Introduction to the Network Virtual Deployment for NaaS

Gladys Diaz Salas

Towards a QoS-Aware Network Virtual Deployment for Network-as-a-Service

Gladys Diaz

L2TI - Sorbonne Paris Nord University- Villetaneuse, France

Abstract

Current digital environment meet a continuous evolved through a diversification of network access technologies (ADSL, WiFi, 3 / 4G, LiFi, Zigbee, etc.), but also through the deployment of new services (mobility, location, telemetric, etc.) and new uses cases (e.g. smart-parking, smart cities, M2M, ubiquitous games, etc.). While a few years ago, the services offered depended on the type of networks (e.g. voice for telecom networks, data for computer networks, and audio / video for broadcast networks), service providers must now to adapt and anticipate new consumption patterns (e.g. user-centric services) in their offer strategies. The challenge is therefore how to agilely deploy new services in this technological and evolutive context?

One of the main objectives of this paper is to identify in this new landscape the impact of the advantageous changes brought by virtualization over the network deployment process. We focus on the introduction of virtualization into the Network Deployment Process (called the VN Virtual Deployment-VNVD). VNVD takes into account the properties of flexibility, adaptability and dynamicity that are vital for the Network-as-a-Service (NaaS), where both SDN and NFV represent major building components in the conception of new network architectures.

Keywords: Network Virtualization, Network-as-a-Service, Network Virtual Deployment, Network Architecture, Virtual Network lifecycle, Virtual Network Operating System, SDN, NFV

1 Introduction

We are witnessing today a great evolution in the context of network architectures. Indeed, the new constraints (E2E availability, mobility, user centric, E2E latency, etc.) imposed by the various access technologies and innovative uses (M2M, IoT, Smart-cities, e-health ...) enforce to operators and service providers to rethink the design and deployment of network services. Traditional network architectures are evolving to meet theses constraints. However, a lot of work remains still to be done to reach with the expected scalability and customization from deploying new networks.

It became obvious to network operators and network providers in general that Virtualization, Cloud and Softwarization through the emergent paradigms of Software-Defined Networks (SDN) ONF [2012] D.Kreutz et al. [2015] and Network Functions Virtualization (NFV) Han et al. [2015] are promising to design simplified, easy to handle, network architectures.

With the arrival of Cloud Computing, the vision of "all is a service" is applied today at the application solutions. However, concerning the current infrastructures, the network is rarely offered as a service and it is considered as part of IT infrastructure and it is not provided "on demand", neither "personalized".

In this technological advancement towards agility and flexibility of network solutions, two potential facilitators are clearly identified: SDN and NFV. Indeed, by abstracting the physical network hardware, SDN offers a central interface to dynamically control the network elements, while isolating the constraints of interfaces and other hardware-related specific aspects et al. [2012]. This programmatic capacity allows network architectures to evolve to become more dynamic. Network applications and services are no longer tightly coupled to the physical network infrastructure. Different network operations can be automated thanks to network layers abstraction and to the programmatic access to network resources. Via network APIs operators and providers of network services be able to define and enforce new and different networking policies to meet new network requirements, and which could be applied by the SDN network controller. A complementary emerging paradigm is NFV. It consists of implementing, as software components, network functions that are performed today on dedicated hardware and placing them over standard hardware instead. Thus, NFV advocates virtualization of network nodes with specific functions to allow dynamic deployments.

Regarding Network Virtualization, SDN abstraction allows SDN-based Network Virtualization Bozakov and Papadimitriou [2012], which define logical network topologies through multiple tenants in a form of network paths, with a SDN controller centralizing the programming of network elements. We consider this view such as a basic set of hardware networking elements that process network flows. It is rather a Network Equipment virtualization. Concerning NFV, it includes virtualized network functions (IDS, firewalls, load balancer, etc.) installed on virtual machines over a virtualized infrastructure with commodity hardware. But, deploying these functions are today regarded according to the constraints of the virtual machine where these functions must be executed.

Indeed, to benefit from relying on virtualization and softwarization advances, network providers must not only master hardware virtualization and network functions virtualization but also fully comprehend Network Virtualization at the *Design* phase and as well as rethink Network Virtualization at *Deployment* phase in order to meet application changing requests. Integrating all these

levels of virtualization, associated to a strong capacity for automation and sufficient network monitoring information will enable the dynamic and efficient virtual network deployment based on ecosystem needs. We specially focus on the need for the abstraction of the Network Operating System, how the way to choose and adopt the most efficient network control and the most dynamic network management.

This paper aims to identify the impact of the advantageous changes brought by virtualization over the network deployment process. QoS being a key aspect to be integrated into the conception and deployment of new network services generation, we integrate our QoS criteria modeling into the description and management of virtual networks (VNs) such as the guiding thread that guides our reflexion.

Three main contributions are identified as the aim of this work:

1. First, we overview the different proposals concerning network virtualization. In particular, we want to explain their points of view regarding the Virtual Network concept and what virtualization objectives are focused. We explain how virtualization can be involved into control plane and network service definition.

2. Second, we describe our cloud-based considerations to define Network as a Service, based in a QoS model.

3. Third, we highlight the challenges and limitation of current solutions to provide a E2E network service delivery.

This work is organized as follows. Section II surveys network virtualization activities.We present in section III our point of view about the NaaS (Network as a Service) definition. Section IV introduces our proposition about the first service into NaaS architecture: the customized network service delivery. We introduce the virtualization into the life cycle of NaaS in section V. Then, we focus on virtual deployment process. Section VI presents a first implementation of NaaS proposal. Finally, we conclude by bringing out the strengths of our proposition and introduce our perspectives.

2 From Network Virtualization to NaaS Architecture

Virtualization allows the abstraction of logical functions from underlying physical resources. This facilitates the adaptability and dynamicity of solutions. Adaptability is the ability to fit to service changes by adapting the resources involved. Dynamicity refers to ability to instantiate, on-the-fly, new services. Cloud Networking was defined as a way to deal these requirements in Cloud environments. However, CloudNetworking is based on virtualize physical networking resources.

We think that a second level of virtualization to build VNs that support, and dynamically adapt, the application flows' constraints on the network must be introduced. We analyse the research works that provide comprehensive studies on network virtualization (2.1). Indeed, isolated connectivity is not enough, VNs must also bear the characteristics of application flows behavior (2.2). This need has led to the concept of NaaS, that we present in section 3.

2.1 Today's Network Virtualization

Network Virtualization stands today as the paradigm for introducing the required network architecture changes to move towards the next generation of networks. Network virtualization allows different actors to share the existing physical network. By introducing a plurality of heterogeneous networks with architectures that coexist on a common physical infrastructure, network virtualization promotes the introduction of innovative and diversified applications, enabling a continuous evolution of the network and its architecture.

A Virtual Network (VN) is often defined in the literature as a collection of virtual nodes and links on a subset of the underlying physical network resources Chowdhury and Boutaba [2010]. Such a definition of network virtualization allows to decouple business roles between: *application service providers* that use network services, *network providers* that implement new offerings and deploy VNs, and *infrastructure providers* that provide and manage physical infrastructures. This decoupling defines a value chain that we keep in mind along our study.

Like IT, network operators have adopted virtualization. It applied mainly to networking nodes and to virtual links between these virtualized nodes. Different virtualization techniques with different objectifs was implemented by research projects. Figure **??** shows some of them. The reference Chowdhury and Boutaba [2010] presents a complet study on network virtualization.

Virtualization objectives		**Projects and research works**								
		Xbone	Violon	PlanetLab	GENI	4WARD	AGAVE	Frederic	Genesis	NetScript
Connectivity of virtual nodes		X	X	X						
Network Nodes virtualization					X	X				
Integration of federated environments					X	X		X		
Experimental support for virtualized environment					X	X		X		
Virtualization techniques	Tunneling		X		X					
	Overlay networks	X		X		X	X			
	Programmability aspects, active and programmable networks techniques								X	X
Virtualization of network level (layer 3 of OSI model)		X				X	X		X	X
IP QoS technologies: DiffServ, InterServ							X			

Figure 1: Some research works in the context of network virtualization

We can see how these different technologies have allowed the evolution of design and deployment of network services. We find all these techniques used nowadays for the implementation of a more recent concept in 5G: the network slicing. Slicing network allows multiple VNs to be created on top of a common shared physical infrastructure. The idea is customized the VNs to meet the specific requirements of services, devices, operators, etc. SDN and NFV will be put in contribution to allow traditional network architectures to be partitioned into virtual elements that can be linked (also through software). But slicing is based on a vertical viewpoint to the resources uncoupling and provisioning. Tenants are defined to provisioning specific and predefined capabilities to instantiate the VNs.

In our opinion, a horizontal point of view must be considered to introduce the personalization of on demanded VNs by NaaS. In this case, VNs must be capable to describe itself. NaaS must accomplish necessary composition of services according to requested. QoS description guide our point of view to introduce a way to describe VNs.

The figure 2 shows both viewpoints.

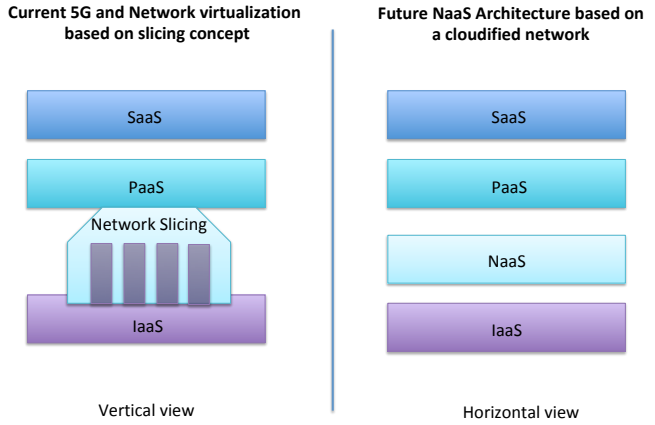

Figure 2: Vertical and Horizontal viewpoints for network virtualization

Other paradigms such as Cloud and Service-Oriented Architecture (SOA) Channabasavaiah et al. [2006] were also integrated in the definition of network virtualization Niebert et al. [2008]. A more completed study on Service-Oriented Network Virtualization is presented in Q. Duan [2012]. Service-Oriented network virtualization supports Cloud computing, SOA and network virtualization. It focuses on the convergence of networks and Clouds through the NaaS paradigm.

We think that all these aspects (Cloud computing, SOA and network virtualization) must converge to allow network virtualization becomes a component of NaaS. However, how does NaaS promotes network virtualization? How are: network virtualization, SDN, NFV and Could, integrated within NaaS? We define in the following sections the awaited characteristics and requirements of Network Virtualization within a NaaS architecture.

2.2 A Way Forward for the Network Service Virtualization

In a virtualization environment, a hypervisor manages the shared machine resources. Hypervisor acts like an abstraction layer to extract the OS of physical machine, so that each Virtual machine (VM) can be installed with its own OS independently from the host OS. Similarly, a Network OS (NOS) manages access to network infrastructure to deliver media between two endpoints **?**. The NOS programs routers, switches and links to establish end-to-end connection. The media delivery is performed in a classical network through the protocol stack defining the rules of network access, addressing, multiplexing, flow control, error control and routing. Therefore, we can conclude that for media delivery, the establishment of a communication session with a NOS is similar to creating a process by a classic OS to run its applications.

We, thus, draw our inspiration from OS virtualization in machines to virtualize the NOS in the network. In the same way as for creating multiple VMs isolated and containing different types of applications, we can create several virtual networks isolated, independent and customized. These VNs would be associated with a dedicated control plane Ayadi et al. [2013]. To continue the analogy, network resources would be shared thanks to a network hypervisor.

In other words, we believe that virtualization should not be limited to network equipment levels, whether these equipment were nodes (virtual routers, virtual switches,...) or links (vLAN, VPNs,...). Virtualization must include the network control plane associated to the control of each network flow delivered through the network.

Two strong network technologies enabling the realization of such an objective encouraged our analysis. First, SDN, with its abstracted network control plane allows this vision of NOS. Where the SDN controller holds all network intelligence and programs the sets of networking elements. Second, NFV, with its virtualized network functions, allows a network provider to operate network functions exactly as software applications. Thus, only the VNFs needed to treat the application service flows would be combined and dynamically deployed over a VN that is dedicated to an application or a user.

By analogy to virtualization of operating systems and resources in vitualization of physical machines, we consider the virtualization of NaaS network infrastructure (NIaaS).

In the context of NaaS, SDN and NFV are already considered as strong building blocks of NaaS architecture as presented in Diaz and Boussetta [2016], Diaz et al. [2017], Boubendir et al. [2016]. We analyse in the next section how our NaaS architecture can improve the definition of on demand and customized VNs.

3 Towards a Network-as-a-Service Architecture

We placed our work in Cloud environment with mechanism of virtualization of resources and the instantiation of services at IaaS and SaaS/PaaS levels. NaaS is a more recent approach that introduces the *Network as a Service* concept. We consider the introduction of Cloud paradigm to define the Network as a Service architecture. Figure 3 shows the levels introduced by considering a "cloudified environment" for the NaaS defintion.

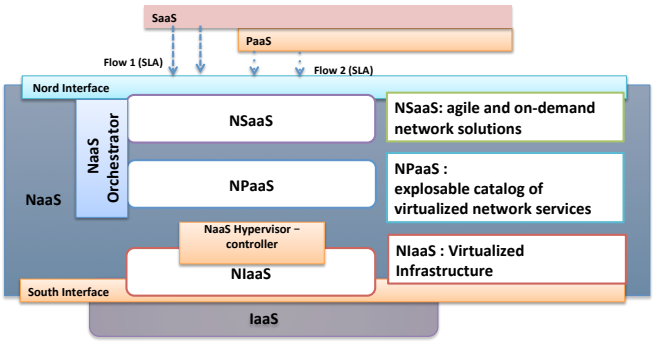

Figure 3: Cloud considerations for NaaS defintion

In our point of view NaaS (3.1) must be able to provide on demand and customized network services to deal

with flow requirements. We introduce a QoS aware approach (3.2) allows reflect an accord between the expected QoS (from SaaS / PaaS layers) in according with the offered QoS NIaaS (4). Our solution is based in a E2E virtualization, we discuss the VN continuous control in (4.2). Figure 4 shows a global view of components for NaaS architecture.

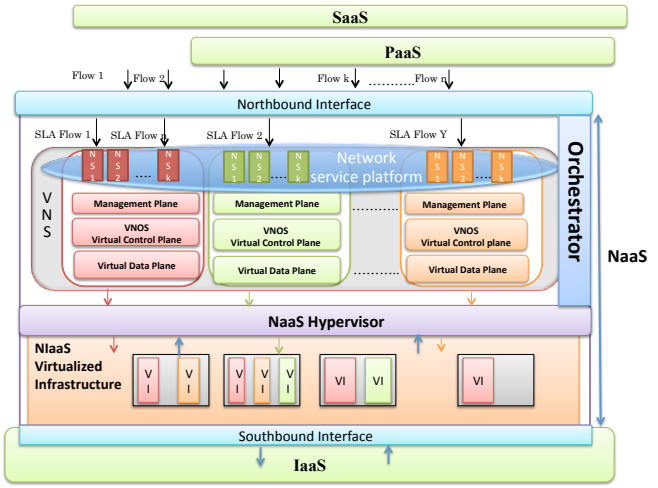

Figure 4: NaaS Architecture

3.1 NaaS objectives

- *Personalized Network treatment:* The first objective of NaaS is to make the appropriate treatments to different types of flows (application, signalization, inter-centers, etc.). In this fashion, PaaS address to NaaS a request to obtain the appropriate network service by type of flow. We consider that NaaS provides VNs by configuring nodes and links. A Virtualized control network plane (VNOS) is defined with each VN. VNOS takes constraints into account and establishes routing. VNOS must program the rules to be applying to guarantee the constrains of application flows. VNOS have two major objectives: be responsible of configuration and installing the control commands in network elements at deployment phase, and to collect status information about these same elements (network devices and links) at exploitation phase.

- *Description of E2E virtual networks:* The second objective of NaaS proposition is to describe the VNs to be deployed in according with both: expected flows treatments and IaaS offers (NIaaS). Thus, the offer of network services is characterized by a set of QoS and placement constraints, which NaaS must be able to meet when VN is deployed. For this objectif we consider a NaaS Hypervisor that, like the machine hypervisor, will play the role of NaaS manager at the network level. Through an abstraction layer of the underlying physical infrastructure, it is the responsible of chooses and describes the configuration of virtual resources involved in VN to be deployed according to flow requirements to be supported.

- *Maping the E2E solution:* The last step is mapping the VN in network infrastructure. NaaS hypervisor makes this placement.

3.2 The QoS aware approach

NaaS must be able to qualify its offers. We apply our QoS generic model to make this qualification and to describe the contracted QoS. In fact, NaaS must be able to consider together the PaaS requirements and the IaaS resources capabilities. The IaaS capabilities restrict the NaaS Capable-QoS to be considered to build the NaaS offers. The final contracted QoS NaaS represent the mapping between the request and the offer, both qualified with the same QoS model. This mapping take into account the placement and network constraints to be considered in the VN description.

Four QoS criteria are defined:

- Availability (A): is the portion of time that a network component can treat the requested service without failure.

- Reliability (R): represents the compliance rate of the provided service compared to the demanded one.

- Capacity (C): is the ability of a network component to fulfill the required service.

- Delay (D): is the total time taken by a network component to fulfill its functions.

Network components (ie. Nodes and links) are in this fashion characterized by theirs capability, availability, reliability and the delay to reach the requested flow treatment. These four QoS criteria are evaluated through three types of measurable values: conception, current and thresholds values (Figure 5).

	Type of values to QoS criteria		
	Conception values	***Threshold values***	***Current values***
Description	maximum capacity of the network component processing	limits values not to be exceeded by a network component in order to ensure a normal behavior	are used to monitor the behavior of the network component. These values would be compared with the threshold values to control the non violation of the component capacities.

Values of QoS criteria	Life cycle phases		
	Design	***Deployment***	***Operational***
	Conception values	Threshold values	Threshold values Current values

Figure 5: QoS Criteria applied to lifecycle of virtual networks

To ensure the E2E QoS requirements we spread our approach of QoS model at all phases of the life cycle of a service 5.1. Conception values are defined at "Design phase". Thresholds values are used at "Deployment" and "Operational" phases. Current values are maintained at operational phase.

4 A customized Network Service delivery

The role of NaaS is to determinate the most appropriate services to be consider and to configure the network components involved. By enforcing the virtualization at different levels NaaS supports two types of services: Network Service Delivery and Media Service Delivery.

The Network Service delivery is the E2E network service offered at NSaaS / NPaaS levels. The Service Delivery offered by NaaS concern with the selection of the adequate VNFs when building the VN and when selecting the control plane functions.

The Media Service Delivery is providing by the programmation of the control plan and the data plan when VN is deployed at NIaaS. The management plan is also programmed through monitoring components that monitor the behavior of both vertical and horizontal compositions.

4.1 Describing the Media Service Delivery

The request providing of SaaS / PaaS correspond to the transfer requirement to a type of flow. The role of NaaS is to program the most appropriate Media Service Delivery to deal with the flow QoS requested. VNOS comes to program rules which guaranteed the flow treatment and the respect of QoS constraints. Our approach aims to determine the requirements of different traffic flows, and, then to select or program the adequate control plane that meets these requirements.

By using our QoS model, the media delivery service can be described thought the definition of the following network services. These services must to be considered in each network node of a VN:

- Addressing (availability): is the probability to connect to a specific node. If the node can be addressed the service delivery is available. Addressing represents the possibility of a node to treat the flows arriving.
- Routing/commutation (delay): is the time of treatment of flow in a node. The first function in delivery service is the commutation of flow between two nodes. Routing service represent this treatment.
- Flow control (capacity): is the ability of node to respond to a request. Flow control enables to reflect the capacity of treatment of a node.
- Error control (reliability): is the compliance rate between the transmitted and received data. Error control measures the integrity of treatment of flows in a node.

In fact, to meet the QoS requested by the applications, we propose to combine network services with the QoS criteria. Thus, availability will depend on the addressability. Time will be respected according to routing choices and the number of nodes to be traversed. The capacity will induce a type of flow control. And the level of integrity will depend on the chosen error control. These different services can be implemented by different NFVs instantiated at virtual nodes.

4.2 A Continuos control for VNs deployment

Dynamicity in deployment of future VN will be a reality only by the knowledge about the status of shared resources. To give a continuous control of the network service we propose two complementary elements at NaaS: NaaS Hypervisor/controller and NaaS Orchestrator, see figure 6.

NaaS Hypervisor has the role of manager: the virtualization, sharing, and allocation of the virtualized infrastructure. It controls the NIaaS. NaaS Hypervisor must treat with the forwarding, distribution and specification of VN at deployment phase. Forwarding involved the possibility of programming the forwarding behavior desired by network applications, distribution concerning with the information about status of network elements, and specification involved the description of the desired network behavior by a network application. It maintains the abstract view of network resources.

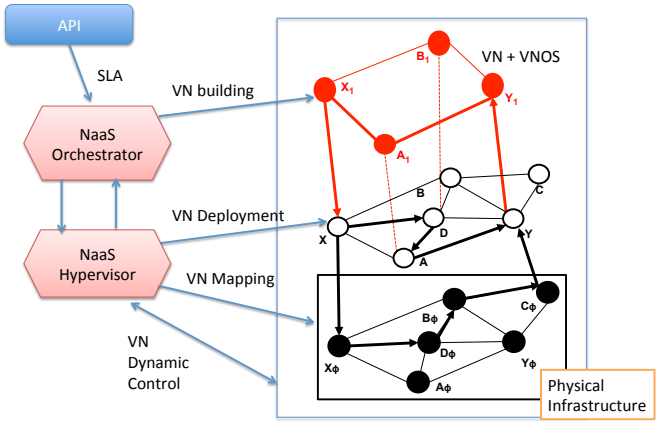

Figure 6: NaaS Orchestrator and NaaS Hypervisor

In other hand, the NaaS Orchestrator, which automates the continuous deployment of plans (control, data, and management) in the elements of the VN based on information gathered about the status of these elements during the operational phase. It represent the entity, which control and manages the knowledge needed to facilitate the programming of control, management and data planes in the VN building process.

NaaS Hypervisor and NaaS Orchestration must communicate to exchange information about network elements and to maintain a global view of resources offered by NIaaS.

In the same way, VNOS ensures the VN continuous control, and adaptation if the initial constraints are not respected. VNOS must communicate with the NaaS Hypervisor to report any break in expected behavior and any need for adaptation if the current use is changed. NaaS Hypervisor can rethinking the virtual deployment, or call to NaaS Orchestrator to re-building the VN if needed.

5 Virtualizing the Network Deployment process

Such as mention in the last sections, virtualization introduces changes into the lifecycle of NaaS services. Deployment process can takes advantage of this changes to introduce new building step and a continuous deployment of VNs.

By introducing virtualization at Network deployment phase, the customization of network service can be represented by two viewpoints:

- The service viewpoint: that is the *Network Service Delivery*, which is the E2E network service characterizing the NaaS offer.

- The placement viewpoint: that is the *Media Service Delivery*, which take into account the QoS and placement constraints when mapping the VN to the physical infrastructure.

Changes in network service deployment could be supported by both concepts. A *Network Service Delivery* change, resulting from SLA move, occurs by the changes of application use (user mobility for example). In this case, a re-thinking of VN must be envisaged, that is a new VN must be building to take into account the new SLA (VN re-building). This decision is taking by the Network Orchestrator after notification coming from Network Hypervisor and VNOS.

In another side, *Media Service Delivery* can change to be adopted the current network conditions (e.g., new QoS rules to apply flows aggregation) or to a QoS disruption. In this case, we speak of VN re-deploy, which is a possibility given by virtualization at Network deployment phase. Thus, the changes at *Media Service Delivery* level are carried out in order to maintain the *Network Service Delivery* behavior.

Thus, changes in *Network Service Delivery* (service viewpoint) are reflected by changes on the *Media Service Delivery* (placement viewpoints). We show in the next section how to describe these viewpoints at NaaS lifecycle.

5.1 Introducing virtualization into the lifecycle of NaaS

Deployment is an important phase of the network lifecycle. However, in the current ecosystem impacted by the massive adoption of virtualization the question posed is how modify/rethink the process of network deployment to introduce dynamicity? We answer to this question by decoupling the current phase of deployment in two distinct process: the virtual deployment of VN 5.1.2 and the mapping phases 5.1.3. We detail in the next sections the transformation involved into the lifecycle of VNs, in special the introduction of the virtual deployment phase 5.1.2. Figure 7 shows the phases described for lifecycle of VNs into the proposed NaaS.

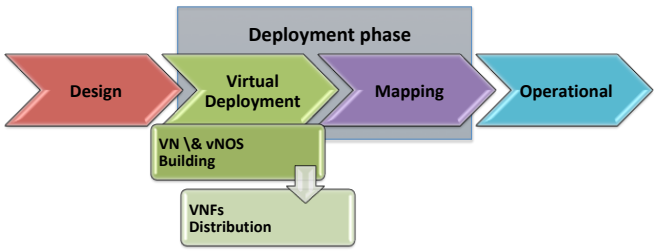

Figure 7: VN lifecycle at NaaS

5.1.1 The Design Phase

It is concerning the design of the new offers of network services at NaaS. In fonction of PaaS/SaaS requests NaaS built the most appropriate VN by making a choose on NFVs into NaaS-NFV catalogue (NPaaS). The result of building phase could be described today with the notion of NFV chaining, which represents a set on composable virtualized network functions.

We propose to use our QoS model to characterize the behavior of each NFV in the catalogue, and also the behavior of the expected E2E network service. This phase enables to define the NaaS SLA, which is the contract that describes the relation between the QoS requested and the QoS offered. Thus, this phase provides the E2E Service Delivery (Network Service Delivery), with an associated QoS, to provide network service between access points.

5.1.2 The Virtual Deployment of VN Phase

In this phase we are interested in expressing a translation of the needs and requirements coming from the application to characterize the behavior of the VN to be deployed, it is the service composition answering the demand, the logic of the network service to be offered.

This phase describes the Media Service Delivery, given by the VN description. We called this phase Network Virtual Deployment because it is a virtual deployment over NIaaS, the physical mapping being performed in a second time at Mapping phase (see 5.1.3).

Two process are considered into the *Virtual Network Virtual Deployment* (VNVD):

- First, the *VN & vNOS Building step:* This phase corresponds to the characterization of the virtual elements in accord with data flow to be processed, aiming at translating the requirements into a virtualization at the network service level. We obtain a virtual network whose network components contain the NFVs necessary for supporting the Network Service Delivery defined at Design phase.

  The Virtual Control Plan (VNOS) is programmed in this step. It defines, among other things, the constraints of the flows, the rules of routing (RIB-Rules Information Base) for the service of Media Delivery. These RIBs will induce the FIB (Forwarding Information Base) that will be placed in the data plan (NIaaS). The programmation of control plane concerns the choose of network functions (addressing, routing, control flow, error control) to be placed at data plane to control and guarantee the QoS requested.

- Second, the *VNFs Distribution step:* NaaS is concerning here by the installation of E2E network service defined at building phase, and how to configure the VN components that support it. It is about of how to distribute VN components and how to link them, by guaranteeing at the same time the E2E behavior described at precedent phase.

  VNOS is responsible for the configuration of different planes of network components in accordance with the QoS negotiated in the precedent phase. Thus the current resources selected must be deduced from the available resources at NIaaS. Hypervisor NaaS chooses the virtual resources (nodes and links) on the abstract view of the real infrastructure (based on the resources negotiated with IaaS). NaaS also incorporates the description / translation of the requirements for the placement of the various elements of the virtual network to satisfy the demands of requirements of the flows.

5.1.3 Mapping Phase

The VN builded at Virtual deployment phase is finally installed into the physical infrastructure (IaaS). The mapping is made by respecting the QoS and Placement constraints described before. This phase conditionnes the performance of the Media Service Delivery. Different proposition of mapping can be considered at this phase. We have characterized and proposed some first treatment for optimal network slices placement in Kammoun et al. [2018].

5.1.4 Operational Phase

In this phase, services are instantiated and running. The role of VNOS is to verify and control the VN resources available, and also the respect of QoS rules defined at virtual deployment phase. These considerations enable to maintain a correct behavior of each VN deployed. NaaS must also to implement the monitoring and management functions needed to enable the flexibility through the reconfiguration of VN (VN re-building) in the case of a disruption in the QoS (described by the SLA) or a change in use of the VN (e.g., caused by mobility). VNOS must inform NaaS Hypervisor in case of needed changes at Media Service Delivery.

To support an E2E QoS provision and management, lifecycle must represented complementary point of views (e.g, customer and provider). We associate each points of view with the QoS (requested/offered) into each phase of NaaS lifecycle. We have presented in Diaz et al. [2018] a QoS-aware model for the specification of non-functional constraints for virtual elements constitutives of VN. Figure 8 shows the application of QoS model into the proposed VN lifecycle at NaaS.

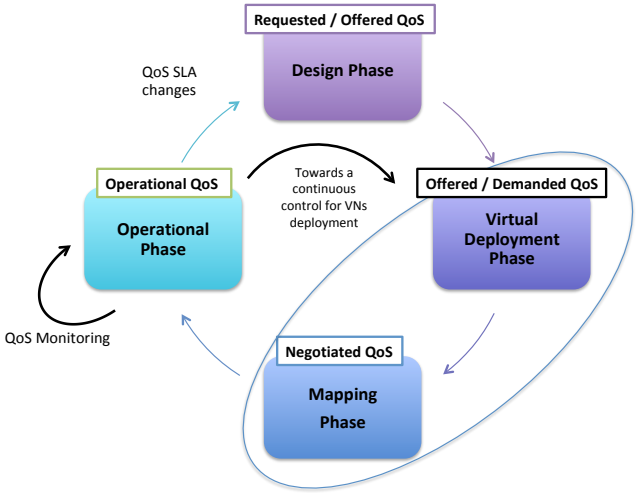

Figure 8: Introducing QoS model into NaaS lifecycle

5.2 Virtual Network description

To enable automatisation in deployment process, it is necessary to describe the complete information used by it. Thus, configuration of each network element must be described as well as the global view of the VN (service point of view). At operational phase, this information is used for management issues. The monitoring process can use these information to verify that current behavior of elements is in according with deployment configuration and that elements guarantee the QoS constraints described.

We have proposed in Diaz and Simoni [2016] an extension of OVF file to describe all the necessary information involved in the virtual network to be deployed by NaaS. We describe the elements involved in the Network section of OVF++ file. By apply our QoS approach, the VN is described by three parts: the virtual link section, the virtual node section and the QoS network constraints section. We include in our description the QoS and placement constraints for each level (E2E VN, virtual links and virtual nodes) such as mentioned in Ayadi et al. [2013].

Tables 1 and 2 show the QoS and placement constraints considering at node and link components.

6 Feasibility Study & Experimentation

We present in this section a study for the deployment feasibility of NaaS architecture. To achieve this implementation, various choices have to be made about the currently available open source tools and solutions. We study the different open source tools in section and present a discussion about the feasibility study on 6.1 . A first proposal of implementation is described in 6.2.

6.1 Feasibility Study Discussion

Different aspects must be studied before to met in place our solution. The difficult of this task consist in the choice of open technologies at different levels and their compatibility.

6.1.1 NaaS Layers

Our proposal architecture proposes different layers (whose must be implemented), which are the bottom to up as following:

- IaaS-Physical Infrastructure : We consider a SDN-based network. IaaS represents the programmable physical resources (the forwarding devices, the data plane from a SDN viewpoint). NaaS negotiates with IaaS provider(s) to built its virtualized infrastructure.
- Southbound interface : Defines a bridge to connect control and forwarding elements. Southbound interface enables the programing of commodities entities. A SDN protocol must to be defined at this level.
- Virtualized Infrastructure - NIaaS : this level represents the abstracted view of physical resources negotiated by NaaS (i.e. the network and compute capacities of NaaS). Virtual network deployment will be made at this level first.
- NaaS Hypervisor : it is the component responsible to manage the virtualization of resources to NaaS. In the same way that a hypervisor manages the deployment for virtual machines, a network hypervisor manages and optimize the network components deployment into the NIaaS.

Placement constraints	Description
Node isolation (v:set <VNE>)	All virtual nodes (VNEs) that belong to v group should not be colocated with existing VNEs
Node affinity (v:set<VNE>)	All virtual nodes (VNEs) that belong to v group should be mapped on the same physical node
Node anti-affinity (v:set<VNE>)	All virtual nodes (VNEs) that belong to v group should be mapped on different physical node

QoS node constraints	Description
Node availability (vl:VNE, α:rate)	Availability rate of v1 should be at least equal to α
Node capacity (vl:VNE, c:string, z:number)	Capacity of type c of node v1 should be at least equal to z
Node integrity (vl:VNE, β:rate)	Integrity rate of v1 should be at least equal to β
Node delay (vl:VNE, d:time)	Processing time of v1 should be kept below d time

Table 1: QoS and placement constraints at node level

Placement constraints	Description
lsoPath(vl:VNE, v2:VNE, nb:number)	Number of intermediate nodes between nodes v1 and v2 should be kept below nb
BanPath(vl:VNE, v2:VNE, s:set <INNode>)	Path between nodes v1 and v2 should not include intermediate nodes belonging to s

QoS link constraints	Description
LinkAvailability(vl:VNE, v2:VNE, α:rate)	Availability rate of link between virtual nodes v1 and v2 should be at least equal to α
LinkCapacity(vl:VNE, v2:VNE, t:string, v:numbrer)	Capacity of type t of link between virtual nodes v1 and v2 should be at least equal to v
LinkIntegrity(vl:VNE, v2:VNE, β:rate)	Integrity rate of link between virtual nodes v1 and v2 should be at least equal to β
MaxLinkDelay(vl:VNE, v2:VNE, d:number)	Delivery time between virtual nodes v1 and v2 should be kept below d time
LinkUtilisationRate(vl:VNE, v2:VNE, E:rate)	Utilization rate of link between virtual nodes v1 and v2 should be kept below E
LinkActivityRate(vl:VNE, v2:VNE, θ:rate)	Activity rate of link between virtual nodes v1 and v2 should be kept below θ

Table 2: QoS and placement constraints at link level

- NPaaS : It is responsible to maintain the set of network services and virtualized functions. Specific VNOS are defined by VN, a VNOS play the role of virtual controller to program the network components of each deployed VN.
- NaaS Orchestrator: It is a vertical layer that control and manages the continuous deployment of VN.
- NSaaS : It is represented the E2E network service offered by NaaS provider.
- Northbound interface: API enabling capture the application flow sensibilities' (SLA and QoS requirements).

6.1.2 Open source tools and solutions

To give an overview of today's ecosystem supporting the network services provision, we examine in this section several SDN open source tools supporting the virtualization of data plane (virtual switchs), SDN southbound interfaces protocols, SDN controllers and SDN Network abstraction.

We report some of the main products into the following tables:

Table 3 lists some SDN data plane software.

Table 4 lists and describes some examples of SDN control plane related software.

Table 5 lists and describes some SDN Network abstraction layers more precisely examples of Network hypervisors.

A more complete list about existing SDN tools and products is presented by Diaz et al. [2017].

6.2 Experimentation

A first implementation was performed on the Grid platform GRID5000 GRID5000. We describe in the next sections the components implemented for this feasibility study.

6.2.1 Architecture Components Implementation

We describe in this section the layers and components implemented and the choice made concerning the open sources tools.

- *Physical Infrastructure.* Computing and storage resources are provided by Grid5000 platform. Each of used physical nodes offers resources characteristics as recorded in Table 6.
- *The virtualized NaaS Infrastructure.* Such as was mentioned in last sections, we consider that the NaaS provides its resources across a virtualized infrastructure: the NIaaS layer. For implementation purposes, we have chosen to use Open VSwicth Version 2.3.1 Open-vSwitch to define virtual forwarding nodes (virtualized switches). Open VSwitch allows us to separately deploy virtualized nodes over different physical nodes, unlike Mininet that requires the installation of the entire network topology on a single virtual machine. This has allowed us also to have information specific of each virtualized network node deployed. Figure 9 illustrates the experimentation environment.

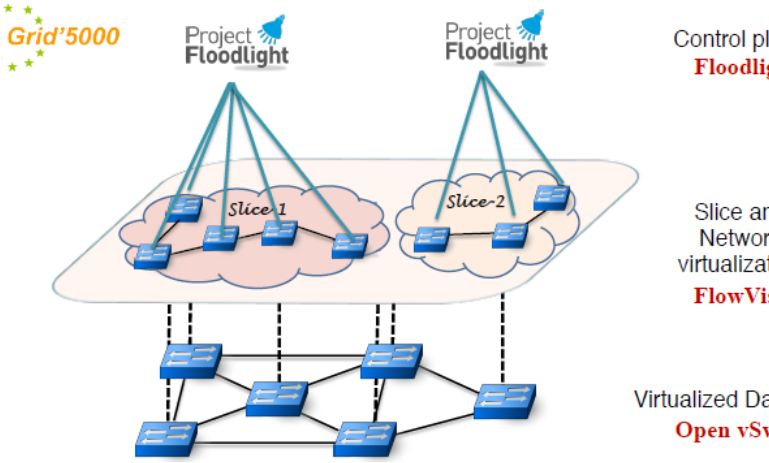

Figure 9: Experimentation Environment

- *SDN-based virtual network.* One of the objectives of our study is to allow multiple VNs to share the same NIaaS, where each VN would own a dedicated network control plane. For this purpose, we have used FlowVisor FlowVisor. It acts as the Network Hypervisor allowing the programming and isolation of virtual networks, each of one with their own controller.
- *Virtual networks' controllers.* Based on a preliminary study Shen [2014], on OpenFlow-enabled controllers, we have chosen to use the open source Floodlight controller Floodlight [2012] for each deployed slice. Other controllers are currently in study.
- *Southbound Interface.* It is implemented by using OpenFlow protocol McKeown et al. [2008]. OpenFlow supports the communication between Open-VSwitch (OVS) switches and FlowVisor.
- *Northbound Interface.* For this first experiments, formal descriptions for describing network flows, topology and network constraints to respect are manually programmed then enforced. To introduce dynamicity in the application (to VN description and deployment), we are currently integrating our OVF description.

6.2.2 Experimentation Objectives

Using our approach for building virtual networks based on "VN + VNOS", this first phase of experimentation aims at evaluating the performance of available open source tools and realize the needed connections between the various entities invoked in both control and data planes. As we described in precedent section, we study how changes at network media delivery (e.g. number of network nodes to be crossed) affecte the E2E network service delivery. We have studied the impacts of changes at the network nodes starting with the number of nodes to cross and their capacity.

SDN data plane	Description
OpenvSwitch (OVS) Open-vSwitch	It is a multilayer software switch over open source Apache 2 license. OvS supports standard management interfaces (e.g. NetFlow, sFlow, IPFIX, RSPAN, CLI, LACP, 802.1ag) and opens the forwarding functions to programmatic extension and control.
OpenSDNCoreSwitches (OSCS) OpenSDN Core	A software implementation of an extended Openflow 1.4 switch with specific oriented extensions as GRE tunnelling.
eXtensible Datapath daemon Switches (xDPd) xDPd	A multi-plateform, multi Openflow version, opensource datapath built focusing on performance and extensibility.
IndigoIndigo	It is an open source project enabling support for OpenFlow on physical and hypervisor switches.
Pantou Pantou	It is an OpenFlow port to the Open-WRT wireless environment..
Ofsoftswitch Ofsoftswitch	It is is an OpenFlow 1.3 compatible user-space software switch implementa- tion. The code is based on the Ericsson TrafficLab 1.1 softswitch implementation, with changes in the forwarding plane to support OpenFlow 1.3.

Table 3: Examples of SDN data plane softwares

SDN control plane	Description
Opendaylight (ODL)Opendaylight	Modular open platform for customizing and automating networks of any size and scale.
ONOSONOS: An open source distributed SDN OS	It provides scalability, hight perfor- mance, resiliency leagcy device support and next generation device support.
Floodlight Floodlight	An enterprise-class, Apache-licensed, java-based Openflow controller.
Ryu Ryu	Designed to increase the agility of the network by making it easy to manage and adapt how traffic is handled.
POX POX	Written in python and used for aca- demic research and studies.
BeaconErickson [2013]	A fast, cross-platform, modular, Java- based OpenFlow controller that sup- ports both event-based and threated operation.
Opencontrail Opencontrail	An Apache 2.0-licensed project that is built using standards-based protocols and provides all the necessary components for network virtualization-SDN controller, virtual router, analytics engine, and published northbound APIs.

Table 4: Examples of SDN control plane related software

Network Hypervisor	Description
Flowvisor – Flowvisor Controller	A special purpose Openflow controller that acts as a transparent proxy between Openflow switches and multiple openflow controllers. It creates rich slices of network resources and delegates each slice control to a different controller.
OpenVirteX (OVX) – OVX	A network hypervisor that can create multiple virtual and programmable networks on top of a single physical infrastructure.
xDpd Virtualisation – xDPd	A modified version of xDPd with a virtualisation module to allow multiple parallel experiments to be executed on the same physical infrastructure without interfering each other.

Table 5: Examples of SDN Network hypervisors

Machine type:	X86 64
Operating system (OS):	3.2.0.4 - amd64
CPU:	2 CPUs
CPU Speed:	2390 MHz
Memory Size:	2060472 KB

Table 6: GRID 5000 Platform - Physical Nodes Characteristics

6.3 Implementation

We consider two different applications generating two different flow types. One video traffic application generating User Datagram Protocol (UDP) flows and one VoIP application generating Real-time Transport Protocol (RTP) flows.

We have performed a first study to establish the network service delivery. We have then realized the virtual deployment phase taking into account the type of application and therefore the type of its network flow that will be generated (network service delivery). Next, we have designed the distribution of VN components (VN topology) over the NIaaS (by using FlowVisor) to finally realize the placement over available resources provided by Grid 5000 platform to ensure media delivery. Once all components have been made active after being instantiated, we have performed some measurements of QoS parameters to monitor the compliance with application constraints at network transport level for the end-to-end network flow (or stream). Two main tests regarding the delay parameters: response time and jitter.

6.3.1 Use-case 1: Round Trip Time measurements

Round Trip Time (RTT) is one of the QoS parameters and represents one of the constraints that an application may dictate when expressing its needs at network level. One of objective of this experiment is to measure the response time or the RTT for an application flow when crossing the network in the architecture implementation described above. From the application perspective, QoS and performance will depend on the RTT measurement. RTT is tightly linked to the number of network nodes to be crossed by these flows.

We consider a linear network topology then conduct 3 experiments where we vary the number of nodes crossed: 3, 7 then 12 nodes to study the impact. We generate traffic using the Ping command tool for each of the three tests where we perform 20 simulations. Each simulation lasts for 240 seconds. Table 7 below shows the experiment results on RTT measurements for this first use-case.

In this scenario, the time needed for a packet to cross one OVS node and one network link is 0.05 *ms*. We have noted an increase in the RTT time each time we increase the number of OVS nodes. The time needed for application flows to go through network links and forwarding elements (OVS nodes) is more important as the number of nodes increases. If an application constraint is to respect a certain RTT value, the network provider through the network controller will program the Other controllers are currently in study nodes for this application accordingly. For instance, VoIP application using RTP is a time-sensitive application that does not tolerates an RTT superior to 300 *ms*. This represents a network response to application SLA requests at the media delivery level and network service delivery level.

6.3.2 Use-case 2: Jitter measurements

We consider building two VNs for the two different applications above. The first applications sends UDP streams and the second application sends RTP streams. We used two flow generators: Iperf Iperf for generating UDP flows and Mausezahn Mausezahn for generating RTP flows. Tables 8 and 9 resume the experiments configurations and characteristics for each application flow and thus for each VN.

We consider changing the VN topology twice. In each topology scenario we conduct three experiments by varying the number of crossed OVS nodes in each VN as shown in the two figures 10 and 11.

Initial results for the two VNs are shown in tables 10 and 11 for both topology scenarios. We can note that the

Experiment	RTT (ms)
1	0,276700
2	0,722950
3	1,222000

Table 7: RTT measurements results

UDP traffic generator	Iperf (UDP)
Number of simulations	10
Duration of each simulation	120 seconds
Number of packets sent	102042
Flow sends	10 Mbps
Packet size	1470 bytes
Speed link	10 Mbps

Table 8: Video Application Flow

jitter is affected by the change of network topology, and the number of crossed nodes. But these jitter changes are not significant and, as expected, they are more important in the case of sharing a common node which is used by two VNs. The VoIP application RTP flows encounter greater variations than the video application UDP flows whose average jitter remains stable. More tests must be necessary to obtain the limits of current behavior and corroborate these initial results.

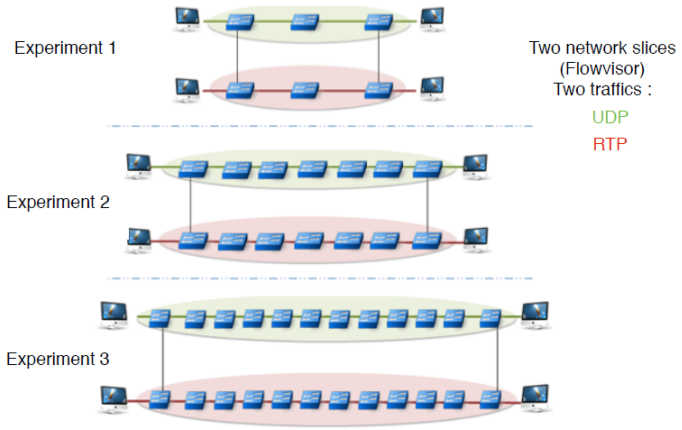

Figure 10: Jitter Measurements - Topology Scenario 1

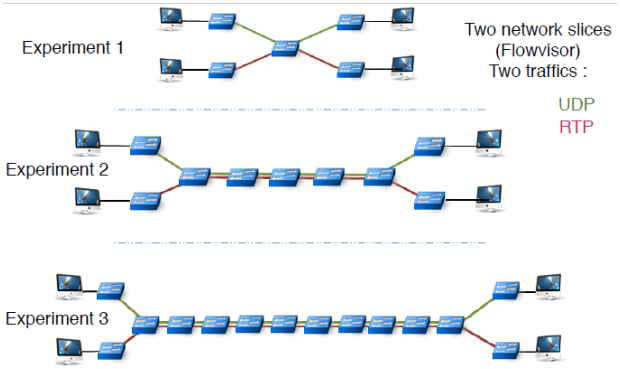

Figure 11: Jitter Measurements - Topology Scenario 2

6.4 Results Analysis

We have seen that the changes involved in NIaaS deployment choices', such as the number of nodes crossed by changing the VN topology, affect the media delivery level. However, these changes do not affect all time the service media delivery. In our case, the response time measured here through RTT, have not been seriously affected. This first experiment also allowed us to set up a calibration phase for decision making. Currently, we complete this experiments with other to find the correlation between Network Service Delivery and Media Service Delivery.

Our study shows that today's tools act at the media delivery level. Indeed, operations consist very often in calculating best paths. A service delivery that seeks to be user-centric and even content-centric, should orchestrate a global and dynamic view of the service to provide. Relying on the network hypervisor is necessary to achieve this objective as it additionally maintains media delivery level dynamically (virtual network topology, link capacities, tolerated response time for links, etc.). Our current work deal to integrate with our implementation the use of Network hypervisor.

7 Conclusions & Future Work

The network ecosystem is driven by major mutations introducing by new uses and divers access network technologies. Network operators need to be able to offer new solutions with flexibility, adaptability and dynamicity. We sustain the idea that NaaS should support on demand user-centric services by deployed them over heterogenous networks. Complementary approches such as Virtualization, SDN, NFV and SOA can be used to design future NaaS architectures.

Our approach is based on a NaaS architecture design and we have examined in this paper the conditions needed so that network virtualization reaches the design objectives of NaaS. Indeed, we define a NaaS model where we distinguish "network equipment virtualization" and "virtualization of network controllers (VNOS)".

NaaS architecture is based on a QoS approach, where QoS captured by the North interface are considering by both phases: VN virtual deployment and VN placement, apply in this fashion QoS constraints at each level of abstraction: NSaaS (Network Service Delivery), NPaaS (NFV catalogues) and NIaaS (NaaS Virtualized infrastructure). NaaS architecture, which provides network services depend on the capacities of resources negotiated at the IaaS. This approach allows to capture by the North interface the QoS constraints to be respected and considering by the placement of virtual nodes and links, and

RTP traffic generator	Mausezahn (RTP)
Number of simulations	10
Duration of each simulation	120 seconds
Number of packets sent	112000
Codec	G711
Transmission delay	1 ms
Packet size	172 bytes
Speed link	50 Mbps

Table 9: VoIP Application Flow

UDP flow	Scenario 1	Scenario 2
Experiment	Jitter (µsec)	Jitter (µsec)
1	80.4	79.95
2	80.90	80.59
3	81.90	81.17

Table 10: Video Application Flow jitter results

RTP flow	Scenario 1	Scenario 2
Experiment	Jitter (µsec)	Jitter (µsec)
1	1.907	3.7322
2	3.2799	34.0190
3	8.1415	75.2890

Table 11: VoIP Application Flow jitter results

also apply these constraints at each level of abstraction (NSaaS, NPaaS, NIaaS).

The proposed approach allows to express the constraints to be respected by the nodes and links at each level of abstraction (service constraints, QoS constraints, placement constraints), applying application requirements along these abstraction levels down to the physical level.

Our proposal suggests a change in network life-cycle within the NaaS by introducing virtualization at deployment phase. Thus, VN virtual deployment phase refers to the VNs design associated to dedicated control planes while the placement phase over physical infrastructure is a problem of VNs placement. The VN virtual deployment phase introduces two new processes: the building of "VN and VNOS", and the process of distribution of network components into the NIaaS.

More precisely, the design phase is characterized by the SLA with its functional and non-functional aspects. Then the first phase of "Virtual Network Deployment" called "Building VN + vNOS" conceives the Virtual Network (VN) and its control plane (vNOS) according to the considered constraints. Later, based on the abstraction of resources, the second phase called "VNF distribution" performs distribution of network functionalities in selected links depending on their offers in CPU and memory capacities.

We conducted a first implementation of the NaaS architecture that allowed us to understand operating the various existing Open-Source experimentation tools. We have concluded that the southbound interfaces meet the advocated promises. However, for more agility, we need to further develop a certain number of features and network automations. Thereby, we could have northbound interfaces up to standard for a network service delivery that answers the various users' requests. NaaS orchestrator is still an open issue.

We have worked to validate our proposal in Grid5000 testbed infrastructure and our first experiments allowed us to implement calibration and monitoring steps that would be very useful to pursue our feasibility and performance study.

Currently, we gradually integrate components to build a satisfactory NaaS that supports realistic network use cases. We work to complete our study in ordre to find the correlation between Network Service Delivery and Media Service Delivery, and by introducing new experiments with other controllers, e.g. ONOS and Opendaylight. Also, we work to make new simulations by using GEANT testbed, because GRID 500 introduces some limitations about for example the control of bandwidth reservation that was not possible to make in our first experience.

Acknowledgements

I would like to thank Amal Kammoun and Mohammed Chahbar for their contribution to the simulation work. We would like to thank Inés Ayadi and Noemie Simoni for their contributions in precedent work cited in this paper.

References

Inés Ayadi, Gladys Diaz, and Noëmie Simoni. Qos-based network virtualization to future networks: An approach based on network constraints. *IEEE Network of the Future (NOF), 2013 Fourth International Conference on the*, 9 (4):1–5, October 2013. doi: 10.1109/NOF.2013.6724515.

A. Boubendir, E. Bertin, and N. Simoni. NaaS architecture through SDN-enabled NFV: Network openness towards web communication service providers. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, pages 722–726, April 2016. doi: 10.1109/NOMS.2016.7502885.

Zdravko Bozakov and Panagiotis Papadimitriou. Autoslice: Automated and scalable slicing for software-defined networks. In *Proceedings of the 2012 ACM Conference on CoNEXT Student Workshop*, CoNEXT Student '12, pages 3–4, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1779-5. doi: 10.1145/2413247.2413251. URL `http://doi.acm.org/10.1145/2413247.2413251`.

K. Channabasavaiah, K. Holley, and E. Tuggle. *OASIS, Reference model for service-oriented architecture*, Oct 2006. URL `http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf`.

NM Mosharaf Kabir Chowdhury and Raouf Boutaba. A survey of network virtualization. *Computer Networks: The International Journal of Computer and Telecommunications*, 54(5):862–876, 2010. doi: 10.1016/j.comnet.2009.10.017.

G. Diaz, M. Sibilla, and N. Simoni. Towards information modeling for a qos-aware support in the lifecycle of virtual networks. In *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*, pages 1–6, Nov 2018. doi: 10.1109/ATNAC.2018.8615430.

Gladys Diaz and Khaled Boussetta. An overview about current Network enablers for the future NaaS architecture. *2016 7th International Conference on the Network of the Future (NOF)*, pages 1–3, 2016. doi: 10.1109/NOF.2016.7810129.

Gladys Diaz and Noëmie Simoni. Network service description for virtual network deployment: A constraints based ovf extension proposal. *IEEE/ACM 12th International Conference on Network and Service Management (CNSM)*, pages 363–366, 2016. doi: 10.1109/CNSM.2016.7818448.

Gladys Diaz, Amina Boubendir, and Noëmie Simoni. *Virtual Deployment of Virtual Networks in Network as a Service*. The Institution of Engineering and Technology (The IET), Michael Faraday House, Six Hills Way, Stevenage, SG1 2AY, United Kingdom, 2017.

D.Kreutz et al. Software-defined networking: A comprehensive survey. In *Proceedings of the IEEE*, volume 103, pages 14–76, Jan 2015. doi: 10.1109/JPROC.2014.2371999.

D. Erickson. The beacon openflow controller. In *Proc. of the 2nd ACM SIGCOMM Workshop Hot Topics Software-Defined Networks*, page 13–18, 2013.

B. Raghavan et al. Software-defined internet architecture: Decoupling architecture from infrastructure. In *Procedeeings of the 11th ACM Workshop Hot Topics Netw.*, pages 43–48, 2012. doi: 10.1145/2390231.2390239.

Floodlight. [online] available on: http://www.projectfloodlight.org/floodlight/.

Project Floodlight. *Floodlight Controller. [Online]. Available on: http://floodlight.openflowhub.org/*, 2012.

FlowVisor. [online] available on: www.sdxcentral.com/projects/on-lab-flowvisor/.

Flowvisor Controller. [online] available on: https://github.com/opennetworkinglab/flowvisor.

GRID5000. [online] available on: www.grid5000.fr.

Bo Han, V. Gopalakrishnan, Lusheng Ji, and Seungjoon Lee. Network function virtualization: Challenges and opportunities for innovations. *Communications Magazine, IEEE*, 53(2):90–97, Feb 2015. ISSN 0163-6804. doi: 10.1109/MCOM.2015.7045396.

Indigo. [online] available on: http://www.projectfloodlight.org/indigo/.

Iperf. [online] available on: https://iperf.fr.

Amal Kammoun, Nabil Tabbane, Gladys Diaz, Abdulhalim Dandoush, and Nadjib Achir. End-to-end efficient heuristic algorithm for 5g network slicing. *The 32-nd IEEE International Conference on Advanced Information Networking and Applications (AINA-2018)*, Mai 2018. doi: 10.1109/AINA.2018.00065.

Mausezahn. [online] available on: http://netsniff-ng.org.

N. McKeown, T. Anderson, H. Balakrishnan, and al. Openflow: enabling innovation in campus networks. 38(2):69–74, 2008. doi: 10.1145/1355734.1355746.

N. Niebert, S. Baucke, I. El-Khayat, M. Johnsson, B. Ohlman, H. Abramowicz, K. Wuenstel, H. Woesner, J. Quittek, and L.M. Correia. The way 4ward to the creation of a future internet. In *Personal, Indoor and Mobile Radio Communications, 2008. PIMRC 2008. IEEE 19th International Symposium on*, pages 1–5, Sept 2008. doi: 10.1109/PIMRC.2008.4699967.

Ofsoftswitch. [online] available on: https://github.com/jean2/ofsoftswitch13.

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

ONOS: An open source distributed SDN OS. [online] available on: https://onosproject.org.

Open-vSwitch. [online] available on: www.openvswitch.org.

Opencontrail. [online] available on: https://tungsten.io.

Opendaylight. [online] available on: https://www.opendaylight.org.

OpenSDN Core. [online] available on: https://www.opensdncore.org/en/opensdncore/testbedcomponents.

OVX. [online] available on: https://openvirtex.com.

Pantou. [online] available on: https://www.sdxcentral.com/projects/pantou-openwrt/.

POX. [online] available on: https://github.com/noxrepo/pox.

A. V. Vasilakos Q. Duan, Y. Yan. A survey on service-oriented network virtualization toward convergence of networking and cloud computing. *IEEE Transactions on Network and Service Management*, 9(4):373–392, dec 2012. doi: 10.1109/TNSM.2012.113012.120310.

Ryu. [online] available on: https://osrg.github.io/ryu/.

Qing Shen. *Vers un Contrôle Programmable pour le Futur Internet et le Cloud Computing*. PhD thesis, PhD repport. TELECOM ParisTech, 29 Septembre 2014.

xDPd. [online] available on: https://github.com/damomeen/xdpd-for-ezappliance.