocol [7], namely the traffic of the data will be based on flow-based forwarding instead of destination-based forwarding, in this context a flow is a set of packet field values acting as a match rule and a set of actions to operate on all packets belonging to the same flow.

B. Elements of a SDN Network

To understand the changes in the network architecture, the features and advantages that it could bring, it is suitable to analyze the structure and operation of the SDN-compatible network devices. An example of the structure of SDN devices is shown in Fig.5, where the devices are a controller and a switch.

As shown in Fig.5, unlike a traditional network element where the control and data planes are implemented in the hardware platform as an integrated system, see Fig.1, in a SDN network there is a separation between the control and the data planes, which is given by using a SDN controller and a SDN switch, respectively. Although these devices have different functionality compared to traditional ones, this does not limit that the network devices can be upgraded and can be compatible with SDN networks, there are currently network devices that can work in conventional and SDN environments. In addition to the SDN controller and the SDN switch other component that the SDN architecture introduces is the network applications (services), as shown in Fig.6.

A description of the main features of the application, the controller and the switch is given below.

- **SDN switch:** The switch can be physical or virtual (network device) and does not have to implement all protocols or features, it has an abstraction layer, which allows to perform minimum control and management functionalities, and which will be used in the communication with the controller through the southbound interface, namely using a control-switch protocol, such as OpenFlow [7]. The control and management functionalities in the switch are also needed for cold-start operation and configuration. The SDN switches are also called white boxes because they can be developed based on open-source standards and solutions.
- **SDN controller:** The decisions are made in the control plane, which is part of the controller, which offers a central management of all devices within the network and which has an entire view of the network, as shown in Fig.6. The SDN network controller hides the network complexity and it understands the constraints in the network. The controller software communicates with both the network application and the underlying network through interfaces, the northbound and southbound interfaces, respectively. When the control plane is separated from individual network devices (network infrastructure),

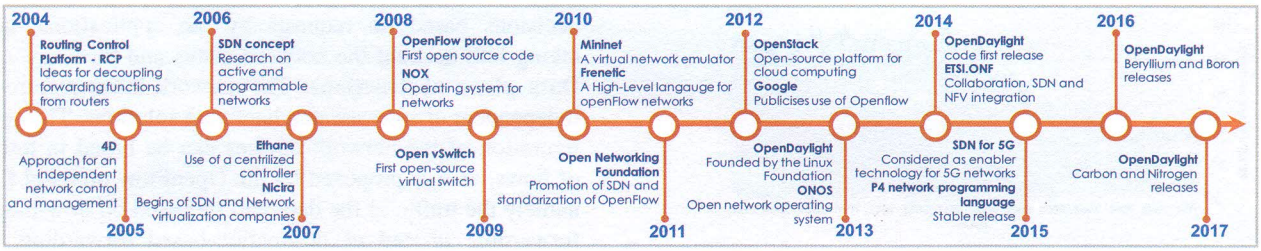


Fig. 4. SDN timeline, adapted from [3] and [8]

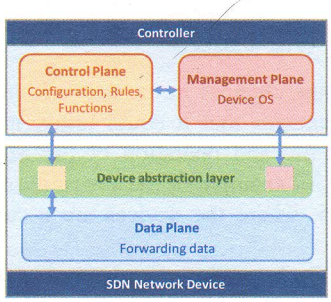


Fig. 5. SDN network devices: controller and switch

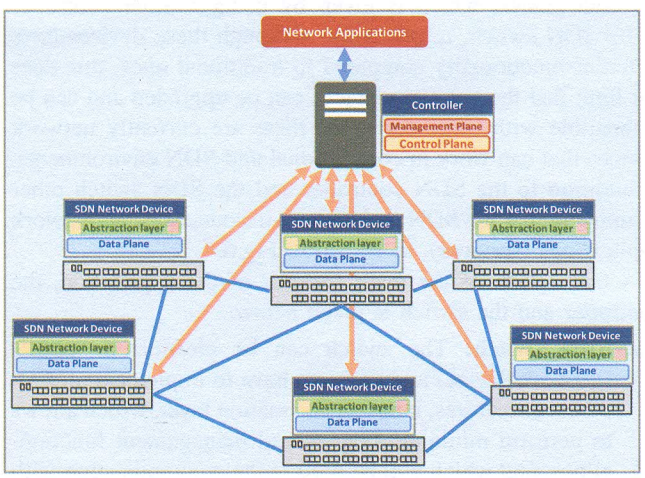


Fig. 6. A SDN network with one controller and multiple SDN-switches

and it is centralized in a single device (SDN controller), it is able to perform a better management of multiple network elements (SDN devices), and it can provide underlying network abstraction, which facilitates the deployment of network applications (programmability). The SDN controller has the following characteristics:

- Logically centralized software platform. It offers a unique abstract view of the network state (e.g. topology).
- The SDN controller is sometimes referred to as the Network Operating System (NOS) [9], because it acts as such. Network applications run on the SDN controller, similar to computer applications running on computer operating system.
- For scalability and reliability reasons, to support very large and complex networks, the controller can be distributed in different servers.
- There are commercial and open-source solutions. Examples of open-source controllers are: Beacon, Floodlight, Trema, NOX, POX, Ryu and OpenDaylight [10], the latter being one of the most relevant projects today. The controller should truly interoperable and multivendor and its operation cannot be tied to a specific network element.
- The data flows can be dynamically adapted by the controller.
- The controller has APIs (open) to interact with both applications and data plane.

- **Network applications:** The network applications running on the controller, these applications can be diverse and can range from routing tasks to the implementation of network services such as: network access control, load balancers, firewalls, etc. The application communicates with the controller using the northbound interface, examples of this interface are: REST API and Java API.

C. SDN Benefits

There are significant benefits related with SDN technology, among the most important ones are the abstraction capability and the level of programmability. The control plane becomes directly programmable through the use of APIs and underlying network can be abstracted from applications and network services, allowing a programmatically control of network resources. Other benefits of SDN technology include:

- Dynamic network behavior and a central view for running network services and applications.
- Better utilization of network resources (physical or virtual).
- Improvements in network performance, making the network more manageable, cost-effective and adaptable to ongoing dynamic requirements.

D. SDN Architecture

The Open Networking Foundation (ONF) was founded in 2011 to promote SDN through the development of open standards, such as OpenFlow, and to provide an architectural framework [11], defining the roles of application, control and data planes, as well as the interfaces to interconnect them.

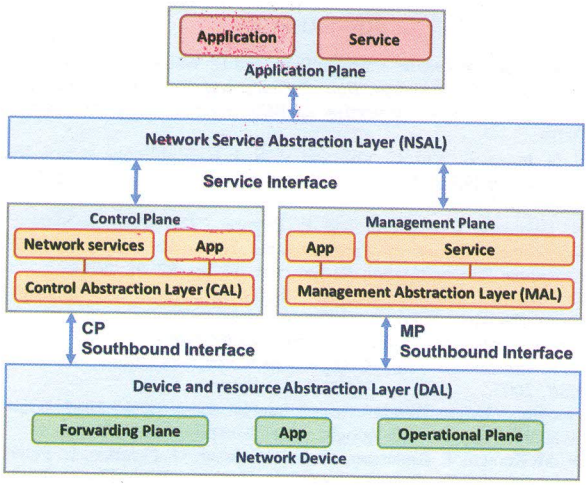


Fig. 7. SDN Architecture, adapted from [12]

Although, in [11] the description of the layers is presented in a friendly way, with the passage of time and with the increasing popularity of SDN, a bit of confusion was created regarding the layers, interfaces and the architecture itself, so that in 2015 as a response to clarify the SDN concepts and terminology, the Internet Research Task Force (IRTF) through its Software Defined Networking Research Group (SDNRG) worked intensively in a proposal, giving as result the RFC7426 [12], which defines in a more detailed way the SDN layers and architecture. The Fig. 7 depicts the architectural framework defined by the IRTF.

The SDN architecture gives the applications information about the state of the entire network from the controller, as opposed to traditional networks where the network is application aware. The architecture is: directly programmable because of forwarding functions decoupling, agile due to the abstraction that lets networks administrators dynamically adjust network-wide traffic flow to meet changing needs, centrally managed given that the controllers maintain a global view of the network, and open standards-based and vendor-neutral, because instructions are provided by SDN controllers instead of multiple, vendor-specific devices and protocols. A description of the planes, the interfaces and the abstraction layers that make up the architecture is presented below.

- **Application Plane:** This plane depicts the applications and network services. SDN Applications are programs that communicate behaviors and needed resources to the SDN Controller via the use of APIs. The applications can build an abstracted view of the network and they interact with the controller collecting information and requesting certain requirements for decision-making purposes. Network applications can be very varied and they could include networking management, analytics, or business applications.
- **Network Services Abstraction Layer:** This layer provides access to services of the control and management planes to other services and applications.
- **Northbound Interface:** Also called as service interfaces are a set of APIs available to network administrators (application developers), for example: CLI (Command Line Interface), GUI (Graphical User Interface) REST API and JAVA API, which offer the ability to pragmatically shape traffic and launch services, these APIs can be used to facilitate innovation and enable efficient orchestration and automation of the network. The northbound interface is used to communicate between the SDN Controller and the services and applications running over the network, this interface abstracts the low-level instructions to program forwarding devices needed to develop the network applications.
- **Control Plane:** The control plane is composed of the controller, which acts as the brain of the network and provides an entire view of the network, allowing network administrators to instruct network devices (switches and routers) how the network traffic should be forwarded. The controller is a logical entity that retransmits the instructions and requirements received from the applications to the underlying network. The information that is communicated to the applications includes statistics and events about what is happening within the network.
  - *Control Abstraction Layer:* Abstracts the control plane southbound interface and DAL from the applications and services of the control plane.
- **Management Plane:** The management plane is in charge of monitoring, configuration and maintenance of network devices. This plane interacts mostly with the operational plane than with the forwarding plane.
  - *Management Abstraction Layer:* Abstracts the management plane southbound interface and DAL from the applications and services of the management plane.
- **Southbound Interface:** The southbound interface allows the controller to communicate with the connectivity devices (data plane), this interface facilitates efficient control over the network and it enables the SDN controller to dynamically make changes according to real-time requirements, demands and needs. The southbound APIs can be open or proprietary, and the most well-known open-source southbound interface is OpenFlow, which is promoted by the ONF and which has been heavily adopted by the IT industry.
- **Device and Resource Abstraction Layer (DAL):** Abstracts the resources of forwarding and operational planes of the device to the control and management planes. The services that the rest of the planes provide depend on this abstraction layer.
- **Forwarding Plane:** Also known as infrastructure layer or data plane, is composed by connectivity devices, physical or virtual. This plane is responsible for handling data packets according the instructions provided by the controller. Some actions that are performed in this plane are: forwarding and dropping packets.
- **Operational Plane:** This plane manages the operational state of the network device, *i.e.* the activity or inactivity, the status and number of ports, the amount or processor

and memory among others parameters.

IV. SDN PERSPECTIVES

There is a large number of fields and applications where SDN technology can be developed and applied. Some prominent areas and topics are:

1) *Open-source solutions and ecosystems*: Since its conception SDN has promoted the development and the use of open-source interfaces and ecosystems. At the moment, there is a large number of projects that are active, a classification of them is provided in Appendix A.
2) *Network Functions Virtualization (NFV)*: SDN may provide a dynamic network behavior and ensure the delivery and quality of the network traffic between NFV's virtualized functions [13].
3) *5G networks*: SDN is considered as a key enabler for the deployment of 5G networks and working together with NFV, these technologies can offer a dynamic network behavior, agility and flexibility in the deployment of network services and applications [14].
4) *Transport networks*: Used in transport networks, SDN offers full fledged virtual networks (VNs) to the customers. An example is the first SDN WAN implementation deployed by Google in 2013 [15].
5) *Internet of Things (IoT)*: SDN is considered as a promising architecture to meet the requirements of scalability and heterogeneity of IoT devices and services [16].
6) *Smart grids*: SDN provides to the power grids a better network monitoring and improved network management, as well as programmability and the ability to evolve over time [17].

V. CONCLUSIONS

In this paper are presented the concepts, a summary of the chronological evolution over time and the terminology about SDN. The difference with the traditional networks, the benefits, the advantages and some perspectives of the use of this technology are described, as well as, the architecture with its planes, interfaces and abstraction layers as defined by the IRTF in the RFC7426. Further, a classification of SDN open-source projects is also provided. This document describes the SDN technology and its architecture under a structured and modular approach, aims to clarify the main ideas and fundamentals about this networking paradigm, and is intended to become a useful guideline of the more important aspects about SDN, which can be used to understand, analyze, design and deploy future systems and applications.

APPENDIX A
SDN OPEN-SOURCE PROJECTS

In this section is provided a classification of more than 130 SDN open-source projects. In this classification is presented information about: the leader, institute or company, the provision of an API, the status, a brief description of the project and the official website. The list is available at: https://goo.gl/Eqj8JX

REFERENCES

[1] F. Heart, A. McKenzie, J. McQuillian, and D. Walden, "Arpanet completion report," *BBN Report. Bolt, Beranek and Newman Inc.(BBN). Also published in an edited version as BBN Report*, vol. 4799, pp. 58–63, 1978.

[2] V. N. Index, C. Vni, C. Vni, and V. N. I. F. Highlights, "Cisco Visual Networking Index: Forecast and Methodology, 2016–2021," pp. 2016–2021, 2017.

[3] N. Feamster, J. Rexford, and E. Zegura, "The Road to SDN : An Intellectual History of Programmable Networks," *ACM SIGCOMM Computer Communication Review*, vol. 44, pp. 87–98, 2014.

[4] D. Kreutz and F. Ramos, "Software-Defined Networking: A Comprehensive Survey," *arXiv preprint arXiv: ...*, p. 49, 2014. [Online]. Available: http://arxiv.org/abs/1406.0440

[5] A. S. Tanenbaum *et al.*, "Computer networks, 4-th edition," *ed: Prentice Hall*, 2003.

[6] Google, "Google Trends - Search interest about SDN," 2017. [Online]. Available: https://trends.google.es/trends/explore?date=all&q=sdn

[7] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, p. 69, 2008. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1355734.1355746

[8] Algaba Enrique – Telefónica I+D, "Telecom Service Evolution with Network Virtualization," 2015.

[9] S. Shenker, "The future of networking, and the past of protocols," 2011. [Online]. Available: https://www.youtube.com/watch?v=YHeyuD89n1Y

[10] Z. K. Khattak, M. Awais, and A. Iqbal, "Performance evaluation of OpenDaylight SDN controller," *Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS*, vol. 2015-April, pp. 671–676, 2014.

[11] Open Networking Foundation, "Software-Defined Networking : The New Norm for Networks," 2012. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[12] I. R. T. F. (IRTF), "Software-Defined Networking (SDN): Layers and Architecture Terminology," Internet Research Task Force, Tech. Rep., 2015. [Online]. Available: https://tools.ietf.org/html/rfc7426

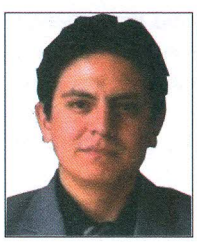
[13] Z. Michael, D. Allan, Shirazipour, M. Cohn, N. Damouny, C. Kolias, J. Maguire, S. Manning, D. McDysan, E. Roch, and M. Shirazipour, "OpenFlow-enabled SDN and Network Functions Virtualization," *Open Network Foundation*, pp. 1–12, 2014.

[14] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider, "Nfv and sdn-key technology enablers for 5g networks," *IEEE Journal on Selected Areas in Communications*, 2017.

[15] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu *et al.*, "B4: Experience with a globally-deployed software defined wan," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 3–14, 2013.

[16] N. Bizanis and F. A. Kuipers, "Sdn and virtualization solutions for the internet of things: A survey," *IEEE Access*, vol. 4, pp. 5591–5606, 2016.

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



Christian Tipantuña received his bachelor degree in Telecommunications Engineering from Escuela Politécnica Nacional, Ecuador, in 2011, and his MSc degree in Wireless Systems and Related Technologies from Politecnico di Torino, Turin Italy, in 2013. He is currently pursuing the Ph.D. degree in Network Engineering at Universitat Politècnica de Catalunya, Barcelona, Spain. His research interests include network functions virtualization, green networking, software defined networking and fog computing.

Carla Parra born in Quito - Ecuador. She completed her high school education at "María Angélica Idrobo", where she was the standard-bearer of the city flag. She completed his higher education at Escuela Politécnica Nacional, obtaining her bachelor degree in Electronics and Information Networks in 2018. Her interest areas include wireless communications, SDR (Software Defined Radio) and cellular systems.

Jorge Carvajal born in Quito - Ecuador on 02 August 1984. He studied at Escuela Politécnica Nacional, earning a bachelor degree in Electronics and Telecommunications Engineering. Then, he got his MSc degree in Information Technology at the University of Applied Sciences Mannheim, Mannheim Germany. He currently works at Escuela Politécnica Nacional as an assistant professor. His research interests include wireless communication systems and digital signal processing.

Ricardo LLugsi received his bachelor degree in Telecommunications Engineering from Escuela Politécnica Nacional in 2008. He earned his MSc degree in Communication Engineering from The University of Manchester in 2013, and a MEng in Connectivity and Telecommunications Networks from Escuela Politécnica Nacional in 2017. He has worked as a professional in governmental technical institutions, as well as in higher education institutions. He is currently a full-time professor at the Faculty of Electrical and Electronic Engineering at Escuela Politécnica Nacional. His areas of interest include design of antennas, radars and signal and image processing.