Journal of Electrical Engineering and Information Technologies, Vol. **5,** No. 1, pp. 47–69 (2020)
Article 171 ISSN 2545–4250
Received: April 30, 2020 e: ISSN 2545–4269
Accepted: May 16, 2020 UDC: 621.39:004.77.05

Original scientific paper

VIRTUALIZED PLATFORMS FOR NETWORK RESOURCES ORCHESTRATION AND MANAGEMENT

Borče Jovanovski, Valentin Raković, Liljana Gavrilovska

Faculty of Electrical Engineering and Information Technologies,
"Ss. Cyril and Methodius" University in Skopje,
P.O. box 574, 1001 Skopje, Republic of North Macedonia
jovanovskiborce4@gmail.com

A b s t r a c t: The concept of virtualization has contributed to improved structural functionality especially in modern telecommunications networks. The introduction of Software-Defined Networks (SDN) and Network Functions Virtualization (NFV), especially in mobile and wireless networks, marks the era of softwarization, centralized control and programmability of network entities. Various software platforms that support the concept of virtualization, provide a range of auspicious features such as modularity, multiplicity, interoperability, reliability. All of these features represent the building blocks of current fifth generation of mobile networks (5G). This paper focuses on the wireless (radio) virtualization. The paper gives the short overview of the latest and most important concepts related to wireless system virtualization and presents a platform operating a fully virtualized LTE network, capable of dynamically reconfiguring its radio parameters. The performance analysis focuses on the deployment and reconfiguration of the presented system. The results clearly show that wireless virtualization fosters significantly improved deployment and reconfiguration times, compared to conventional systems.

Key words: virtualization**;** software-defined networking (SDN); network function virtualization (NFV); long-term evolution (LTE); 5G network

ВИРТУЕЛИЗИРАНИ ПЛАТФОРМИ ЗА ОРКЕСТРИРАЊЕ И МЕНАЏИРАЊЕ НА МРЕЖНИТЕ РЕСУРСИ

А п с т р а к т: Концептот на виртуелизација придонесе за подобрена структурна функционалност, особено во современите телекомуникациски мрежи. Воведувањето на софтверско дефинираните мрежи и виртуелизацијата на мрежните функции, особено во мобилните и безжичните мрежи, ја означува ерата на софтверизација, централизирана контрола и програмабилност на мрежните ентитети. Различни софтверски платформи кои го поддржуваат концептот на виртуелизација, обезбедуваат голем број ветувачки карактеристики како што се модуларност, слоевитост, интероперабилност, доверливост. Сите овие карактеристики претставуваат градбени блокови на актуелната петта генерација на мобилни мрежи (5G). Овој труд се фокусира на виртуелизацијата во безжичен домен (радио-пристапен дел). Даден е кус осврт на најновите и најважни концепти поврзани со виртуаелизацијата на безжичните системи, а исто така е претставена платформа која работи во целосно виртуелизирана LTE мрежа, способна за динамично конфигурирање на нејзините радио-параметри. Анализата на перформансите се фокусира на инсталирање и реконфигурација на презентираниот систем. Резултатите јасно покажуваат дека виртуелизацијата во безжичен домен овозможува значително подобрување на времето на инсталирање и реконфигурација, споредено со конвенционалните системи.

Клучни зборови: виртуелизација; софтверско дефинирани мрежи; виртуелизација на мрежните функции; четврта генерација на мобилни мрежи (4Г/LTE); петта генерација на мобилни мрежи (5Г)

1. INTRODUCTION

The last decade networking was mostly focusing on the developments of wireless technologies (wireless and mobile networks) aiming to successfully follow up the achievements in the fixed networks. The increased Internet traffic and the enormous growth of the number of users boost these

developments, making the conventional way of managing networking and resources allocation in current networks inappropriate. Therefore, it was necessary to develop a concept that would respond appropriately to these problems. One solution was to introduce the virtualization techniques [1], [2], [3] in the networking paradigm.

The emerging demanding networking environment has introduced a several novel concepts such as Software-Defined Networking (SDN) [4], [5], and Network Function Virtualization (NFV) [6], [7], that introduce, a separation of control from data plane, enabling the control plane to be implemented as software running on a standard server.

Virtualization and software-based control fosters swift and dynamic reallocation of the network resources. Furthermore, centralizing the pool of resources, i.e. cloud-mediated environments (Cloud computing) [8], [9], enables even more efficient resource management and orchestration.

The next generation of mobile networks (5G) is based on the aforementioned concepts (their expansion and upgrading), and provides support for more radio access technologies (cellular, Wi-Fi and fixed), more services (broadband Internet), massive growth of machine type of traffic and time critical machine type of traffic, variety of networks and service operators [10]. An integral part of these complex network`s architecture is the use of Virtual Machines (VMs) [11] and Containers [12] that provide, improvements in communication, security, integrity, and increased speed of information transmission, while reducing the costs of network maintenance.

The paper provides an overview of the current State-of-Art in wireless network virtualization, specifically focusing on the most prominent solutions, Open Source MANO and Open Network Automation Platform, for NVF and network slicing. Moreover, the paper elaborates the practical implementation (demo platform) of a fully virtualized Long Term Evolution (LTE) mobile system based on the core concepts of Open Source MANO. The developed demo platform is capable of dynamic and on-the-fly reconfiguration, providing some possible solutions towards the future wireless networking. A novel algorithm developed as a script in the JavaScript programming language, and working in the open source environment Nodejs represents the reconfiguration capabilities. The focus of this script is to reconfigure the base station parameters in real time, by polling real-time data regarding the system performances and behavior. The performance analysis and the presented results demonstrate the proper operation of the developed reconfiguration script and justify the benefits of container-based virtualization.

The paper is organized as follows: Section 2 presents a theoretical explanation of the concept of virtualization as a major transition driver towards the fifth generation of mobile networks. Section 3 highlights the characteristics and benefits of SDN and NFV implementation. Section 4 focuses on containerization, comparing virtual machines with containers, as well as understanding their advantages and disadvantages, Section 5 presents the practical implementation and appropriate performance analysis and Section 6 concludes the paper.

2. VIRTUALIZATION IN WIRELESS NETWORKS

Virtualization as a core component of the transition between the conventional networks and the new generation of mobile networks can be defined as a novel paradigm where hardware and software are being decoupled. The purpose of implementing this concept comes from the inability of traditional architecture to serve multiple applications that use separate software at the same time, resulting in increased operating and capital costs. Therefore, the introduction of the virtualization layer into the network architecture allows operation in a multi-tenant fashion where a number of different applications can operate concurrently (Figure 1).

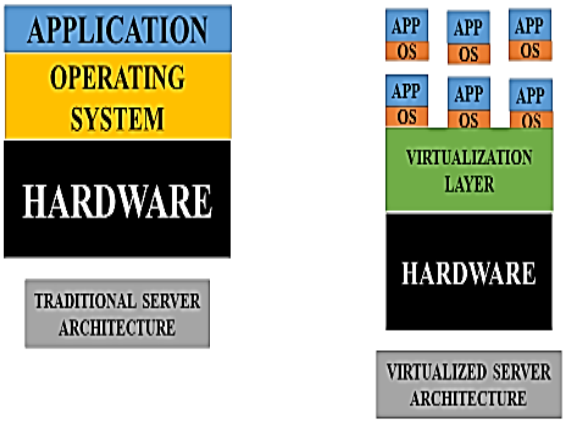

Fig.. 1. Traditional architecture vs. virtual architecture [1]

This provides a range of benefits: efficient use of server resources, high availability, power saving and installation space, less cooling power, simple

management, fast and easy commissioning, scalability, compatibility with a range of hardware platforms, automation and simplification upgrades.

Virtualization has three features that make it recognizable:

- **Partitioning** – a large number of applications and operating systems are supported over the virtual layer that facilitates the uniform sharing of available resources.
- **Isolation** – each operating system together with the appropriate running application represents a separate virtual machine that runs on its host machine isolated from the others. If during the event one virtual instance fails, it does not affect the work of the others.
- **Encapsulation** – enables the virtual machine to be stored as a single file that can be easily identified later, in terms of the services it provides.

The implementation of the network virtualization has started with wired networks. The idea continues towards the virtualization of wireless networks. Even more complex, these developments are based on potential to improve the spectrum utilization and to create new services. The *wireless network virtualization* [13] is following the exponential growth of number of users and aims to be designed to support billions of users' devices and to include new wireless services for cellular systems and the Internet of Things (IoT). This type of virtualization combines different wireless networks with different access technologies and network-based (infrastructure-based and infrastructure-less) technologies that facilitate efficient convergence, sharing and abstraction. Currently, there is no unified and universal architecture for wireless network virtualization available for commercial use (such as, Universal Virtualization, Cross-infrastructure Virtualization, Limited Intra-infrastructure Virtualization).

Moreover, wireless virtualization can be performed using cross-platform and intra-infrastructure applications, spectrum levels and (cloud) computing levels. Thus, wireless network virtualization has the potential to alleviate the problem of artificial spectrum depletion and offer new services to support huge wireless domain subscribers, for future generations of wireless systems. The following section will describe the main features of the focal technologies related to network virtualization, SDN and NFV.

3. SDN AND NFV IN A SHELL

This chapter shortly highlights the main SDN and NFV features. Even being distinct technologies they complement each other in fostering flexibility in network deployment and operation.

3.1. *Software defined networking*

The software defined networking (SDN) [14] offers a new approach to network programming, control, transformation and management of heterogeneous dynamics that emerges through appropriate *open source interfaces*. The general idea is based on installation of a more cost-effective controller that manages the network, despite the rigid configuration of any devices and interfaces at a lower level. This concept offers separation of *control* from *data* plane. The organization dedicated to promotion, development and standardization of SDN is the Open Network Foundation [15]. The providers of software platforms for implementation are: Cisco, Juniper, Open Source, VMware, Radware, Tail-f, OpenFlow, onePK, LISP, OTV BGP flowspec, OpenStack, Contrail, Yang.

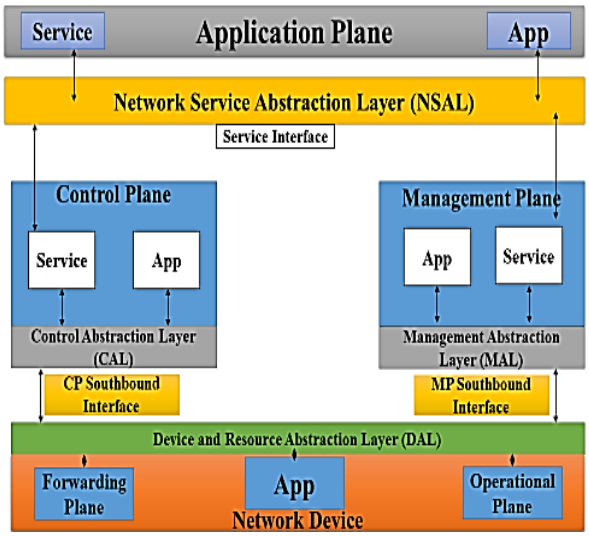

Fig.. 2. SDN network architecture [4]

Figure 2 presents a high level SDN network architecture. The overall design includes solutions for building, architecture designing, managing multi-user cloud-based services, structuring and operating a data center with virtualization applications and different vendor-specific components, including high availability solutions and virtualized backups, integrating data center with MPLS network service etc.

This SDN structure provides emerging *micro-services* and *macro-services* existing in the control and the management sections. Compared to conventional virtual machines, the macro-services are more efficient, can increase the agility in software development, introduce automated process deployment, resilience and scalability. The micro-services are modular and can be used to build and manage the SDN and NFV services. Compared to virtual machines, the micro-services can increase the latency that disrupts the development of interconnected software applications. The main disadvantages of the micro-services can be addressed with the deployment of full stack virtual machines. In case of virtual machines, the interfaces for the network components interoperability need to be well configured and programmable. They can be proposed by some particular protocol – using some local processor communication (LPC), a remote protocol, an open standard defined protocol, etc.

This concept is beneficiary regarding: drastic reduction of CAPEX and OPEX to the equipment providers themselves, simplification of network configuration, behavior optimization of the network capacity, easy upgrades, etc.

3.2. *Network function virtualization*

The Network Function Virtualization (NFV) separation of the hardware components from the software, allowing them to be upgraded as separate entities in order to overcome the problem of interoperability [16]. The NFV represents a virtualization type where the network functionalities are independent of the hardware where they operate. The NFV can address a number of the emerging requirements, also important for 5G development and implementation.

The NFV enables separation of the software from the hardware and allows independent evolution of both entities strictly separating their upgrades and maintenance. In addition it provides flexibility in sharing of network resources and enables usage of the same physical infrastructure for implementation of different services. The NFV also fosters the dynamic performance scaling depending on the conditions (e.g. current traffic load) to achieve the appropriate capacity.

The NFV ensures flexible resource sharing using the same physical architecture for implementation of different services. Figure 3 depicts an NFV example for *Customer Premise Equipment – CPE*, and its virtual counterpart *vCPE* (*virtual Customer Premise Equipment*), with possible NFV implication for redefining the network access port.

This particular organization of network devices and appropriate resource allocation allows reducing the cost of network equipment and energy consumption (the possibility of reusing standard equipment for multiple purposes), reducing the time to appear on the market (Time to Market) through the minimization of the innovation cycle. It also allows: the possibility of targeted deployment of services based on geographical location or type of user; the opportunity to open the market to the academic community and smaller companies; configuration optimization in almost real time (using the capability of standard servers and memory to perform consumption management).

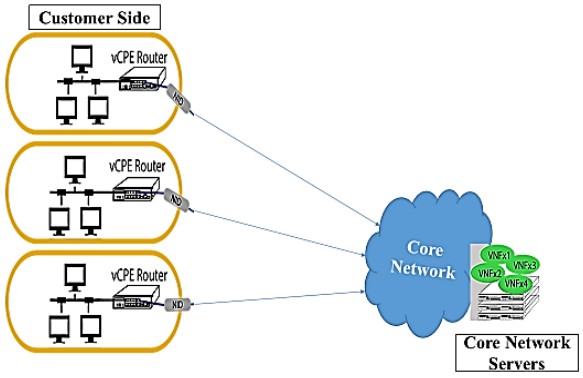

Fig. 3. Implementation of CPE in relation to vCPE [6]

The NFV realizations can be performed with almost any equipment offered by the manufacturers related to software implementations mentioned in the SDN section. The next sub-section describes the complete structure of NFV and such implementation. There exist many solutions that utilize the NFV paradigm. The most notable ones are the Open Source MANO (OSM) and Open Network Automation Platform (ONAP). Moreover, the presented virtualized LTE system builds upon the generic architecture of OSM and its interfaces. Specifically, the communication between the virtualized LTE system and script.sh is based on the generic definitions of the Os-Ma interface. The remainder of this section elaborates these both solutions.

3.2.1. *Open source MANO implementation*

The open source MANO platform [17], [18] facilitates the drivers for network architecture presentation and configuration, drivers of how information is transmitted, and ways of communicating

between the entities on wireless and mobile networks. OSM fosters swift and efficient approach for commercial NFV deployment, and enables possibilities for novel and modular network upgrades. This platform is an ETSI (European Telecommunication Standards Institute) project aiming to introduce standardization approach and define the specifications and requirements for the NFV technologies.

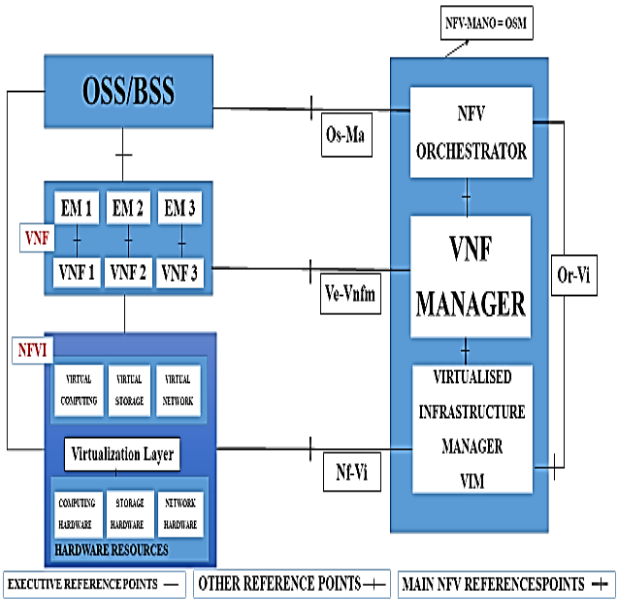

Fig. 4. Upgraded NFV architecture – OSM [18]

Figure 4 illustrates the design of the OSM architecture with components responsible for communication and resource allocation. The lower block represents the infrastructure platform *NFVI (Network Functions Virtualization Infrastructure)* where the hardware resources are passing over the virtualization software layer and have all the features of virtualized functions. The second block (the middle layer) called *VNF (Virtualized Network Function)* allows these network functionalities to be deployed according the appropriate requirements. The block called *NFV Management and Orchestration* is the most important in the MANO platform. It includes the overall logic of the platform, with extremely reliable functions used to manage and monitor components, recover the system, provide effective security and serve all requirements.

The simplification of the performance and functions within the structures themselves is based on the multi-layered approach, where everyone knows their function and performs the assigned task. The MANO architecture enables abstraction, monitoring, modularity and simplicity. It consists of three modules: *the NFV Orchestrator*, *the VNF Manager*, and *the VIM (Virtualized Infrastructure Manager)*.

Mutual communication among the structure`s entities is established through software-configured interfaces that take care of traffic flow, resource sharing, and are system drivers, e.g. in the event of an emergency (due to congestion or request for more resources to perform certain functions). If the system requires more resources than specified in the specification, additional interfaces can be opened up to fill the gap instead of adding resources to the existing interface. This is because the interface that allocates the appropriate resources (connecting the orchestrator to the OSS/BSS system) is parsimonious and basically does not allow the new allocation of more resources than specified in the specification. Also during the emergency situations opening a new interface is allowed. To avoid cluttering the architecture of interfaces or congestion, communication through the interfaces to and from the VNF Manager can be represented by a proxy server and all communication is carried out with containers and virtual machines.

The overall idea of this concept and the introduction of additional functionalities went through several stages known as Release sections. There are total of six finalized OSM Releases, and one draft version (OSM Release SEVEN). The latest versions of OSM [19], enables novel concepts such as end-to-end network slicing [20] and provides possibilities for integration into the fifth generation of mobile networks (5G) [21]. The latest Releases and the concept of Network Slicing are shortly introduced in the following subsections.

3.2.1a. *OSM Release SIX and SEVEN*

OSM Release SIX [22] is characterized by facilitating the management of more complex services, aiming to expand the possibilities for creating more network service`s operations. It provides the support for edge platform enabling end-to-end active service and slice orchestration from edge to core. This release also covers the expansion of the Service Assurance (SA) framework, which enables control, storage and response to a much larger set of events and conditions.

The OSM Release SIX expands the range of technologies already supported by the OSM (new connectors have been developed for FOG05 edge clouds, VMware's vCloud Director 9.5, public clouds, etc.). Accordingly, there is an upgrade of the

set of connectors already used in earlier releases designed to support the EPA (Enhanced Platform Awareness) and the additional attributes (addition of multi-segment networks).

All these enhancements lead to a more flexible operator's experience, with an improved control over orchestration roles (fine-grained control of operations per role and project), and better real-time feedback to the operator, along with various improvements to ease VNF on boarding and testing phases.

OSM Release SEVEN [22] is characterized by bringing cloud-native applications for NFV deployments, enabling OSM to manage more than 20,000 Kubernetes applications already existing, with no need for any translation or repackaging.

This Release extends the OSM SDN framework to support next-generation SDN solutions by providing higher-level primitives, increasing the number of options available to support I/O-intensive applications. The SDN plug-in and inter-hub add-on models have been consolidated, and the management, addition and maintenance of the SDN jacks have been greatly simplified. Release SEVEN brings great enhancements designed to enhance the overall user experience and interoperability choices. It includes the improved VNF configuration workflow, which enables much faster and more complex operations, that support additional types of infrastructure, such as Azure and VMware's vCD 10, complementing previously available options (VIMs -based-OpenStack-based VIMs, VMware VIO, VMware vCD, AWS, Fog05 and OpenVIM).

Several existing OSM versions, include OPEN BATON (Fraunhofer FOKUS), ONAP (The Linux Foundation) and, OpenStack Tacker. However, the OSM is still on top as a compatible offering due to limited activities, poorly structured components, or appearances of holes in the architecture. It does not mean that it cannot be overcome by some new appearing concepts. There are various MANO implementations of real solutions in terms of its architecture. Some of them include: POSENS [23], 5GTANGO [24], T-MANO (SK-Telecom Korea implementation) [25].

3.2.1b. *Network slicing*

The Network Slicing is one of the most important contributions of the OSM Release FIVE [26]. Its main purpose is to provide *end-to-end functionality covering all network segments*. It enables the simultaneous deployment of a large number of resources in terms of logical, autonomous and non-divisible components so that they can function on a common infrastructure platform. The management of the resource allocation is based on the flexibility and the requirements of the network services (Figure 5). Additionally, the OSM Release FIVE supports Artificial Intelligence (AI) and enables the network to subscribe to different services at the same time without interfering with each other [27].

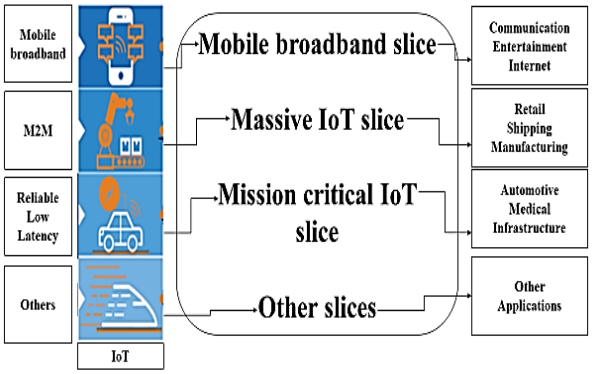

Fig. 5. 5G network slicing [27]

Some of the transient issues related to Network Slicing, and even to business justifications of Network Slicing, may disappear if the final specifications carefully follow the principles of cloud environments. As a result, the advantages and disadvantages of Network Slicing cannot be justified during its current development stadium.

3.2.2. *ONAP platform architecture*

The ONAP platform facilitates network virtualization [28], [29] and provides manufacturers with independent capabilities to design, create and manage the life cycle of network services. It also provides a unified vendor-specific operating framework, policy-driven service design, management, analytics and life cycle management for large-scale workloads and services. With ONAP, network operators can synchronously orchestrate the physical and virtual network`s functions. This approach enables operators to take advantage of existing network investments and solutions, while at the same time, the openness and ubiquitous acceptance of ONAP by major network providers around the world accelerates the development of the VNF ecosystem.

The ONAP community also defines solutions for key use cases, such as 5G, BBS, CCVPN, Voice over LTE (VoLTE) and vCPE, which the user community expects to download immediately. Testing these solutions with a variety of open source com-

mercial network elements during the development process, enables ONAP platform developers to have real-time feedback on in-progress code, and guarantees a trusted framework that can be quickly adopted by other users like the final release.

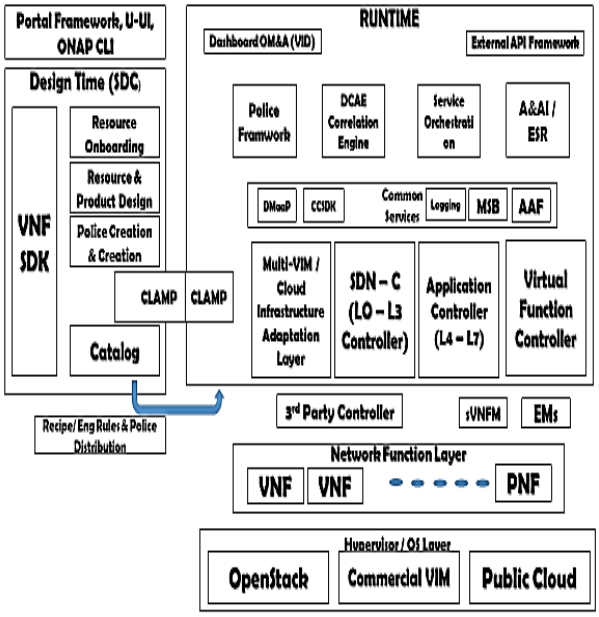

Fig. 6. ONAP platform architecture (Dublin Release) [29]

Figure 6 presents a high-level view of the ONAP architecture with its micro-services-based platform components. As a cloud-native application that consists of numerous services, ONAP requires sophisticated initial deployment as well as post-deployment management. The ONAP Operations Manager (OOM) is responsible for orchestrating the end-to-end life cycle management and monitoring of ONAP components. It is integrated with the Microservices Bus, which provides service registration/discovery and support for internal and external APIs and key SDKs. OOM uses Kubernetes to provide CPU efficiency and platform deployment. In addition, OOM helps enhance ONAP platform maturity by providing scalability and resiliency enhancements to the components it manages.

The platform provides tools for service designers as well as a model-driven run-time environment, with monitoring and analytics to support closed-loop automation and ongoing service optimization. Both design-time and run-time environments are accessed through the Portal Framework, with role-based access for service designers and operations personnel.

The run-time environment executes the rules and policies distributed by the design and creation environment, as well as the Controllers that manage physical and virtual networks. The Active & Available Inventory (A&AI) component provides real-time views of a system's resources, services, products and their relationships with each other. In a fast-moving environment with rapid deployment and teardown of virtual resources, this real-time monitoring and mapping is critical to service assurance. The run-time service execution components are in constant communication with the closed-loop automation modules, which provide real-time monitoring, analytics, alarm and event correlation, etc. These modules, in turn, provide up-to-the-moment intelligence about existing services to service designers, who may see opportunities to fine tune them, or replicate high-performing portions of deployed services in creating new ones.

The design-time framework provides a comprehensive development environment with tools, techniques, and repositories for defining and describing resources, services and products. This includes policy design and implementation, as well as an SDK with tools for VNF supplier packaging and validation.

The following section will present the main facilitators of NFV, with respect to the virtualization process.

4. VMS AND DOCKER CONTAINERS IN MODERN TECHNOLOGIES

The novelties and innovations of today's applications and their development are represented by container-based virtualization and virtual machines, which is one of the most used in Cloud Computing technology. This concept is driving the development of newer technologies that is gaining momentum. *Containers*, also known as operating system virtualization, are used to launch and run the distributed applications. They can be considered as super minimalist virtual machines that do not operate through a hypervisor. The containers themselves are presented as portable interfaces for applications uploading, along with all libraries, and configuration files. They allow applications to run safely in different environments with operating system and physical infrastructure abstraction. Containerized applications share the host OS kernel with other containers and the readable part of the operating system. Inside the containers we have a task that needs to be performed and the appearance of microservices. Depending on the situation on the network, the containers can be deleted, modified or, upgraded.

Popular container management and management software are: Kubernetes, Docker Swarm, Amazon ECS, Azure Container Service, Marathon, CoreOS Fleet, Open Stack Magnum, Diego, Hashicorp Nomad, Mesos. Opposite containers are *virtual machines* that represent a simulation of a computer system, i.e. a larger number of hardware components represented by a single software entity. Operating systems and their applications share hardware resources on a single server called host or server-based or multiple hosts. Each virtual machine requires its own basic operating system while the hardware is virtualized. Crucial to virtual machines is the presence of the *hypervisor* (a key difference in the structure between containers and virtual machines, Figure 7), or known as a *virtual machine monitor* that represents software, i.e. the firmware for creation and functionality of virtual machines. The virtual machines enable all operating system resources to run the applications and, to have available all established management tools. It also provides availability of fixed security tools, and enables better security controls. Popular VM providers are: VMware vSphere, VirtualBox, Xen, Hyper-V, KVM.

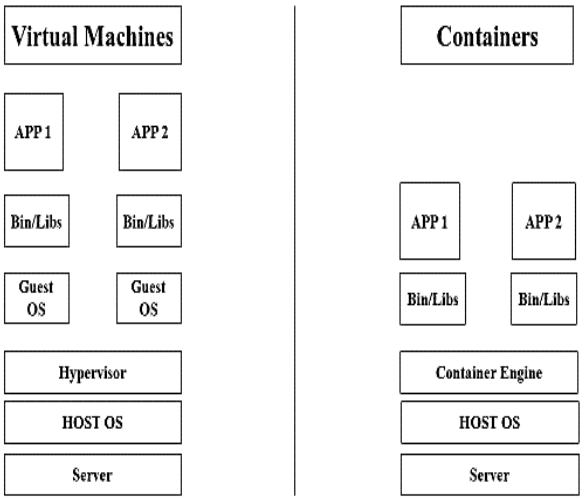

Fig. 7. Virtual machine structure vs. container structure

Although they share common characteristics in terms of sharing physical resources, there are still differences between them. Initially this is the appearance of the hypervisor in the virtual machine`s architecture, that is not present in the containers. Moreover, the smaller overhead, faster operation and faster initial lift are features of the containers while the virtual machines are far better with reliability mechanisms. The virtual machines have the isolation feature that enable operating system to be separate for each application and, to each virtual machine and is not shared with other virtual machines. Isolation represents the concept that allows all VMs to continue to operate, when one of them experiences a system outage or security issue.

Communication with containers is also used in medicine when setting up appropriate software that will communicate with the database and gather the necessary information for appropriate medical treatment. All this is done in order not to interfere with this data and to properly pronounce the necessary diagnosis and document it in the database.

The following subsections elaborates on some of the most popular container use and management software.

4.1. *Docker containerization*

The Docker [30], is one of the three most popular container (usage and management) software besides *Kubernetes* and *Mesos*. There are several noteworthy terms when it comes to this type of technology. First and foremost is the notion of container image (based on the function of this container software), which allows containers to be used on any host, usually with a Linux kernel. Although containers have been known for several years, they have become popular with the introduction of Docker. This approach enables much faster implementation and deployment of applications compared to access using VMs. Container image is a snapshot of a file system on a disk that can be saved to disk. The container file system is organized in levels so that each container image can indicate from which parent image it originates. Because of this tiered organization, container images are small. Every desktop image allows software tools, such as Docker, to simplify the design and management process. So, using Docker the whole image container can be raised with a single command. One Docker container has everything necessary for the application to be active. Docker containers are created from the Docker image and they can be dropped, shut down, stopped, moved or deleted. Each Docker container (runtime instance) has an operating system, user files and meta-data, and can be run on a physical server, a virtual machine, a data center, in the cloud (private or public), on a laptop, or on a desktop computer (Figure 8).

Other additional tools for Docker containers are:

- **Docker machine –** enables to change the number of Docker hosts;

- **Docker compose –** facilitates the development of complex and distributed applications;
- **Docker swarm –** provides clustering.

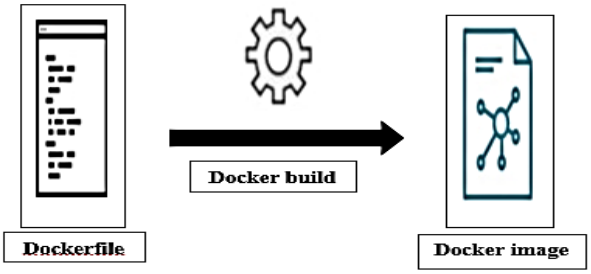

Fig. 8: Docker image – the way of creating

4.2. *Docker VS OpenStack*

Based on the previously available open source technologies, the Docker has created a standard way to deploy applications in appropriate containers that will later allow them to work in different environments. It has several advanced open source projects that complement this platform. It allows expanding towards the companies such as the giant Microsoft and Google. Needless to mention, OpenStack (Cloud Based Operating System) [31], [32], [33], also supports containers that can run on virtual machines or bare-metal. There was a need to find a solution from OpenStack that would make a good combination of communication between containers and virtual machines (the solution fell on OpenStack in 2014 with the introduction of the Magnum projects [34], Murano [35] and Kolla [36]). Containers do not incorporate the same security features and mechanisms as virtual machines. In order to alleviate this issue, the combination of hypervisor-based virtualization (e.g. OpenStack) with containers (e.g. Docker) provides a hybrid solution that incorporates the flexibility and agility advantages of containers and the security and insulation benefits of hypervisor-based virtualization.

4.3. *Kubernetes*

Kubernetes [37] is an open source platform for deploying and managing containers. It provides a container runtime, container orchestration, container-centric infrastructure orchestration, self-healing mechanisms, service discovery and load balancing. It's used for the deployment, scaling, management, and composition of application containers across the clusters of hosts.

It aims to reduce the burden of orchestrating the underlying compute, network, and storage infrastructure, and enables application operators and developers to focus entirely on container-centric workflows for self-service operation. It allows developers to build custom workflows and higher-level automation to deploy and manage applications consisting of multiple containers. Kubernetes is a very flexible and extensible platform. It allows consuming its functionality, or use proprietary solution in lieu of the built-in functionality. It can be also integrated into specific environment with added additional capabilities.

A Kubernetes environment consists of a control plane (master), a distributed storage system for keeping the cluster state consistent, services, networking, pods and a number of cluster nodes (Kubelets) (Figure 9).

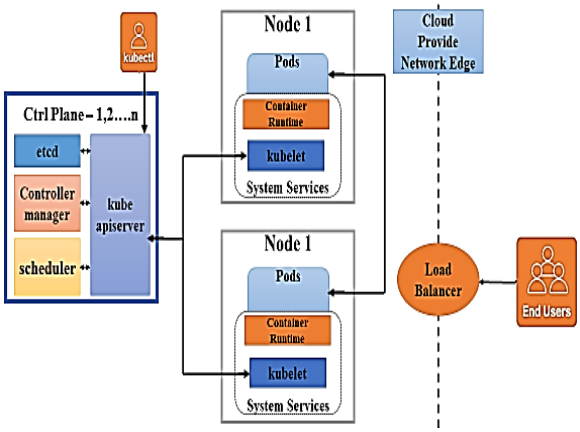

Fig. 9: Overview of Kubernetes architecture [37]

Kubernetes control plane. The control plane is the system that maintains a record of all Kubernetes objects. It continuously manages object states, responding to changes in the cluster; it also works to make the actual state of system objects match the desired state. As the above illustration shows, the control plane is made up of three major components: kube-apiserver, kube-controller-manager and kube-scheduler. These can all run on a single master node, or can be replicated across multiple master nodes for high availability. The API Server provides APIs to support life cycle orchestration (scaling, updates, and so on) for different types of applications. It also acts as the gateway to the cluster, so the API server must be accessible by clients from outside the cluster. Clients authenticate via the API Server, and also use it as a proxy/tunnel to nodes and pods (and services),

Cluster nodes. Represent machines that run containers and are managed by the master nodes. The Kubelet is the primary and the most important controller in Kubernetes. It's responsible for driving the container execution layer, typically Docker;

Pods and services. Pods are one of the crucial concepts in Kubernetes, as they are the key construct that developers interact with;

Kubernetes services. Services are the Kubernetes way of configuring a proxy to forward traffic to a set of pods. Instead of static IP address-based assignments, services use selectors (or labels) to define which pods uses which service. These dynamic assignments make releasing new versions or adding pods to a service really easy. Anytime a Pod with the same labels as a service is spun up, it's assigned to the service;

Kubernetes networking. Networking Kubernetes has a distinctive networking model for cluster-wide networking. In most cases, the Container Network Interface (CNI) uses a simple overlay network (like Flannel [38]) to obscure the underlying network from the pod by using traffic encapsulation. In both cases, pods communicate over a cluster-wide pod network, managed by a CNI provider.

5. PRACTICAL IMPLEMENTATION

The practical implementation consists of a demo platform with designed interfaces between two client-server entities in a fully virtualized LTE system and a proposed algorithm for reconfiguring base station parameters in real time. It allows fully flexible and programmable communication between the entities of the network, and can perform full adaptation to the requirements of the users. This section presents the full structure and detailed description of the demo platform and its functionalities. The reconfiguration of the base station's parameters is performed according the developed program code (denoted as **script.sh** in the remainder of the paper) introducing thresholds set to three selected parameters: the *number of max users*, the *bit rate from the base station to the user (downlink)* and *the bit rate from the user to the base station (uplink)*.

5.1. *The platform overview*

This subsection describes the technical specifics of the demo platform. The platform consists of a virtualized LTE system (3GPP Rel. 14) that can be used for practical experimentations related to V-RAN deployment and orchestration. It consists of four core logical entities, Remote Radio Head (USRP X310), compute node, LTE UEs and system controller (Figure 10). All components are implemented and designed on commercial based hardware.

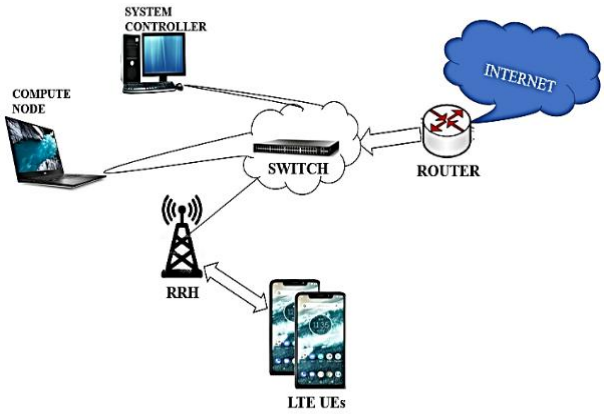

Fig. 10. Logical entities on demo platform

The major functionalities of these entities are:

The **system controller** represents the core control and management entity responsible for the orchestration and operation of the virtualized LTE system. It incorporates the **script.sh** functionalities and facilitates the reconfiguration process of the virtualized LTE **eNodeB**.

The **Remote Radio Head (RRH)** is implemented on a Software Defined Radios (SDR) platform utilizing the Universal Software Radio Peripheral (USRP) X310 devices, developed by National Instruments. They act as the radio hardware of the virtualized LTE system, i.e. the radio hardware of the virtualized **eNodeB**.

The **compute node** is responsible for the RAN-specific baseband processing. It runs the baseband processing in a container-based virtualization that utilizes the Docker framework. The specific experiment platform uses commercial LTE software, *Amarisoft*, which implements a commercial-grade full stack LTE Rel. 14. The Amarisoft can be executed on General-Purpose Processors (GPP) and is compatible with any RF front-end, such as the USRP devices.

The **LTE UEs** are represented by two Handheld handsets (Samsung Galaxy S9). The UEs are responsible for triggering the communication process with the virtualized LTE system. The UEs incorporate the Rhode & Schwartz *QualiPoc software* that is specifically designed for commercial grade performance monitoring of cellular systems.

5.2. *Performance analysis*

This subsection presents the performance analysis of the virtualized LTE system by exploiting the **script.sh**. The main goal of script.sh is to *dynamically* reconfigure the virtualized LTE system based on the underlying communication parameters, (e.g. channel utilization, aggregate system throughput, number of active devices). These types of reconfigurations can be especially important for plethora of scenarios such as flash crowds, emergency situations, energy efficiency, etc. Figure 11 depicts the generic message sequence chart (MSC) of the scriptsh.

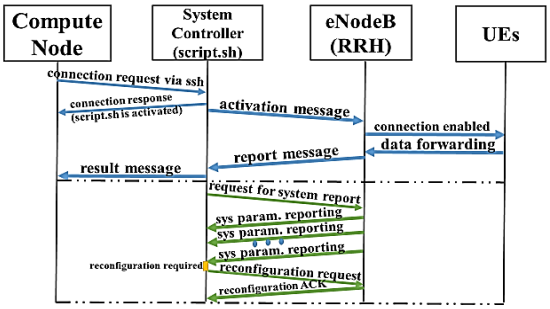

Fig. 11. MSC of script.sh

The performance analysis focuses on the eNodeB *channel bandwidth reconfiguration* capabilities, orchestrated by the scrip.sh. The presented results delineate specific Key Performance Indicators (KPIs) related to the virtualized system reconfiguration, i.e. channel bandwidth reconfiguration. Specifically, the KPIs of interest are the *required time for LTE system reconfiguration* and the UE *communication reestablishment period.* Table 1 presents the relevant system parameters for the performance analysis.

Figure 12 depicts the eNodeB reconfiguration and UE connection reestablishment process, for different LTE channel bandwidths. The presented results are gathered from the Rhode & Schwartz QualiPoc software and show case the system behavior during the reconfiguration process. The arrow in Figure 12, denotes the reconfiguration period of interest. It is evident that the reconfiguration time and UE connection reestablishment is in the order of seconds. This is significantly faster, compared to the conventional (non-virtualized) wireless system. The results in Figure 12 demonstrate that RAN virtualization can foster fast reconfiguration and deployment capabilities, which are of outmost importance for future wireless systems, such as 5G. For example, the 5GPPP requirements for fast creation of services specify that the average creation time should be less than 90 minutes [39]. This is significantly higher than the measured reconfiguration and the communication reestablishment periods, of the presented virtualized LTE system.

T a b l e 1

System parameters

System parameters	Parameter value
Channel bandwidth (MHz)	5, 10, 15, 20
No. of eNodeB	1
No. of UEs	2
Antenna mode	2×2 MIMO
LTE earfcn	3350
Modulation	Adaptive based on channel conditions
UE type	Samsung Galaxy S9

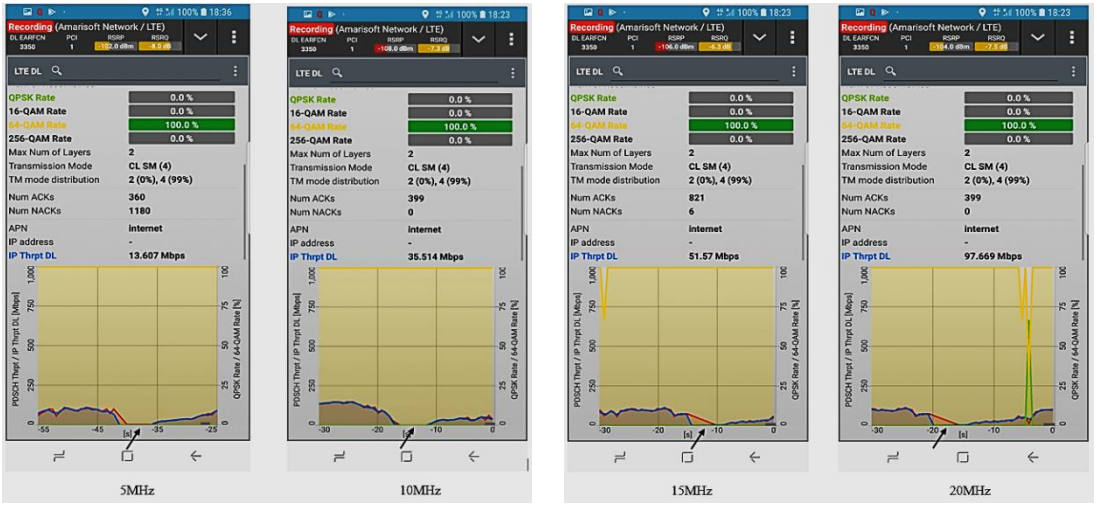

Fig. 12: eNodeB reconfiguration for different channel bandwidths

Table 2 presents the communication reestablishment period with respect to the system reconfiguration bandwidth.

T a b l e 2

Communication reestablishment period

Bandwidth (MHz)	5	10	15	20
Communication reestablishment period (s)	3.5	3.86	5.02	6.23

The results in the table show that the reconfiguration and the communication reestablishment process are proportional to the channel bandwidths. Higher channel bandwidths result in longer system deployment and communication reestablishment periods. This is a valuable insight that should be taken into consideration when performing system reconfigurations, especially for mission critical scenarios, where the system outage should be minimal.

6. CONCLUSION

Traditional networks have a big challenge to respond to huge customer demands, exponentially growing number of devices, and explosive requirements for broadband and network access everywhere. It raises a problem of network capacity. The increase in capacity initially has been solved with hardware extensions that make this process too slow and expensive.

The next generation of mobile networks is facing new key requirements such as: increasing data speeds, connecting a large number of devices to the network, efficient utilization of hardware resources, energy efficiency of systems, etc. It inevitably requires design of new network architectures that will offer a flexible and adaptable solution. The SDN and NFV are possible solutions which design is based on software-defined approach, utilization of virtualization technology and easy and dynamic adaptation to new requirements. The SDN separates the data from the control plane and introduces centralized control, thereby making the networks programmable. The NFV seeks to optimize and improve network services and features by virtualizing and moving their implementation from specialized, proprietary hardware to software running on standard servers. Together, they offer powerful tools for creating smarter, more agile and more efficient networks.

The paper provides an overview of the SoA solutions for wireless network virtualization, i.e. OSM and ONAP. Moreover, the paper presents the results of the designed demo platform (based on the OSM concept), which consists of a virtualized LTE system designed for deployment and orchestration of network resources. A developed program code was implemented to reconfigure the base station in real time according to pre-set thresholds. The performance analyses show that wireless virtualization *can foster fast and efficient network deployment and reconfiguration*. Moreover, the results show that there is a *correlation between the network deployment and communication reestablishment with the system channel bandwidth*. This aspect should be taken into consideration in situations where the system outage, due to reconfiguration, should be kept minimal.

Acknowledgement. This work was funded by the NATO SPS programme through the G5269 FALCON project (2017–2020). The authors would like to thank the Ass. Prof. Daniel Denkovski (*"Ss. Cyril and Methodius" University in Skopje, R.N. Macedonia*) for his cooperation and involvement in setting up the demo platform.

REFERENCES

[1] Online information: https://www.sdxcentral.com/sdn/network-virtualization/definitions/whats-network-irtualization/

[2] Online information: https://ostec.blog/en/general/virtualization-concepts-and-terminologies.

[3] Gavrilovska, L., Raković, V., Denkovski, D.: Aspects of resource scaling in 5G-MEC: Technologies and opportunities, *IEEE Globecom* (Dec. 2018).

[4] Online information: https://www.cisco.com/c/en/us/solutions/software-defin ed-networking/sdn-vs-nfv.html.

[5] Kreutz, D., Ramos, F. M. V., Verissimo, P., Rothenberg, C. E., Azodolmolky, S., Uhlig, S. , "Software-Defined Networking: A Comprehensive Survey", *Proceedings of the IEEE* ,Vol. 103, No. 1 (January, 2015).

[6] Han, B., Gopalakrishan, V., Ji, L., Lee, S.: Network Function Virtualization: Challenges and opportunites for innovations, *IEEE Communications Magazine* (February 2015).

[7] Mijumbi, R., Serrat, J., Gorricho, J., Bouten, N., Turck, F., Boutaba, R.: Network Function Virtualization: State-of-the-art and Research Challenges, *IEEE Communications Surveys & Tutorials*, Volume PP, Issue 99 (September 2015).

[8] Pianese, F., Bosch, P., Duminuco, A., Janssens, N., Stathopoulos, T., Steiner, M.: Toward a Cloud Operating System, *IEEE Communications Magazine*, 2010 DOI: 10.1109/NOMSW.2010.5486552.

[9] Online information: https://www.zdnet.com/article/what-is-cloud-computing-everything-you-need-to-know-from-public-and-private-cloud-to-software-as-a/.

[10] Xia, W., Wen, Y., Foh, C., Niyato, D., Xie, H.: A Survey on Software-Defined Networking, *IEEE Comminunication Surveys and Tutorials*, Volume 17, Number 1 (2015).

[11] Online information: https://searchservervirtualization.tech targ et.co m/definition/virtual-machine.

[12] Online information: https://www.docker.com/resources/w hat-container.

[13] Wang, X., Krishnamurthy, P., Tipper, D.: Wireless net-work virtualization. *2013 International Conference on Computing, Networking and Communications*, ICNC, 2013, pp. 818–822.

[14] Principles and Practices for Securing Software-Defined Networks, *ONF White Paper* (January, 2015).

[15] Online information: https://www.opennetworking.org/

[16] Network Operator Perspectives on NFV priorities for 5G, at the *NFV#17 Plenary meeting*, February 21st, 2017, Bil-bao, Spain.

[17] Online information: https://www.sdxcentral.com/nfv/defi nitions/nfv-mano/.

[18] *A White Paper Prepared by the OSM and User Advisory Group*, Issue 1 (May 2018).

[19] Online information: https://osm.etsi.org/images/OSM-Wh itepaper-TechContent-ReleaseFOUR-FINAL.pdf.

[20] Online information: https://www.sdxcentral.com/5g/defini tions/5g-network-slicing/

[21] *5G Communications Systems (5GCOM)*, ETSI views and contribution on 5G standardisation, ETSI (May 2017).

[22] Online information: https://osm.etsi.org/news-events/news

[23] Garcia-Aviles, G., Gramaglia, M., Serrano, P., Banchs, A., POSENS: A Practical Open Source Solution for End-to-End Network Slicing, *IEEE Wireless Communications*, Vol. **25**, no. 5, pp. 30–37, October 2018, DOI: 10.1109/MW C.2018.1800050,

[24] Online information: https://5gtango.eu/

[25] Online information: https://www.prnewswire.com/news-releases/sk-telecom-commercializes-nfv-mano-platform-n amed-t-mano-300486664.html.

[26] *OSM Release FIVE Technical Overview", OSM White Paper*, 1st edition (January 2019).

[27] Online information: https://transition.fcc.gov/bureaus/oet/t ac/tacdocs/reports/2018/5G-Network-Slicing-Whitepaper -Finalv80.pdf.

[28] Online information: https://www.ericsson.com/en/reports-and-papers/white-papers/onap-and-the-telecom-industrys-open-source-journey.

[29] Online information: https://www.ericsson.com/en/blo g/20 19/3/onap-and-5g--the-role-of-automation-in-5g-and-bey ond.

[30] Online information: http://dockerdocs.gclearning.cn/.

[31] Nitin Agarwal – "Docker on OpenStack" , *CERN Openlab Summer Student Report* (2014).

[32] Exploring Opportunities: Containers and OpenStack", *OpenStack White Paper* (2015).

[33] Online information: http://www.electronicdesign.com/dev -tools/what-s-difference-between-containers-and-virtual-machines.

[34] Online information: https:///docs.openstack.org/magnum latest/user/#overview.

[35] Online information: https://docs.openstack.org/murano/lat est/.

[36] Online information: https://docs.openstack.org/kolla/late st/.

[37] Online information: https://kubernetes.io/docs/concepts/o ver view/what-is-kubernetes/.

[38] Online information: https://coreos.com/flannel/docs/latest/

[39] 5GPPP, *Advanced 5G Network Infrastructure for the Fu-ture Internet*, [Online]. Available: https://tinyurl.com/w m8vzeh. [Accessed 15 May 2020].