

Received August 10, 2020, accepted August 18, 2020, date of publication August 21, 2020, date of current version September 4, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3018583

Reducing Service Creation Time Leveraging on Network Function Virtualization

WINNIE NAKIMULI¹, (Graduate Student Member, IEEE),
JAIME GARCIA-REINOSO¹, (Member, IEEE), BORJA NOGALES¹,
IVAN VIDAL¹, (Member, IEEE), DIOGO GOMES², (Member, IEEE),
AND DIEGO LOPEZ³

¹Departamento de Ingeniería Telemática, Universidad Carlos III de Madrid, 28911 Madrid, Spain

²Instituto de Telecomunicações, Universidade de Aveiro, P-3810-193 Aveiro, Portugal

³Telefonica I+D, Distrito Telefonica, 28050 Madrid, Spain

Corresponding author: Jaime Garcia-Reinoso (jgr@it.uc3m.es)

This work was supported in part by the European Commission under the European Union's Horizon 2020 Program (5G EVE Project) under Grant 815074.

ABSTRACT Fifth-generation (5G) networks are envisioned to simultaneously support several services with different connectivity requirements. In this respect, service creation time is a key performance indicator (KPI) for service providers when planning the migration to 5G. For example, the European 5G infrastructure public private partnership (5G-PPP) suggests to reduce this time from 90 hours to 90 minutes, in the different phases of the service creation time KPI identified by this organization. This reduction can be achieved by leveraging on 5G state-of-the-art technologies: network function virtualization, network slicing, software-defined networking, and cloud computing, among others. Although some authors and projects have already studied the service creation time KPI in 5G, there is no literature that comprehensively analyzes and presents results related to each phase of this KPI. In this article, we explore the potential of network function virtualization technologies to reduce service creation time. To this end, we investigate the various phases of the service creation time KPI by designing and implementing, a realistic as well as complex network service that leverages on network function virtualization and related technologies. For our use case, we chose a content delivery network service specifically designed to distribute video. This decision was based on an analysis where we considered several parameters, like the complexity in the phases of design, fulfillment, and service assurance. We dissected all phases of the service creation time KPI required to turn our service blueprint into a deployment by utilizing network function virtualization tools. Henceforth, we defined and conducted several experiments, which were oriented to analyzing the different phases of the service creation time KPI. After analyzing the obtained results, we can conclude that using these new tools permits a substantial reduction in the time taken by each phase of the service creation time KPI.

INDEX TERMS 5G, MANO, network slicing, NFV, service creation time KPI.

I. INTRODUCTION

The fifth-generation (5G) of wireless mobile networks has been designed to boost usability and provide enhanced performance, aiming at supporting new services with stringent requirements. Due to the heterogeneity of these services, they can be classified into three groups [1]: enhanced mobile broadband (eMBB), massive machine-type communications (mMTC), and ultra-reliable low-latency

communications (URLLC). The former covers services requiring a high throughput; mMTC requires low throughput but a large number of devices connected to the network; whereas the latter imposes a very low latency in all parts of the network, including the radio part. To deploy such complex networks over a shared infrastructure, the next generation mobile networks (NGMN) alliance defined the concept of network slicing [2]. With this capability, a 5G mobile network can support the instantiation of specific network slices by creating virtual instances of the main components of the 5G architecture. Thus, a given network slice would be

The associate editor coordinating the review of this manuscript and approving it for publication was Mahdi Zareei¹.

designed to support one of the aforementioned groups of services. One of the main cornerstones of network slicing is network functions virtualization (NFV). ETSI NFV has defined an architectural framework [3] where a management and orchestration (MANO) block controls the lifecycle of virtual network functions, as well as the connectivity among them. This block acts on behalf of the operations/business support system (OSS/BSS), taking advantage of the available hardware resources. Telecommunication operators have shown particular interest in these technologies to boost their revenues, allowing the deployment of new services on top of their 5G infrastructure [4].

Several alliances and partnerships have emerged to speed up the deployment of 5G networks. In Europe, the 5G infrastructure public private partnership (5G-PPP) was created between the European Union and the 5G Infrastructure Association, integrating the essential 5G stakeholders. The contract signed by these parties¹ includes a clause to monitor the progress of the partnership, specifying a set of key performance indicators (KPIs) in the areas of business, performance and societal KPIs. One of the ambitious performance KPIs listed in this document is the service creation time, and 5G-PPP proposes the reduction of this KPI, i.e., the average service creation time from 90 hours to 90 minutes. According to 5G-PPP, the service creation time KPI is composed of five phases: phase 0, platform provision; phase1, onboarding; phase2, instantiation, configuration and activation; phase3, modification; and phase4, termination. In traditional networks, service creation requires the installation and configuration of hardware, potentially from different vendors, connection and reconnection of physical data links, and heterogeneous software configuration. All these actions led to increased delays in making services operational, so reducing all these times will bring significant benefits to all the involved actors. Although some previous authors have studied the 5G service creation time KPI, existing results are incomplete or inconclusive. Therefore, to the best knowledge of the authors, we are the first to present a comprehensive analysis and results regarding all phases of the 5G service creation time KPI.

Accordingly, this article aims to explore the potential of state-of-the-art NFV technologies to effectively reduce the average service creation time KPI of complex network services in all the constituent phases. Thus, it is crucial to analyze the whole lifecycle of any service, from the design stage to the fulfillment of the end-to-end (E2E) service, including service assurance. Subsequently, our study will leverage a complex network service to dissect the different phases of the service creation time KPI, suggested by 5G-PPP. In this respect, we follow a practical approach, using well-known, widely adopted, open-source technologies to automatically deploy a real-world multi-site content-delivery network (CDN) service. This service was selected to showcase the advantages of using NFV technologies under a

realistic context, given its intrinsically challenging operational aspects regarding service design, fulfillment, update, and assurance. We studied the service creation times that may be achieved in the context of our CDN service implementation, analyzing the potential of NFV technologies to scale the resources of a network slice and adapt to varying user demands. The tests have been carried out on a multi-site enabled NFV platform built in the context of the European H2020 5G EVE project [5]. Henceforth, the key contributions of this article are:

- Design of a realistic, moderately complex CDN management service, composed of network slices, implemented across multiple sites in two different countries, that leverage on NFV technologies.
- Implementation and deployment of the CDN service using state-of-the-art open-source NFV technologies, following all the phases involved in the 5G service creation time KPI.
- For each of the phases encompassed in the 5G service creation time KPI, we comprehensively analyze all the activities involved in each phase by conducting several experiments to measure, and quantify the time taken regarding our CDN management service.

The rest of the paper is organized as follows. Section II presents a brief discussion of the other works that have been carried out on the subject matter while pointing out the major differences with our work. Section III provides a discussion of the main 5G technologies used in our work; the tools used to implement these technologies and finally, a brief introduction to content delivery networks. Section IV dissects the different phases of the service creation time KPI, by describing the various experiments designed to analyze these phases. Section V summarizes the time required in each phase of the service creation time KPI, presenting a discussion of the key findings of this article. Finally, Section VI concludes with the most important lessons learned in this work, presenting some future work.

II. RELATED WORK

There have been several 5G-PPP projects whose work has focused, among other tasks, on measuring the 5G service creation time KPI, considering numerous use cases. In this section, we present the definition of the service creation time KPI in each project, measurement results, and how it relates to our work. Besides, we provide a summary table indicating the comparison results.

In the 5G-TRANSFORMER project [6], the service creation time KPI is defined as the time taken to deploy a network service. This time is measured from the time a network service deployment request is sent from the 5G-TRANSFORMER Vertical Slicer to the time a positive response is received from the 5G-TRANSFORMER stack [7]. Accordingly, their service creation time KPI measurements focus on the instantiation, configuration, and activation phase. For the KPI measurement, they considered

¹<https://5g-ppp.eu/contract/>

four use-cases, i.e., Entertainment, E-Health, E-Industry, and mobile virtual network operator (MVNO). All these use-cases were orchestrated using the 5G Transformer platform [7], and the corresponding service creation times have been documented in [6]. In comparison, our work also leverages a complex network service to measure the service creation time KPI considering four phases, however, i.e., (i) Onboarding of descriptors, (ii) Instantiation, configuration and activation, (iii) Modification of the service, and (iv) Service termination. Moreover, the network service was orchestrated using the publicly available open-source tool, i.e., ETSI OSM.

The MATILDA project [8] provides some operational KPI values related to the 5G-PPP service creation time KPI, i.e., onboarding time, deployment time, and scaling time which in our case is similar to service modification phase. They provide values corresponding to these times for different use-cases: (i) Emergency Infrastructure with Service Level Agreement (SLA) Enforcement (5G PPDR), (ii) High-Resolution Media on Demand Vertical, with Smart Retail Venue integration (5GPAGE), and Industry 4.0. Hereto, the use-cases were demonstrated utilizing the MATILDA project test facilities [9]. However, they do not provide a formal definition of the service creation time KPI in regards to their project. Secondly, the methodologies used to measure these KPI differ across the distinct use-cases. Henceforth, we find the provided service creation time KPI results inconclusive.

On the other hand, the 5G-MoNArch project [10] evaluated the service creation time KPI utilizing two testbeds, i.e., The Smart Sea Port testbed in Hamburg, Germany, and the Touristic city testbed in Turin, Italy. Each of these testbeds is capable of supporting different use cases with the aid of network slicing. In the Hamburg testbed, the service creation time KPI was measured considering the following phases: (i) Network Slice instantiation and activation, and (ii) Network Slice re-configuration, which in our case is similar to service modification. Whereas, in the Touristic city testbed, the KPI measurement was decomposed into the following phases: (i) Onboarding, (ii) Instantiation and Activation of the network slices, and (iii) Network Slice modification. The measurement results for each phase for the touristic testbed are documented in [10]. Nevertheless, it is essential to note that the “Instantiation and Activation” phase does not include the configuration stage as proposed by 5G-PPP because the 5G-MoNArch project makes use of pre-provisioned virtual machines (VMs). The 5G-MoNArch virtual infrastructure manager (VIM) platform [11] always maintains a pool of pre-provisioned running VMs to reduce on the network slice deployment time. Consequently, the KPI values presented for the instantiation and activation phase do not include the creation and configuration time of the VMs composing the network slice and, henceforth, incomplete.

Lastly, the 5GTANGO project [12] specifies a performance KPI known as “automation KPI”, which is similar to the 5G-PPP service creation time KPI. The automation KPI

is defined as the time to deploy a network service and should always be as small as possible. The project uses three pilots, i.e., Communications suite, Immerse Media, and Smart Manufacturing. Each of these pilots supports some use-cases, which are orchestrated by the 5GTANGO service platform [13]. In [12], they present the automation KPI results for one of the use cases of the Immersive Media Pilot, i.e., Single-location scenario, where the camera, end-users, and the network service components are located in the same location. The KPI results are split into two phases: Instantiation, configuration and activation time, and Termination time. Accordingly, the minimum, average, and maximum values for each of these phases are provided. For the other two pilots, a qualitative analysis of the Automation KPI is provided. Although the 5GTANGO project provides some results related to two of the phases of the service creation time KPI for one of the pilots, the presented results are partial. Moreover, in the other two pilots, the provided results are qualitative.

A summary of the comparison between our work and the mentioned 5G-PPP projects regarding the provided service creation time KPI results are provided in Table 1.

III. STATE OF THE ART AND BACKGROUND

In this section, we discuss the leading 5G technologies utilized in our service; the open-source tools used to implement them and finally, we provide an introduction to content delivery networks.

A. KEY 5G TECHNOLOGIES

This section presents the main 5G enabling technologies (i.e., Network Slicing and ETSI NFV) used in this work.

1) 5G NETWORK SLICING

5G Network Slicing is a networking concept that supports the creation of complete and independent logical networks on a shared physical network infrastructure [2]. These logical networks are composed of virtualized and physical resources and are designed to satisfy a specific network requirement [14]. For example, on the same physical network infrastructure, we can create two or more network slices, for example: (i) providing low-latency communications to exchange sensors data between vehicles, (ii) provisioning video content from a CDN to those vehicles, and (iii) offering application and network security services to other users.

According to [2], the network slicing paradigm is composed of three layers: Resource Layer, Network Slice Instance Layer, and the Service Instance Layer, as depicted in Fig. 1. The Resource Instance layer provides the physical and virtual network resources (which may be shared and/or dedicated) to the network slices. At the Network Slice Instance Layer, these network resources and functions are combined in diverse ways to form complete and isolated logical networks that satisfy the network requirements of the slice-supported services. Lastly, the Service Instance layer represents the services supported by the network slice, e.g.,

TABLE 1. Related work - Service creation time KPI results evaluation.

Phase	5G-TRANSFORMER	MATILDA	5G-MoNArch	5GTANGO	OUR WORK
Phase0: Platform provision	Not Applicable	Not Applicable	Not Applicable	Not Applicable	Not Applicable
Phase1: On-boarding of descriptors	X	✓	✓	X	✓
Phase2: Instantiation, configuration and activation	✓	✓	Only instantiation and activation	✓	✓
Phase3: Modification of the service	X	✓	✓	X	✓
Phase4: Service Termination	X	X	X	✓	✓
Final remark on presented results	Incomplete	Inconclusive	Incomplete	Incomplete	Complete

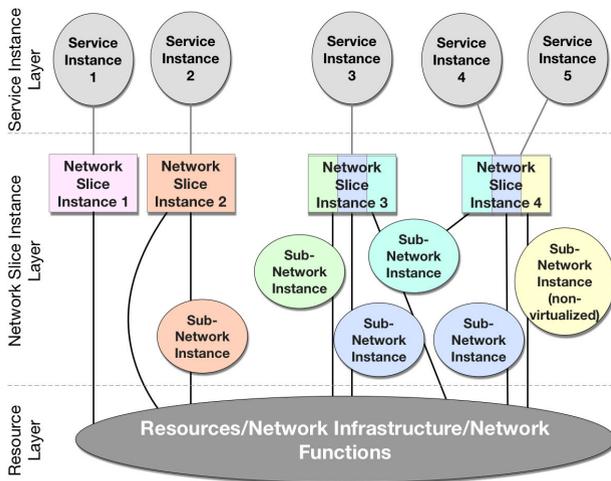


FIGURE 1. Network slicing paradigm [2].

a network slice supporting a CDN service is also capable of supporting services that require load-balancing mechanisms.

To create and instantiate network slice and network slice subnet instances according to the use case requirements, network slice templates (NSTs) are used [15].

2) ETSI NFV

With the impending advent of the fifth generation of mobile networks or 5G, the softwarization of network functions, commonly known as network functions virtualization (NFV), has recently acquired particular relevance, coming to the forefront in the research and development efforts of relevant stakeholders in the telecommunications market.

NFV reduces the dependency of network operations on specific-purpose proprietary hardware devices, with the utilization of network appliances that can be implemented in software and run in more general-purpose cloud/edge computing environments. On the one hand, this enables to alleviate investment and maintenance costs, by a) using commodity equipment that can be provided attending to economies of scale; b) supporting the on-demand deployment of value-added network services in a reduced timeframe, and c) enabling the flexible allocation of infrastructure resources to network services and tenants and adapting resource provisioning to varying service demands. On the other hand, the utilization of NFV technologies reduces the development cycle of new functions and services, which may be prototyped and tested in production-like environments and facilitates

their installation and evolution in operational environments. Moreover, it fosters innovation, providing better opportunities to reach the market of network appliances to telecommunication operators and vendors and other stakeholders in the sector of information technology and communications, such as vertical service providers, software developing companies, and academia.

Standardization activities on NFV technologies are primarily conducted by the European Telecommunications Standards Institute (ETSI), through a specific Industry Specification Group. The ETSI NFV reference framework [16] specifies the main building blocks and open interfaces that are needed for a functional NFV deployment, as well as for the interoperation of different vendor implementations.

In this reference framework, a network service descriptor (NSD) clearly defines any network service that has to be deployed using NFV technologies. An NSD is composed of virtual network functions (VNFs) and the virtual links connecting these VNFs. The VNFs provide the software implementation of network functions and service specific-functionalities utilizing their constituent virtual deployment units (VDUs). A VDU refers to a virtualized environment that hosts a network function, and a VNF can be composed of one or more VDU components. In addition, VNFs are interconnected to build a network and vertical-specific services. The NFV framework uses virtual networks, which are established as abstractions over the NFVIs. The relationship between network slices, network services, virtual network functions, and virtual deployment units is shown in Fig. 2. VNFs and, in turn, VDUs, are executed over a programmable substrate of hardware and software resources, which may be provided by server computers and other heterogeneous capacity equipment. These types of equipment conform to what is commonly known as the NFV infrastructure (NFVI), which supports the instantiation of VNFs with the utilization of virtualization technologies.

A management and orchestration (MANO) system coordinates all the operations related to the lifecycle management of network and vertical-specific services on the NFV infrastructures. These include the commission/decommission of VNFs and their deployment, interconnection, and termination, using the open interfaces defined by the ETSI reference architectural framework. To this purpose, the MANO system is further decomposed into three components: the virtual infrastructure manager (VIM), which provides the functions needed to allocate, deallocate and scale the NFVI compute,

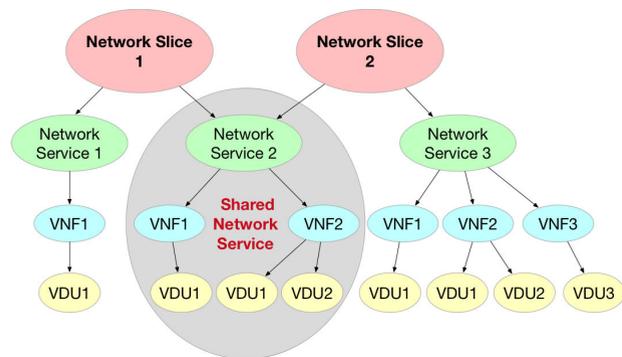


FIGURE 2. Relationship between network slices, network services, VNFs and VDUs.

storage and network resources to running VNFs, as well as to monitor the NFVI status; the VNF manager (VNFM), in charge of managing the VNF lifecycle (i.e., instantiation, configuration, modification, and termination); and the NFV orchestrator (NFVO), which coordinates the allocation of resources interacting with multiple VIMs, as well as the lifecycle of network and vertical-specific services by interfacing with different VNFM entities.

B. MANAGEMENT AND NETWORK ORCHESTRATION COMPONENTS

In this section, we introduce two of the most used solutions to implement the MANO components presented in subsection III-A2. For the NFVO and VNFM components, we chose the ETSI OSM [17] platform, whereas OpenStack [18] was chosen as the VIM solution. The motivation behind these choices and the main functionalities used in each platform (i.e., ETSI OSM and OpenStack) are discussed in the subsequent sections III-B1 and III-B2 respectively.

1) ETSI OSM

ETSI Open Source MANO (OSM) is an ETSI initiative to create an Open Source NFV-MANO framework aligned with its reference architecture and standards [17]. ETSI OSM separates virtual resource orchestration from service orchestration, resulting in a modular framework which can be extended in several directions without affecting its overall operation. This modular design principle allows ETSI OSM to provide an end-to-end network service orchestration functionality through the interaction of the core components of its architecture.

In order to carry out its modular architectural framework, ETSI OSM adopted the cloud-native implementation technique to offer the functionalities encompassed by the MANO stack as a set of standalone components. These components have been implemented using the container virtualization technology (in particular, Docker containers starting from release FOUR). This technology allows ETSI OSM to be flexibly and agilely deployed, reduce resource consumption to carry out its operation and facilitate the integration of forthcoming modules that will extend the functionality offered by the software stack.

Within these components comprising the ETSI OSM architecture, the module referred to as Resource Orchestrator (RO) plays a fundamental role since it is the block in charge of managing the allocation of computational resources (i.e., the compute, network and storage). In turn, these will accommodate the subsequent execution over an NFVI of various virtual network functionalities included in a network service. For this purpose, the RO supports the interactivity with a large variety of VIM solutions such as OpenVIM [17], OpenStack [18], VMware Cloud Director [19] and Amazon Web Services (AWS) [20].

Another major aspect contributed by the RO module to the ETSI OSM stack is the capacity to deploy multi-site network services by managing the resource allocation across several NFVI domains distributed in different geographic locations and controlled by the diverse types of VIMs. In this context, the RO also offers a plug-in to coordinate the operations with a WIM (WAN Infrastructure Manager) to enable the data exchange between VNFs running in different NFVI domains, configuring dynamically for that purpose the intermediate network entities that will allow the inter-site communications.

Concerning the service orchestration, the entity in charge of this milestone is closely linked to the RO and is called the Life Cycle Management (LCM) module. In this case, the LCM serves the operations related to the lifecycle management, handling operations such as the deployment, scaling, and deletion of network services once the RO has completed the required resource allocation. This management operation is included in those services that encompass several network services as an entity, commonly referred to as network slices by the ETSI OSM community. Thus, the ETSI OSM software stack provides a functional block aligned with the ETSI NFVO (see section III-A2) through the interoperability between the LCM and the RO modules.

On the other hand, the VNF configuration, and abstraction (VCA) module allows the configuration of the VNFs composing the network services, including both the initialization of the services to be provided by the VNF (Day-1 configuration) and the execution of the defined operations at run-time (Day-2 configuration). In particular, the VCA module leverages on Juju [21], which is an open-source application tool for software modeling, to carry out the configuration and monitoring tasks associated with the VNFs in OSM. Furthermore, the VCA module communicates with the LCM aiming to manage the lifecycle of the VNFs through the interface provided by the network to the VNF configuration (N2VC) plugin. Thus, the VCA includes the functionality associated with the VNFM entity specified within the ETSI-NFV architecture.

Moreover, the monitoring (MON) module is responsible for collecting the VNF and VIM metrics specified inside the VNF descriptor. Inside the VNF descriptor, the metrics to be monitored by the ETSI OSM platform are specified both at the VNF and virtual deployment unit (VDU) levels. For performance management purposes, the MON module exposes these metrics to a Prometheus module [22], which

is used as a time-series data store for the metrics. Once inside Prometheus, these metrics can be queried and displayed using any metric analytic tool capable of retrieving Prometheus data. In addition, inside the VNF descriptor, it is also possible to set scaling policies (in/out) for a VDU component of the VNF. This feature is also known as “Auto-scaling”, in the ETSI OSM platform, which implies deploying more instances (i.e., scaling-out), or removing the newly deployed instances (i.e., scaling-in) of the VDU depending on specific criteria. For a specific VIM metric being monitored, a user sets threshold values for which the scaling operations will be triggered for a VDU component(s) inside the VNF descriptor. Once the user has set these threshold values, then the Policy (POL) module of ETSI OSM sets alarms inside the MON module for these metrics. After that, the MON module continuously monitors these metrics. Once the set threshold values have been traversed, the MON module places a notification on the ETSI OSM Kafka bus [23], which is used to ensure asynchronous communication between its different modules. This notification is consumed by the POL module, which triggers the configured scaling operations for this scenario, according to the VNF descriptor. These scaling operations are consumed by the LCM module, which either scales out or in a given VDU component(s) of the VNF, depending on the received notification.

As mentioned earlier, in ETSI OSM, network slices are modeled as a composition of network services, i.e., the network slice subnets of a network slice are synonymous to ETSI OSM network services. ETSI OSM offers two modes of handling network slices: full end-to-end (E2E) management (integrated modeling) and standalone management. In full E2E management, ETSI OSM acts as a slice-manager and therefore handles the entire lifecycle operations of the network slice. As the slice manager, ETSI OSM supports the sharing of the same network service(s) between different network slices. In this mode, ETSI OSM uses network slice templates (NSTs) to define the slice requirements, composite network services, and the virtual links connecting the network services. Whereas, in the standalone management mode, the lifecycle management operations of the network slice are handled by an external agent, and ETSI OSM is only responsible for orchestrating the network services associated with the slice.

2) OpenStack

OpenStack is an open-source cloud computing platform capable of pooling large amounts of virtual resources (i.e., compute, storage, and network), building and managing public, and private clouds [18]. As a VIM solution, OpenStack is supported by a large, vibrant community of developers and end-users, with a new release every six months with advancements and additional features. Besides, OpenStack is always accompanied by excellent user documentation for each release. Moreover, it does not require any specialized vendor hardware, but rather runs on standard general-purpose hardware. OpenStack is composed of a number of plug and

play services that can be bundled up in different ways, to provide customized cloud deployments to OpenStack users. Henceforth, the main OpenStack services are:

- **OpenStack Compute:** The compute service also known as “Nova” is the core component of the OpenStack cloud computing platform, and is therefore used to provision and control cloud computing systems. OpenStack compute service relies on other services to achieve this functionality: (i) Identity service for authentication and authorization to access all the required OpenStack services, (ii) Image service to provision the images to be used by the compute instances, (iii) Networking service to provide the physical and virtual networks that the compute instances connect to, and (iv) Placement service for resource and resource-usage tracking.
- **OpenStack Image:** The Image service alternatively known as “Glance” provides a RESTful API that enables an OpenStack user to register, discover, provision and store virtual machine (VM) images, as well as to query other data related to the VM images, such as image metadata. This metadata may include such information as the image disk and container formats.
- **OpenStack networking:** The networking service alias “Neutron” is used to create, and manage networks and/or sub-networks. During instance creation, the OpenStack networking service interacts with the compute service to create network interfaces and provide the required connectivity to the compute instances. OpenStack networking provides two networking options, i.e., pre-created and tenant networks. The former networks are created beforehand by OpenStack users inside OpenStack and are used to provide layer-2 (bridging/switching) connectivity between compute instances and the physical infrastructure. The latter are private networks created on-demand by OpenStack users that augment pre-created networks with layer-3 (routing) connectivity. Tenant networks can be connected to the physical network infrastructure with the help of OpenStack routers. Furthermore, to create tenant networks, an OpenStack user does not need to know about the existing physical network infrastructure, whereas to create pre-created networks, an OpenStack user needs in-depth knowledge about the existing physical network infrastructure.
- **OpenStack Telemetry:** The telemetry service is composed of metering, monitoring, and alarming services. In our experiment, we only focused on the metering service, which is in charge of efficiently polling, collecting, storing, and publishing metering data produced by OpenStack services. To achieve this functionality, OpenStack utilizes two software tools; Ceilometer [18] and Gnocchi [24]. Ceilometer is in-charge of metering data polling, as well as collection, and subsequently publishes these data to “Gnocchi”. Gnocchi provides time-series storage for OpenStack

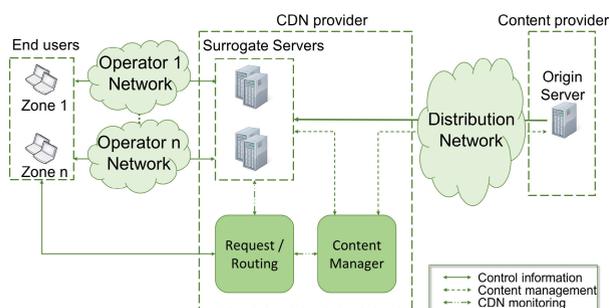


FIGURE 3. Block diagram of a CDN.

data in a persistent and scalable way. In addition, Gnocchi offers resource-indexing for OpenStack resources.

C. INTRODUCTION TO CONTENT DELIVERY NETWORKS

This section presents a brief introduction to Content Delivery Networks (CDN) [25]. A CDN arranges a large number of servers, usually placed close to end-users, facilitating the reduction of the latency to distribute content, improving the quality of experience of end-users. Usually, content providers contract CDN providers to delegate their content, so end-users of content providers can access such information from servers managed by CDN providers. Fig. 3 shows the main building blocks of a complete system, including a content provider, a CDN provider, and end-users. Focusing on the blocks of the CDN provider, it is possible to distinguish three main blocks: (i) Surrogate Servers (SS) offering content to end-users, (ii) Request/Routing block, which receives requests from end-users and redirects these requests to the most suitable Surrogate Server, and (iii) the Content Manager, which manages the distribution of the original content from the Origin Server, where the content provider has all the information to be delivered to end-users, to the Surrogate Servers.

There exist different mechanisms to implement the services provided by these blocks. For example, the Request/Routing block can be implemented using the *http* redirection mechanism, where web servers redirect part of its routes to other web servers; URL rewriting, requiring modifications to the web pages; anycasting, where different web servers have the same IP address; and a mechanism based on Domain Name System (DNS) redirection, which is one of the most used mechanisms to implement this block in CDNs. Because of the popularity of DNS redirection, this will be the mechanism used in this work, so a brief introduction is presented next.

DNS [26] is a service mainly used to map domain names to IP addresses. Usually, end-users have software DNS clients used to resolve names to IP addresses, which in turn contact DNS servers. These DNS servers are hierarchically distributed: after receiving a DNS request, a DNS server may redirect or forward such request towards another DNS server, until an authoritative DNS for the requested domain

is reached. This redirection is exploited by the DNS-based mechanism to implement the Request/Routing block in a CDN. After receiving a DNS request from an end-user device, the DNS server of the CDN selects the most appropriate Surrogate server to handle such requests and replies to the request using the IP address of that server. After receiving the reply, the end-user will use that surrogate server to request the proper content.

IV. EVALUATING THE SERVICE CREATION TIME KPI

This section is devoted to evaluating the service creation time KPI of a complex service, mainly focusing on a content delivery network (CDN). In order to analyze the average service creation time of a service using state-of-the-art NFV technologies, in this article we have decided to create a complex service composed of three main providers offering different services: (i) a video content provider offering access to its video catalogue through its web portal to end-users by means of, (ii) a CDN provider offering a service to automatically deploy and scale surrogate servers close to end-users who are connected to the Internet using, and (iii) a telecommunications operator. For the sake of readability and without any loss of generality, we assume a scenario with only one telecommunications operator offering access to a 5G network, MANO platforms, and the capacity to handle network slices.

As shown in Fig. 4, in the design of our service, we envision a video content provider with clients located both in Spain and Portugal. To provide an outstanding quality of experience to their clients, the content provider contracts the services of a CDN provider who has a pool of surrogate servers located both in the 5TONIC laboratory² (Spain) and IT-Aveiro³ (Portugal). The telecommunications operator places these two services in independent slices, to isolate both services. Henceforth, the overall scenario was designed and executed using the different tools provided by ETSI OSM and OpenStack platforms in the 5TONIC laboratory and IT-Aveiro datacenter.

To better understand the implication of the service creation time KPI, it is essential to highlight once again all phases involved in this time, namely: phase 0, platform provision; phase 1, onboarding of descriptors; phase 2, instantiation, configuration and activation; phase 3, modification; and, phase 4, termination [27]. These phases showcase the need to perform several steps to deploy and manage a service on a 5G network completely. In the rest of this section, we will follow the order of these phases to structure all experiments performed.

A. PLATFORM PROVISION

As already stated in Section I, the platform used in this article was built in the context of the European H2020 5G EVE

²5TONIC is an open research and innovation laboratory focusing on 5G technologies. <https://www.5tonic.org/>

³Instituto de Telecomunicações in Aveiro, Portugal. <https://www.it.pt/ITSites/Index/3>

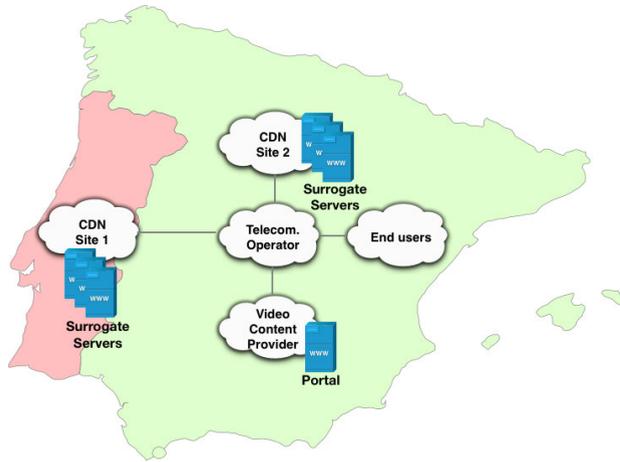


FIGURE 4. High overview of the CDN management service.

project, which in turn is based on the infrastructure deployed for the H2020 5GinFIRE project [28]. In this section, we discuss the steps involved in getting this infrastructure platform ready to provide access to the users of the aforementioned projects and, in particular, for the experiments defined in this work.

In order to carry out the experimentation described in the introduction of this section, here we present an NFV system mainly constituted by a software MANO stack and three private cloud platforms, as depicted in Fig. 5. The cloud platforms represent the NFV infrastructures that enable the deployment of the designed CDN service for the experimentation procedure. Related to management and orchestration, ETSI OSM (see section III-B1) is in charge of providing the MANO implementation and, in particular, our experiments utilize OSM Release SEVEN [17]. To facilitate the installation and maintenance of the OSM software stack, it is provided in a virtual machine within the 5TONIC laboratory and in compliance with the recommended footprint established by the OSM community (2 CPUs, 8 GB of RAM, and 40GB of hard drive).

Besides the OSM stack, the 5TONIC laboratory hosts two of the aforementioned cloud platforms, both based on the OpenStack Ocata release (see section III-B2). For the first platform, two high-profile server computers provide a computational capacity of 48 vCPUs, 256 GB of RAM, and 8 TB for storage. To manage these resources, the OpenStack node that fulfills the VIM operations (i.e., the OpenStack controller) runs in a virtual machine with 4 vCPUS, 16 GB RAM, and 180 GB of storage. This cloud platform is represented as “5TONIC-site1 DC” in Fig. 5. A second cloud platform is also included in the 5TONIC laboratory, encompassing three server computers that account for total resources of 24 vCPUs, 96 GB RAM, and 6 TB of storage. Analogously to the previous case, the VIM of this cloud platform is based on an OpenStack controller, which runs in a virtual machine with 4 vCPUs and 12 GB RAM, 600 GB of storage. The second cloud platform is referred to as “5TONIC- site2 DC” in Fig. 5,

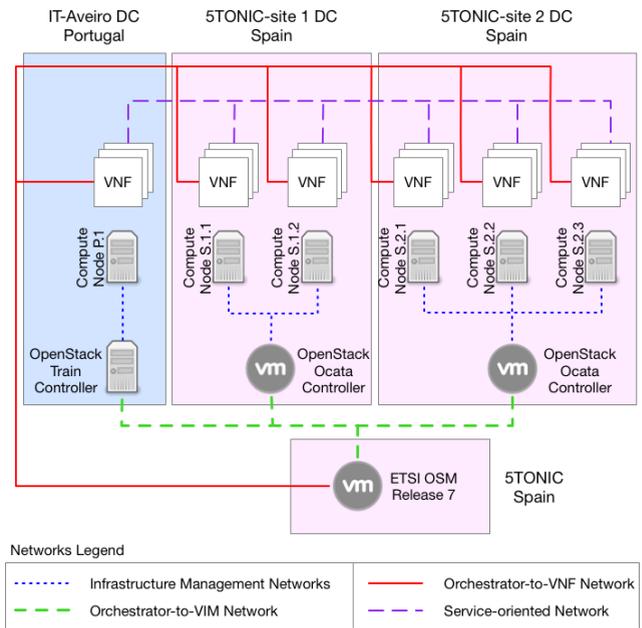


FIGURE 5. Platform infrastructure used to implement and support the CDN management service.

The resources of both cloud platforms are exposed to the OSM stack through their respective VIMs, which are compliant with ETSI OSM Release SEVEN. Besides, the cloud platforms integrate the OpenStack Telemetry service, supporting the performance metrics collection of the deployed VNFs. Thus, the experimentation procedures can benefit from the auto-scaling functionality provided by ETSI OSM. Moreover, both infrastructures implement the OpenStack networking service, where both the pre-created and tenant network options have been configured.

To complete the testbed infrastructure, IT-Aveiro provides a third cloud platform to the NFV system (shown as “IT-Aveiro DC” in Fig. 5). The cloud platform features a two-node deployment based on the OpenStack Train release. The controller node managing this NFV infrastructure is a physical server. The compute node is also a physical server and has the computational capacity of 24 vCPUs, 200GB of RAM, 1 TB for computing storage, and 1 TB for volume storage.

On the other hand, it is worth mentioning that the testbed integrates the required networking capacity that enables the proper operation of the complete NFV system, supporting the deployment of multi-site network services over the three cloud platforms. In this respect, the following networks have been configured:

- 1) Infrastructure management networks (i.e., the blue round dot line in Fig. 5): these networks aim to allow each VIM to manage the computational resources of the cloud platform under its control. Therefore, these networks exist in all the three cloud platforms of the NFV system.
- 2) Orchestrator-to-VIM network (i.e., the green long dash line in Fig. 5): this network is in charge of

supporting the reservation and allocation of the necessary resources for enabling the subsequent network services deployment. Furthermore, this network is in charge of the lifecycle management of the stated network services and slices. This network supports the communication of the ETSI OSM stack with each of the VIMs, to support NFV management and orchestration operations.

- 3) Orchestrator-to-VNF network (i.e., the solid red line in Fig. 5): the objective, in this case, is to allow the ETSI OSM stack to control and monitor the VNFs lifecycle (i.e., get the VNFs state information and support the scaling operations based on this information), as well as to configure the VNFs during their deployment.
- 4) Service-oriented network (i.e., the purple long dash-dot line in Fig. 5): this network supports multi-site data communications among VNFs deployed on different cloud platforms.

The 5TONIC laboratory provides specific services to support all the networks mentioned above, which need to be realized considering the distributed nature of the NFV system (the ETSI OSM stack and two cloud platforms are available in Madrid, whereas another cloud platform is located at Aveiro). In particular, a VPN service has been configured at 5TONIC [28], allowing the communications with the IT-Aveiro site over secure access with certificate-based authentication across the Internet. The network path between 5TONIC and IT-Aveiro is supported by the Spanish and Portuguese national research and education networks, RedIRIS and RCTS, respectively, which are interconnected through the GÉANT pan-European network. To get a better understanding of the performance provided by this network path, we have measured the maximum throughput and the Round Trip Time (RTT) delay on the path. To this purpose, we have used an open-source traffic scheduler, Traffic [29], to send two consecutive TCP flows at the maximum possible rate, one on each direction of the network path. A pair of TCP flows was scheduled every hour during a period of 10 days. Our results indicate a maximum average throughput of 66.4 Mb/s from 5TONIC to IT-Aveiro, and 61.3 Mb/s in the opposite direction. The average RTT observed by the TCP flows scheduled in the 5TONIC to IT-Aveiro direction was 26.56 ms, considering a period of one day (24 average RTT values). The standard deviation was 0.71 ms, showing a reduced dispersion of the average RTT during the day. These results suggest that the performance of the network path is appropriate to perform the experimentation activities considered in this article.

Thus, the outlined testbed is built with an appropriate amount of hardware resources, allowing the deployment of a wide range of multi-site network services and slices across all the platforms described in this section. The design of this infrastructure requires in-depth knowledge of computer networks, but its management is not as demanding as the former stages.

B. ONBOARDING OF DESCRIPTORS

Even though this phase of the service creation time only comprises the time required for the onboarding of descriptors, for the sake of completeness and in order to better understand the experiment results in next phases, we also include the previous steps to the onboarding phase where the service has to be defined. Thus, in the following sub-sections, we will explain further the following steps: (i) design layout, (ii) formal service preparation, and (iii) the descriptors onboarding phase.

1) CDN MANAGEMENT SERVICE DESIGN LAYOUT

As previously discussed in Section I, in this article, we plan to analyze the service creation time of a complex service. To this end, we plan to provide a service where end-users contract video streaming services from the video content provider, who, in turn, contracts a CDN provider to provide the service to end-users effectively. This way, after an end-user, selects content from the web portal offered by the video content provider, the actual request of the video will be forwarded to the CDN provider, which stores all or part of the content offered by the video content provider. In our design, the CDN service will be implemented in two different sites. The proper site to serve a given request will be selected by the CDN provider management system, using the Domain-Name System (DNS) service.

To efficiently and dynamically compose multiple services on top of a shared infrastructure, our design will leverage the network slicing functionality provided by ETSI OSM. The conducted experiment involves two network slices: the CDN network slice (CDN-NSlice) and the video content provider network slice (VCP-NSlice) implemented across multiple sites, as shown in Fig. 6. The CDN network slice (i.e., the green slice) is used to deliver the video content to the end-users, whereas the video content provider network slice (i.e., the pink slice) is the one that provides the end-user with Internet connectivity. It is essential to highlight that our infrastructure does not provide the intrinsic characteristics of network slices yet to support the three groups of services described in Section I (i.e., eMBB, URLLC and mMTC). This functionality will be incorporated in the next upgrade of the platform. In this work, we only test the capacity of ETSI OSM to compose several network services (possibly shared) into a network slice to provide a given functionality.

The CDN network slice is composed of three network services: CDN-NServ-S, offering CDN services to end-users located in Spain; CDN-NServ-P, offering CDN services to end-users located in Portugal; and the shared operator NS (Op-NServ) is used to interconnect the two previous network services. The CDN-NServ-S provides the CDN service by utilizing the surrogate servers in Spain (SS-S), while CDN-NServ-P achieves the same purpose through the aid of the surrogate servers in Portugal (SS-P). The two CDN network services (i.e., CDN-NServ-S and CDN-NServ-P) are connected through the shared operator network service,

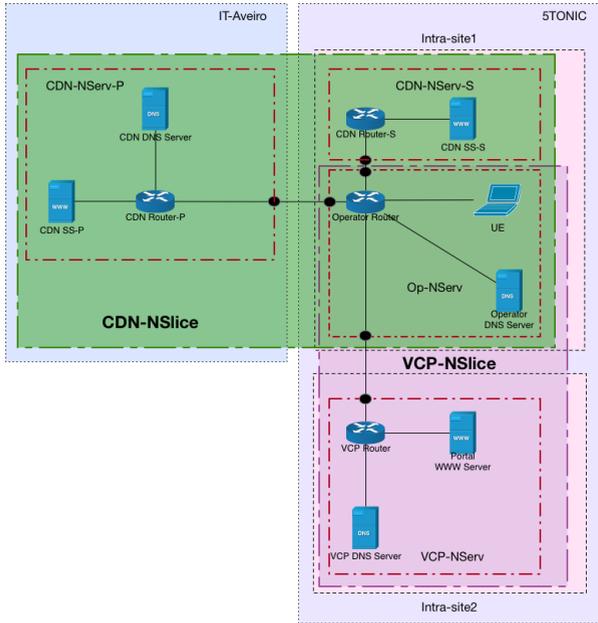


FIGURE 6. CDN management using 5G and NFV principles experiment design.

TABLE 2. CDN and video content provider network slices deployment information.

Network Slice	NServ	DC	DC-loc
CDN-NSlice	CDN-NServ-S	5TONIC-site1	Spain
	CDN-NServ-P	IT-Aveiro	Portugal
	Op-NServ	5TONIC-site1	Spain
VCP-NSlice	Op-NServ	5TONIC-site1	Spain
	VCP-NServ	5TONIC-site2	Spain

which plays the role of the telecommunications operator in our experiment set up.

Conversely, the video content provider network slice (VCP-NSlice) consists of two network services: video content provider network service (VCP-NServ) and the shared operator network service (OP-NServ). The VCP-NServ is used to enable the end-user to access the web portal with all available content. In contrast, the shared operator network service is used to link the video content provider network slice to the CDN network slice and the end-users. It suffices to say, both network slices, VCP-NSlice and CDN-NSlice, share the operator network service (Op-NServ).

To deploy a realistic environment, the CDN network slice spans across two locations: one in the 5TONIC laboratory in Madrid, Spain, the other in the Instituto de Telecomunicações in Aveiro (IT-Aveiro), Portugal. On the other hand, the video content provider network slice is also implemented across two data-centers; however, both data-centers are located inside the 5TONIC laboratory. Table 2 presents a summary of the network slices deployment information, i.e., constituent network services (NServ(s)), data-centers (DC(s)) and the location where these data-centers are deployed (DC-loc).

TABLE 3. Network service(s) constituent VNFs and VDUs in reference to Fig. 6.

Network Service	Functionality	VDU(s)
CDN-NServ-S	Routing	CDN Router-S
	Load-balancing	CDN SS-S - HAProxy
	Webserver	CDN SS-S - Apache
CDN-NServ-P	Routing	CDN Router-P
	DNS	CDN DNS Server
	Webserver	CDN SS-P
Op-NServ	Routing	Operator Router
	DNS	Operator DNS server
	Client	UE
VCP-NServ	Routing	VCP Router
	DNS	VCP DNS server
	Webserver	Portal WWW server

2) FORMAL SERVICE PREPARATION

In this part, we explain the procedures undertaken to precisely define all services used in this validation, following the information models specified by ETSI OSM. This phase aims to prepare a set of descriptors that will be onboarded in the following phase, which is the intermediate step between the design and the operation of the service.

As already explained in section III-B1, ETSI OSM has defined network slices as a composition of network services (NServ), which in turn are composed of virtual network functions (VNF). Similarly, virtual network functions are composed of one or more virtual deployment units (VDU). Furthermore, we know that network slices, network services, and VNFs make use of descriptors in order to deliver the required customized service.

Accordingly, for each of the composite network services in the CDN and video content provider network slices presented in section IV-B1, we introduce the constituent virtual network functions (VNFs) and virtual deployment units (VDUs) in Table 3. In order to prepare a service, an experiment developer⁴ has to take the following steps:

- Firstly, the developer has to prepare the descriptors for all the constituents VNFs, i.e., VNF descriptors. These VNF descriptors describe the compute, storage, and network resources required to achieve the required VNF.
- Secondly, the developer has to prepare and compile the day-1 & day-2 configurations package for each of the VNFs included in the service. As mentioned in section III-B1, in ETSI OSM these VNF configurations are carried out by Juju through the use of charms; a collection of configuration files and scripts that are used to deploy and manage VNFs efficiently and reliably. The compiled charms are packaged together with the

⁴In this case, an experiment developer is someone very familiar with the ETSI OSM and OpenStack platforms, henceforth an expert at creating VNF descriptors, network service descriptors, as well as network slice templates. For inexperienced developers, designing the descriptors can be challenging at the start.

VNF descriptors inside the VNF packages. Additionally, the developer has to reference these VNF charms packages inside the VNF descriptors. The day-1 VNF configurations include actions automatically called when the VNF is launched for example, bringing up network interfaces, setting packet routes, and enabling packet forwarding. In contrast, the day-2 configurations include actions that are to be called on-demand by the telecommunications network operator after the service is already running for example, triggering scaling actions and updating a software package.

- Thirdly, the developer has to define the network service descriptors (NSD) for all the constituent network services. These network service descriptors are used to define the composite VNFs i.e., VNF descriptors and their interconnections to implement the required network service. In addition, they are used to define the network service configuration information.
- Finally, the developer has to prepare the network slice templates (NSTs) for the constituent network slices. These NSTs serve to capture the composite network slice subnets (in this case, network services) and the virtual links connecting the composite network services. Network slice templates reference the network service descriptors, i.e., NSDs composing a slice and their interconnections. They can also contain slice properties, such as slice type (i.e., eMBB, URLLC, and mMTC) and quality of service parameters. We have provided all the descriptors (i.e., VNF and NSD) and the NSTs used in our work.⁵

3) DESCRIPTORS ONBOARDING PHASE

After all the descriptors are ready, the experiment developer has to on-board all these files to the ETSI OSM platform. Besides, developers have to contact the telecommunication operator administrator to upload the required VNF images to the OpenStack image service and ensure that the necessary pre-created networks already exist inside the OpenStack networking service. However, if the required pre-created networks were not already present, the telecommunication operator administrator would have to create them. Once this was done, the experiment developer could contact the telecommunications operator to schedule the execution of the already on-boarded service descriptors.

C. INSTANTIATION, CONFIGURATION AND ACTIVATION

After the service was correctly designed, the next stage is to instantiate, configure, and activate it. The previous step is new compared to the traditional way of guaranteeing the fulfillment of a service. With the integration of the last two steps in the overall process, it is vital to reduce the service creation time. We have designed an instantiation, configuration, and activation experiment to collect results for this phase of the service creation time, which will be presented next. Care to

note, the ETI OSM platform has to perform the following steps in sequential order: (i) Creation and configuration of the requested networks as per the VNF descriptors, NSDs and NSTs, (ii) Creation and configuration of the VNFs as per the VNF descriptors, (iii) Interconnection of the VNFs to form network service(s) as per the NSDs, and (iv) Finally interconnection of the network services to form the network slice(s) as per the NST(s).

In this experiment, we tested the amount of time it takes to launch every network service and, in turn, every network slice already presented in the previous sections. For these tests, we considered two network slicing deployment scenarios:

- Scenario #1:** In this deployment, the CDN network slice was launched first, followed by the video content provider network slice. In reference to Table 2, this implies that the CDN-NServ-S and Op-NServ are launched simultaneously in 5TONIC-site1, while the CDN-NServ-P is launched in the IT-Aveiro site. Once this instantiation is complete, the video content provider network slice is launched, which in turn launches the VCP-NServ in 5TONIC-site2 and connects to the already running shared Op-NServ in 5TONIC-site1. It should be noted that part of the CDN network slice runs in 5TONIC-site1, whereas the video content provider network slice runs in 5TONIC-site1 (Op-NServ) and 5TONIC-site2 (VCP-NServ).
- Scenario #2:** In this case, the video content provider network slice was launched first and subsequently the CDN network slice. With reference to Table 2, this means that the Op-NServ is launched in 5TONIC-site1, and in parallel, the VCP-NServ is launched in 5TONIC-site2. Once this is complete, the CDN network slice is launched, which simultaneously launches the CDN-NServ-S in 5TONIC-site1 and the CDN-NServ-P in IT-Aveiro. It is important to note here, that the video content provider network slice runs in 5TONIC-site2, whereas the CDN network slice runs in 5TONIC-site1.

For each scenario, we instantiated each of the network slices ten times, as the results were stable enough for comparison purposes, as will be shown next. All experiments were executed under similar conditions, starting with an empty and clean environment where all available resources were assigned to these tests. At each instantiation, the amount of time it takes to launch each network service and consequently, each network slice as provided by the ETSI OSM platform was recorded. This time is computed as the difference from the time a network slice instantiation request is sent to the ETSI OSM platform, to the time the network slice is successfully deployed and active. Henceforth, the average results for each deployment scenario #1 and #2 are shown in Fig. 7 and Fig. 8 respectively, all experiments were successfully deployed.

From Figures 7 and 8, we see that deployment scenario #2 launches on average much faster than scenario #1.

⁵https://github.com/Winnie-Nakimuli/NFV_Service_Creation_time

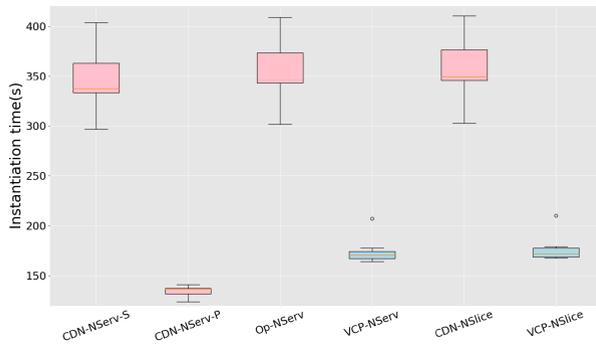


FIGURE 7. Network service(s) and network slice instantiation, configuration, and activation times for Scenario #1.

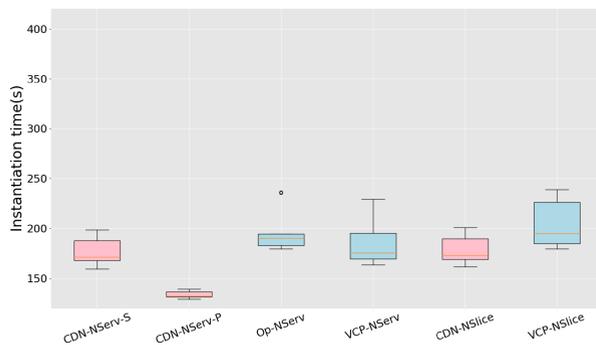


FIGURE 8. Network service(s) and network slice instantiation, configuration and activation times for Scenario #2.

This is because, in scenario #1, we have up to 6 VDUs instantiated simultaneously in 5TONIC-site1. In contrast, in Scenario #2, we have only 3 VDUs instantiated at once in 5TONIC-site1, and the other 3 VDUs are instantiated in 5TONIC-site2. Accordingly, network slicing deployment scenario #2 launches much faster than scenario #1.

Therefore, when designing network slices and network services, it is crucial to account for the number of VDUs to be instantiated in each data center. Consequently, this will affect the instantiation, configuration, and activation time of the network services and, in turn, the network slices.

D. MODIFICATION OF THE SERVICE

For the service modification (or assurance) phase, we have designed two experiments to show how to manage a running service and compute the performance in different conditions. The first experiment explained in IV-D1 is oriented towards showing that it is still possible to manage all services using traditional mechanisms. The goal of the second experiment is to compare the traditional management of the services with the one using the new tools offered by NFV and especially the auto-scaling tools implemented by ETSI OSM using the corresponding OpenStack functionalities. This last experiment is described in IV-D2.

After all the services shown in Fig. 6 are completely instantiated, the first experiment starts with the CDN provider

redirecting all requests to the server located in Portugal. After receiving requests from clients located in Spain, the CDN service is manually⁶ configured to redirect those requests to the Spanish site, where one surrogate server is already prepared to receive requests. In the last experiment, we increase the number of clients in Spain requesting content to the CDN, which will trigger the auto-scaling function of the ETSI OSM platform.

1) DNS REDIRECTION

In this experiment, we investigated the ability of the CDN service to redirect end-user requests to the proper surrogate servers, i.e., the proper web servers designated to service end-users given the prevailing network traffic demands.

For this test and in reference to Fig. 6, the Operator DNS server was configured as a forwarder with two forwarding zones, i.e., *cp.com* and *cdn.com* to the VCP and CDN DNS nameservers respectively. The VCP DNS server has an A record⁷ entry to redirect DNS requests to the CDN DNS. On the other hand, the CDN DNS server was initially configured with an A record value to redirect the CDN end-user requests to the CDN SS-P surrogate server. However, as the number of user requests kept increasing and the CDN SS-P server was over-loaded, the CDN DNS server A record entry was modified to redirect those requests to CDN SS-S. This VNF is capable of scaling out automatically i.e., instantiating more instances of the Apache VDU (indicated as CDN SS-S - Apache in Table 3) as the number of requests increases.

In order to test that the DNS request and redirection function was working, the following steps were taken:

- 1) Initially, we configured the CDN DNS server with CDN SS-P as the surrogate server to service end-user requests.
- 2) Next, we sent up to 20000 *http* GET requests sequentially from the UE to the CDN service with the domain name *server.cp.cdn.com*. The UE is emulated as a VNF with both Ubuntu 16.04 server edition operating system and Apache Bench [30] tool installed. This UE is located in “5TONIC-site1”, in Madrid, Spain, as indicated in Fig. 6. This choice of using a VNF as the UE instead of a physical mobile device was motivated by (i) flexibility in terms of management and configuration and (ii) superior performance due to the lack of limitations imposed by network access technologies, among others.
- 3) Concurrently, we started capturing the *http* traffic that was traversing the UE.
- 4) After approximately 60 seconds, we modified the CDN DNS server with CDN SS-S as the surrogate server this time around and reloaded the new DNS configuration.

⁶For the sake of simplicity, in our experiments, the DNS redirection is done manually. However, this can be done automatically as in current CDN deployments.

⁷An A record is the most basic type of DNS record and is used to map a domain name to an IP address

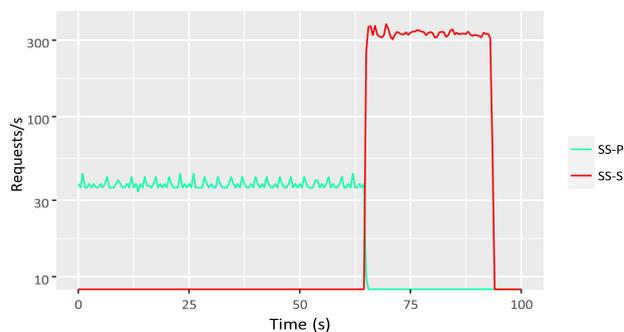


FIGURE 9. CDN DNS request and resolution functionality.

- 5) In another terminal on the UE, we flushed the DNS cache in order to force the UE to do a new DNS resolution.
- 6) Henceforth, we continued to run the experiment with the CDN SS-S as the surrogate server for another approximately 30 seconds.

At the end of the experiment, we plotted the captured *http* request (GET) traffic generated by the UE (in semi-log scale) from step 3 above against the experiment time and the results are shown in Fig. 9.

On the one hand, from Fig. 9, we can see that the rate, i.e., requests/second, handled by CDN SS-P is around 40 requests/second and after approximately 63 seconds, the rate now handled by CDN SS-S increases (i.e., above 300 requests/second). The rationale behind this increase is due to the mechanisms implemented by the transmission control protocol (TCP), where low round-trip times imply high bandwidth and vice versa. In our experiments, the requests of CDN SS-P experience much more delay than the requests handled by CDN SS-S. This is due to CDN SS-P being instantiated in a data-center (IT-Aveiro, Portugal) that is distant from the UE, whereas CDN SS-S is located much closer to the UE (i.e., both are located in 5TONIC, Spain).

These first results show that all services deployed using Network Function Virtualization tools can be managed using traditional protocols. In the next experiments, we will show how auto-scaling improves these benefits.

2) TESTING THE AUTO-SCALING FUNCTIONALITY OF THE CDN SERVICE

In this subsection, we tested our deployment of the ETSI OSM autoscaling, functionality explained in section III-B1 above inside the CDN service. In order to test this functionality, we performed the following steps:

- In IV-B2, we developed the CDN SS-S VNF as a composition of two VDUs, i.e., apache webserver and load balancer VDUs, reflecting this composition in the VNF descriptor. For the load balancer VDU, we used HAProxy as the load balancing scheme of choice [31].
- Inside the CDN SS-S VNF descriptor, we configured the apache webserver VDU to scale automatically

TABLE 4. CDN SS-S scaling policy.

Scaling parameter	Value
min-instances	0
max-instances	5
scaling-type	automatic
monitored parameter	CPU utilisation
scale-out threshold	70%
scale-in threshold	10%
threshold time	10 seconds
cool-down time	180 seconds

according to the scaling policy represented in Table 4. The min-instances and max-instances parameters represent the minimum and maximum number of instances to provision when performing scaling in and out operations. Setting the scaling-type parameter to “automatic” triggers the scaling operations to start automatically as long as the threshold values have been met. Inside the VIM, the CPU utilization parameter is continually being monitored; on the one hand, once the CPU utilization value is greater than or equal to 70% (i.e., scale-out threshold) for more than 10 seconds (i.e., threshold time), then the scale-out operations are triggered until the maximum number of instances (in this case 5) has been reached, however, the ETSI OSM platform has to wait for 180 seconds (i.e., cool-down time declared in Table 4) each time before instantiating a new instance. On the other hand, once the CPU utilization has been less than or equal than 10% (i.e., scale-in threshold) for more than 10 seconds (i.e., threshold time), then the scaling in operations are triggered until the minimum number of instances (in this case 0) are reached. In this case, too, the ETSI OSM platform has to wait for 180 seconds (i.e., cool-down time) each time before removing a scaled instance.

- Accordingly, we configured the monitoring parameters of the CDN SS-S VNF descriptor. These parameters were set to monitor the CPU load of the Apache web-server VDU.
- Finally, we stress-tested the CDN SS-S server using the Apache Bench tool, by emulating 100 concurrent UE clients connecting to the CDN SS-S server, and sending 5.0×10^6 requests in total.

The results of this test are shown in Fig. 10. From Fig. 10, we can observe a continual decrease in the delay. This is due to the autoscaling function implemented inside the CDN SS-S server. To better analyze the impact of autoscaling, we decided to plot the load distribution of the incoming *http* requests among the web servers by the HAProxy, representing the rate per server against time, which is shown Fig. 11. Using Fig. 10 and Fig. 11, we observed the following:

- 1) For the first 320 seconds, we have only one instance of the webserver running that is handling all the end-user

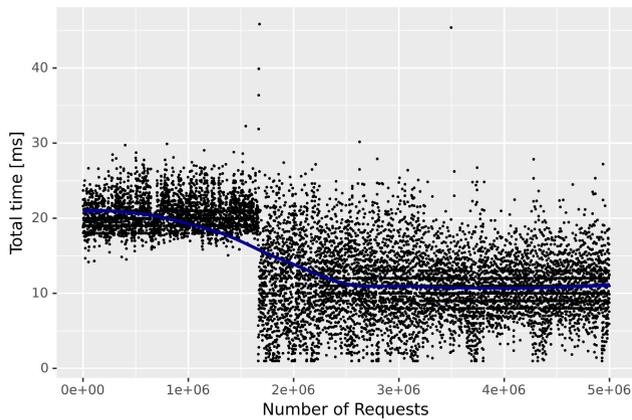


FIGURE 10. CDN SS-S UE experienced delay with autoscaling.

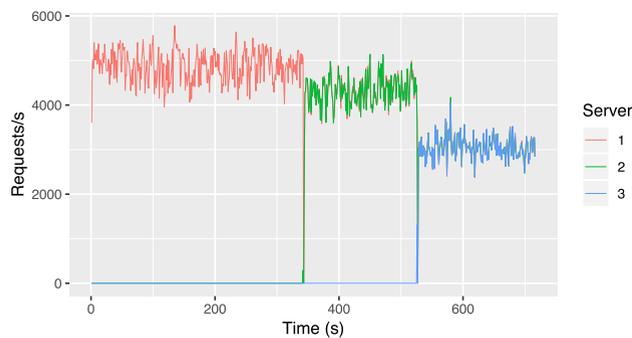


FIGURE 11. CDN SS-S: HAProxy vdu rate with time.

requests. The requests in this phase experience an average delay of 20 ms, with a standard deviation of 3 seconds.

- 2) For the next 190 seconds, the apache VDU inside the CDN SS-S web server has scaled-out to two instances, and now we have two instances of the apache VDU running with the HAProxy perfectly balancing the load between the two instances. Henceforth, we notice a further decrease in the delay with the samples experiencing an average delay of 12 ms, with a standard deviation of 6 seconds.
- 3) For the last 200 seconds, we have three instances of the webserver running. Hereto, the load balancer balances the load perfectly between all three instances, and the result is a much abated average delay of 11 ms with a standard deviation of 4 seconds.

The merits of performing scaling operations using the “Auto-Scaling” feature of ETSI OSM can be summarized as follows: it enhances the ability of VNFs to quickly adapt to varying traffic demands by instantiating new VDU instances of the VNF when monitored parameters go beyond a set maximum threshold (i.e., scale-out threshold); removing these scaled instances when the monitored parameters go below the minimum threshold (i.e., scale-in threshold). ETSI OSM platform is capable of performing these scaling operations automatically without need of manual intervention by

continuously monitoring the VIM and VNF metrics set in the VNF descriptors.

In our experiment, by configuring the CDN SS-S VNF to utilize the autoscaling feature, the VNF was able to serve a higher number of requests, with a much lesser delay. Care to note, for this experiment, the average delay experienced by the UE was 14.258ms. It is worth mentioning that, even the quality of experience perceived by end-users with no autoscaling is really good in our experiments, with delays in the order of ms [32], autoscaling would have a great impact improving the quality of experience in services with thousands of users.

E. SERVICE TERMINATION

This phase involved the termination of the CDN-NSlice and the VCP N-Slice network slices of the CDN management service. The service termination was carried out in two stages:

- Initially, the CDN-NSlice was terminated, which translated to termination of the CDN-NServ-S and CDN-NServ-P network services as well as the deletion of the associated VNFs and networks. However, the Op-NServ network service was not terminated since it is shared with the VCP-NSlice network slice. The CDN-NServ-S termination request was sent to the 5TONIC laboratory in Spain, whereas the termination request for the CDN-NServ-P was sent to the IT-Aveiro datacenter in Portugal.
- Secondly, after the CDN-NSlice was terminated, accordingly, we sent a request to terminate the VCP-NSlice. This request was translated to the termination of the Op-NServ and VCP-NServ network services, as well as the deletion of affiliated VNFs and networks. Both network service termination requests were sent to the 5TONIC laboratory in Spain.

Once terminated, all the resources (i.e., network, compute, and storage) that were initially dedicated to our services are now freely available to be used by other services.

V. RESULTS AND DISCUSSION

In this section, we present a qualitative and quantitative analysis of how much effort is required and the responsible parties for each of the phases of the CDN management service creation times. In our analysis, we consider a telecommunications operator with already existing data centers. OpenStack controls each data center, and ETSI OSM is in charge of service orchestration and VNFs management across the entire infrastructure. After analyzing the obtained results, we discuss the viability to achieve the envisioned service creation time KPI.

A. RESULTS

1) PHASE 0: PLATFORM PROVISION

Since this phase involves the configuration of the existing telecommunications network infrastructure to foster the new service, this phase requires: (i) infrastructure resource

reservation in terms of networks, compute and storage for the new service, and (ii) in case the service providers already have their own customized VNF images, then these images are validated and later uploaded to the infrastructure platform by the OpenStack system administrators. However, if the service providers do not have any customized VNF images, then these images are created in the next phase with the help of both the ETSI OSM and OpenStack system administrators.

Once the infrastructure configuration is done, the platform is ready to be available for the new service. It suffices to say, the amount of time taken in this phase is highly dependent on the new service requirements. However, for most services, the already provisioned platform is sufficient to service all end-user requests with minor adjustments. This was the case for the analysis performed in this article, where we used an existing NFV platform that was built to support experimentation activities in the context of the European H2020 5G EVE project.

2) PHASE 1: ON-BOARDING OF DESCRIPTORS

Considering that this phase encompasses the onboarding of previously generated descriptor packages to the ETSI OSM platform by the developer(s) and optionally uploading of the VNF images to the OpenStack image service by the OpenStack system administrator, this stage can take a few minutes to complete.

3) PHASE 2: INSTANTIATION, CONFIGURATION AND ACTIVATION

Taking into account that this phase involves the instantiation, configuration and activation of the constituent Network slices and in turn the corresponding network services and VNFs of the service, the amount of time taken in this phase will depend on (i) the number of VNFs, Network Services and Network slices that have to be instantiated and configured, (ii) the order in which the network slices and in turn the network services and VNFs are instantiated and configured; and additionally their data center location, and (iii) the amount of computing (i.e., memory, storage, and CPU cores) and network resources assigned to the service by the network operator.

In this article, as already presented in subsection IV-C, we considered two scenarios when it came to instantiating, configuring, and activating our CDN management service. On the one hand, considering scenario #1, the time taken for this phase was approximately 7 minutes, as illustrated in Fig. 7. On the other hand, when we instantiated, configured, and activated our service using scenario #2, the amount of time taken was less than the time taken in scenario #1, i.e., approximately 4 minutes as displayed in Fig. 8. This difference was solely due to two factors: (i) the order in which the network slices and in turn the network services and VNFs were instantiated and configured, and (ii) the data centers where this instantiation and configuration occurred. The former factor cannot be decided by telecommunications operators, as this depends on the order the different service

providers prefer to instantiate and activate their network services and slices (for example, a CDN provider might decide to activate their network slice today, and a month later, a VCP provider decides to activate their slice, or vice-versa); the latter factor can be reduced by improving the telecommunications operator infrastructure.

4) PHASE 3: MODIFICATION OF THE SERVICE

After the service is activated and handling traffic from the above phase (see subsection V-A3), then the performance metrics of the activated network slices and network services are continuously being monitored for possible modification. This modification could involve: changing the network slice and/or the network service configuration(s), manual or automatic triggering of scaling operations, and modifying some elements of the telecommunications network infrastructure.

In our particular service, as already discussed in subsection IV-D, this phase entailed: performing manual DNS redirection actions and triggering automatic scaling in/out operations in one of the constituent VNFs of the service. To manually configure the DNS service to redirect the requests to the proper surrogate server, takes a few minutes. However, if this DNS redirection is done automatically as it is currently the case, then this DNS redirection can be done in the order of *milli-seconds*.

On the other hand, since the scaling operations in our service were set to automatically trigger whenever the performance metric (in this case, the CPU load) went above or below the set threshold for more than 10 seconds, then the scaling operations were triggered immediately. However, after triggering a scaling operation (i.e., in/out), the ETSI OSM platform was configured to wait for 3 minutes and thereafter, re-evaluate the monitored performance metric before triggering the next scaling operation. This waiting time has to be carefully selected depending on the dynamics of the service under consideration. We have selected 3 minutes for experimentation purposes only.

5) PHASE 4: SERVICE TERMINATION

Given that this phase comprises: (i) Termination of the service provider dedicated network slices and in turn affiliated network services and VNFs; plus associated networks, and (ii) Re-configuration of the shared network services with other network slices (in case the terminated network slice was sharing network services with other network slices). Once complete, the telecommunication operator reclaims back all the resources, i.e., compute, storage, and network that were initially allocated to this service provider. Therefore, the amount of time taken for this phase depends on: (i) the number of dedicated network slices and in turn network services, VNFs as well as networks that have to be terminated, and (ii) the capabilities and location of the data center; where the termination request will be sent and finally, on the number of shared network services that have to be re-configured as a result.

TABLE 5. Experiments results - Service creation time KPI.

Phase	Phase Components	Time taken (minutes)
Phase0: Platform provision	Platform configuration	Not Applicable
	Platform deployment	
Phase1: On-boarding of descriptors	VNF packages	4
	Network Service packages	2
	Network Slice templates	1
Phase2: Instantiation, configuration and activation	Scenario #1	7
	Scenario #2	4
Phase3: Modification of the Service	DNS redirection	3
	Autoscaling	0.2
Phase4: Service termination	Termination of CDN-NSlice	4
	Termination of VCP-NSlice	2

For this article, as already presented in subsection IV-E, the termination of the service was done in two stages, i.e., Initially, the CDN-NSlice was terminated, and sequentially, the VCP-NSlice was also terminated. The termination of the CDN-NSlice took close to 4 minutes, whereas the termination of the VCP-NSlice took close to 2 minutes. The difference in the termination times is because the CDN-NSlice contains a network service (i.e., CDN-NServ-P) whose data center is located in Portugal, so that termination request experiences a bit more delay compared to the rest of the network services; whose termination requests were sent to data centers located inside the 5TONIC laboratory.

A summary of the achieved average results in regards to our CDN service is presented in Table 5, considering the different components of the service creation time KPI phases. As commented in previous sections, our analysis of the service creation time phases has been done over an existing NFV platform, built in the context of the European H2020 5G EVE project. Hence, phase 0 (platform provision) is not applicable in the provision and management of the CDN service.

B. DISCUSSION

To analyze the viability of reducing the service creation time from 90 hours to 90 minutes, as proposed by the 5G-PPP, we first have to understand the constraints and starting point of a new service. Because of the lack of a formal definition of service creation time, we have used the report done from several European projects, dissecting this time in several phases [27]. All these different projects have analyzed the service creation lifecycle from different angles, putting together their results in various project deliverables as discussed in section II.

On the one hand, if we consider a scenario where the service creation time besides all five phases, includes service design and preparation as was the case in our experiment, then reducing this time to 90 minutes would be more

than challenging. On the other hand, if we consider a scenario where the platform is already provisioned, which is a reasonable assumption in our point of view; the service descriptors are already designed and prepared, or at least there are pre-defined building blocks to facilitate its composition, then it would be more than feasible to achieve this service creation time in 90 minutes or less. Furthermore, initiatives like the H2020 5G EVE project have defined and implemented different tools to facilitate the implementation of all phases involved in the service creation time [33], [34].

VI. CONCLUSION

In this article, we have presented a comprehensive analysis of the 5G service creation time KPI phases by utilizing a complex network service that leverages on NFV technologies. As earlier mentioned in section I, this time is a Key Performance Indicator in 5G networks, and the goal is to reduce current times from 90 hours to 90 minutes. Henceforth, we have defined and conducted several experiments to dissect the time taken by each phase of the service creation time KPI. From our results, we can conclude that, on the one hand, some of these phases (for example, platform provision) require a significant amount of time to be completed, as well as proper network planning and dimensioning by the telecommunication network operators. On the other hand, NFV tools may have an outstanding impact on reducing the time to create a service in other phases, such as descriptors onboarding, instantiation, configuration, and management, as well as service modification. Therefore, in a scenario where the infrastructure is already provisioned with adequate resources, and the VNF and descriptors preparations are not considered part of the service creation KPIs (as is currently the case), it is possible to reduce the service creation time to 90 minutes or less. These short service creation times will attract new services to 5G, which will strengthen the position of telecommunications providers, and all 5G stakeholders in general.

In future works, we plan to tackle some of the challenges detected in this study. For example, intent-based networking and machine learning may be useful in the service design, instantiation, configuration and activation as well as modification phases. Furthermore, we plan to extend ETSI OSM to include the functionality to update a running network slice and avoid downtimes when a service has to be modified.

ACKNOWLEDGMENT

The paper solely reflects the views of the authors. The Commission is not responsible for the contents of this article or any use made thereof. The authors would like to thank Jorge Oliveira for his support during the realization of this work.

REFERENCES

- [1] *Technical Specification Group Radio Access Network; Study on Scenarios and Requirements for Next Generation Access Technologies; (Release 16)*, document TR 38.913, 3GPP, Jul. 2020.

- [2] *Description of Network Slicing Concept*, document NGMN 5GP, 2016, pp. 1–11.
- [3] *Virtual Network Function Architectural Framework*, ETSI, Paris, France, Nov. 2014.
- [4] L. M. Contreras, P. Doolan, H. Lønsethagen, and D. R. López, “Operational, organizational and business challenges for network operators in the context of SDN and NFV,” *Comput. Netw.*, vol. 92, pp. 211–217, Dec. 2015.
- [5] (Jul. 2020). *5G EVE: 5G European Validation Platform for Extensive Trials*. [Online]. Available: <https://www.5g-eve.eu/>
- [6] 5G-TRANSFORMER. (2019). *D5.4. Report Trials Results*. [Online]. Available: http://5g-transformer.eu/wp-content/uploads/2019/11/D5.4-5G-TRANSFORMER_Report_on_trials_results.pdf
- [7] 5G-TRANSFORMER.(2019). *D1.4. Final System design Techno-Economic Analysis*. [Online]. Available: http://5g-transformer.eu/wp-content/uploads/2019/11/D1.4_5G-TRANSFORMER_final_system_design_and_Techno-Economic_analysis.pdf
- [8] MATILDA. (2020). *D6.7. Validation Results, Performance Evaluation and Adoption Guidelines—First Demonstration and Evaluation Phase*. [Online]. Available: <https://private.matilda-5g.eu/documents/PublicDownload/436>
- [9] (2017). *D1.1. Matilda Framework and Reference Architecture*. [Online]. Available: <https://private.matilda-5g.eu/documents/PublicDownload/119>
- [10] 5G-MoNArch. (2019). *D5.2. Final Reports Testbed Activities Experience Evaluation*. [Online]. Available: http://5g-monarch.eu/wp-content/uploads/2019/07/5G-MoNArch_761445_D5.2_Final_report_on_testbed_activities_and_experimental_evaluation_v1.0.pdf
- [11] 5G-MoNArch. (2019). *D2.3. Final Overall Architecture*. [Online]. Available: https://5g-monarch.eu/wp-content/uploads/2019/05/5G-MoNArch_761445_D2.3_Final_overall_architecture_v1.0.pdf
- [12] 5GTANGO. (2020). *D7.3. Final Demonstrators Evaluation Report*. [Online]. Available: https://www.5gtango.eu/documents/D73_v1.pdf
- [13] 5GTANGO. (2019). *D5.2. Service Platform Final Release*. [Online]. Available: https://www.5gtango.eu/documents/D52_v1.pdf
- [14] L. U. Khan, I. Yaqoob, N. H. Tran, Z. Han, and C. S. Hong, “Network slicing: Recent advances, taxonomy, requirements, and open research challenges,” *IEEE Access*, vol. 8, pp. 36009–36028, 2020.
- [15] *Technical Specification Group Services and System Aspects; Telecommunications management; Study on management and orchestration of network slicing for next generation network;(Release 15)*, document TR 28.801, 3GPP, Jan. 2018.
- [16] *Network Functions Virtualisation (NFV); Architectural Framework*, ETSI, Paris, France, Dec. 2014.
- [17] OSM. *Open Source MANO: ETSI-Hosted Project to Develop MANO Software Stack Aligned With ETSI NFV*. Accessed: Jun. 1, 2020. [Online]. Available: <https://osm.etsi.org/>
- [18] OpenStack. *OpenStack—Open Source Software for Creating Private Public Clouds*. Accessed: Jun. 1, 2020. [Online]. Available: <https://www.openstack.org/>
- [19] *VMWare Cloud Director—The Leading Cloud Services platform for Cloud Provider*. Accessed: Jun. 1, 2020. [Online]. Available: <https://www.vmware.com/products/cloud-director.html>
- [20] *Amazon Web Services—Cloud Comput. Services*. Accessed: Jun. 1, 2020. [Online]. Available: <https://aws.amazon.com/>
- [21] Canonical. *Juju, Simplest Way to Deploy Maintain Application Cloud*. Accessed: Jun. 1, 2020. [Online]. Available: <https://juju.is>
- [22] *Prometheus—From Metrics to Insight; Power Your Metrics Alerting With a Leading Open-Source Monitoring Solution*. Accessed: Jun. 1, 2020. [Online]. Available: <https://prometheus.io/>
- [23] *Kafka—A Distributed Streaming Platform*. Accessed: Jun. 1, 2020. [Online]. Available: <https://kafka.apache.org/>
- [24] *Gnocchi—Metric as a Service*. Accessed: Jun. 1, 2020. [Online]. Available: <https://gnocchi.xyz/>
- [25] I. Vidal, I. Soto, A. Banchs, J. Garcia-Reinoso, I. Lozano, and G. Camarillo, *Multimedia Networking Technologies, Protocols, and Architectures*. Norwood, MA, USA: Artech House, 2019.
- [26] P. Mockapetris, *Domain Names—Concepts and Facilities*, document RFC 1034, Nov. 1987.
- [27] *5G PPP Phase-II Projects Performance KPIs*, document 5G-PPP, 2019. [Online]. Available: <https://bscw.5g-ppp.eu/pub/bscw.cgi/312793>
- [28] B. Nogales, I. Vidal, D. R. Lopez, J. Rodriguez, J. Garcia-Reinoso, and A. Azcorra, “Design and deployment of an open management and orchestration platform for multi-site NFV experimentation,” *IEEE Commun. Mag.*, vol. 57, no. 1, pp. 20–27, Jan. 2019.
- [29] Traffic. (Jul. 2020). *A Traffic Mix Generator Based Iperf3*. [Online]. Available: <https://github.com/mami-project/traffic>
- [30] ApacheBench. (Jan. 2020). *AB—Apache*. [Online]. Available: <https://httpd.apache.org/docs/2.4/programs/ab.html>
- [31] HAProxy. *HAProxy—The Reliable, High Performance TCP/HTTP Load Balancer*. Accessed: Jun. 1, 2020. [Online]. Available: <http://www.haproxy.org/>
- [32] T. Hobfeld, F. Metzger, and D. Rossi, “Speed index: Relating the industrial standard for user perceived Web performance to Web QoE,” in *Proc. 10th Int. Conf. Qual. Multimedia Exper. (QoMEX)*, May 2018, pp. 1–6.
- [33] (2019). *D4.1. Experimentation tools VNF Repository*. [Online]. Available: <https://doi.org/10.5281/zenodo.3628201>
- [34] (2019). *D4.2. 1st version Experimentation Portal Service Handbook*. [Online]. Available: <https://doi.org/10.5281/zenodo.3628316>



WINNIE NAKIMULI (Graduate Student Member, IEEE) received the B.S. degree in telecommunications engineering from Makerere University, Kampala, Uganda, in 2011, and the M.Sc. degree in telecommunications engineering from the University of Trento, Italy, in 2014. She is currently pursuing the Ph.D. degree in telematics engineering with the University Carlos III of Madrid, Spain. She is involved in the EU H2020 5G-EVE Research Project. Her research interests include 5G network slicing, network functions virtualization (NFV), software defined networking (SDN), and machine learning.



JAIME GARCIA-REINOSO (Member, IEEE) received the degree in telecommunications engineering from the University of Vigo, Spain, in 2000, and the Ph.D. degree in telecommunications from the University Carlos III of Madrid, Spain, in 2003. He joined the University Carlos III of Madrid as a Visiting Professor, in 2002. He has published over 60 articles in the field of broadband computer networks in top magazines and conferences. He has been involved in several international and national projects related with broadband access, peer-to-peer overlays, next generation networks, signalling protocols, information-centric networking, 5G, SDN, and NFV like the EU IST MUSE, Trilogy 2, the EU H2020 5G-Crosshaul, EU H2020 5G-Transformer, 5G-EVE, BioGridNet, MEDIANET, and MASSES, leading the last two projects. He was involved in several 5G-PPP working groups in the area of 5G like in spectrum, vision, and trials.



BORJA NOGALES received the bachelor's degree in telecommunication technologies engineering from the University Carlos III of Madrid (UC3M), in 2016, where he is currently pursuing the Ph.D. degree in telematics engineering. He is involved in the European research project 5Gin-FIRE and in the national project 5GCity. His research interests include network functions virtualization (NFV), 5G networking, and unmanned aerial vehicles (UAVs).



IVAN VIDAL (Member, IEEE) received the degree in telecommunication engineering from the University of Vigo, in 2001, and the Ph.D. degree in telematics engineering from the University Carlos III of Madrid, in 2008. He is currently working as a Visiting Professor with the Universidad Carlos III de Madrid. He has been involved in several international and national research projects, including the EU IST MUSE, Trilogy, the H2020 5GINFIRE, MEDIANET, and 5GCITY. He has published more

than 50 scientific papers in several conferences and international journals, lately in the areas of UAVs and network functions virtualization. His research interests include unmanned aerial vehicles (UAVs), 5G networks, multimedia networking, and network security.



DIOGO GOMES (Member, IEEE) received the degree (Hons.) in computers and telematics engineering and the Ph.D. degree in resource optimization for broadcast networks from the University of Aveiro, in 2003 and 2009, respectively. He is currently a Professor Auxiliar with the University of Aveiro. In the last 17 years, he has participated in several EU funded projects, such as IST-Mobydick, IST-Daidalos, IST-Akogrimo, IST-C-MOBILE, ICT-C-Cast, ICT-Onelab2, ICT-

Medieval, H2020-5GinFIRE, and currently H2020-5Growth. He has conducted research on QoS, IP mobility, multicast/broadcast, service and application development, and network orchestration. He has always been deeply involved in the deployment of prototypes and demonstrators.



DIEGO LOPEZ joined Telefonica I+D as a Senior Technology Expert, in 2011. He is currently in charge of the technology exploration activities within the GCTIO Unit. Before joining Telefonica, he spent some years in the academic sector, dedicated to research on network services, and was appointed as a member of the High-Level Expert Group on Scientific Data Infrastructures by the European Commission. He is also focused on applied research applicable to network infrastruc-

tures, with a special emphasis on virtualization, data-enhanced management, new architectures, and security. He chairs the ETSI ISGs on Network Function Virtualization and Permissioned Distributed Ledgers. Apart from this, he is a more than acceptable Iberian ham carver, and extremely fond of seeking and enjoying comics, and good discussions on any (in)appropriate matter.

...