Towards a Telco Cloud Environment for Service Functions

Journal:	*IEEE Communications Magazine*
Manuscript ID:	COMMAG-14-00540
Topic or Series:	February 2015/Network and Service Virtualization
Date Submitted by the Author:	25-Jun-2014
Complete List of Authors:	Soares, João; Portugal Telecom Inovação e Sistemas, Gonçalves, Carlos; Instituto de Telecomunicações, Parreira, Bruno; Instituto de Telecomunicações, Tavares, Paulo; Instituto de Telecomunicações, Carapinha, Jorge; Portugal Telecom Inovação e Sistemas, Barraca, Joao Paulo; Instituto de Telecomunicações, ; Universidade de Aveiro, Departamento de Electrónica Telecomunicações e Informática Aguiar, Rui L.; Universidade de Aveiro, Instituto de Telecomunicações Sargento, Susana; Universidade de Aveiro, Instituto de Telecomunicações
Key Words:	Service Function, Service Function Chaining, Network Function Virtualization, Software-Defined Networking, Cloud Computing

Towards a Telco Cloud Environment for Service Functions

João Soares[1,2], Carlos Gonçalves[2], Bruno Parreira[1,2], Paulo Tavares[1,2] Jorge Carapinha[1], João Paulo Barraca[2], Rui L. Aguiar[2], Susana Sargento[2]

[1]: Portugal Telecom Inovação e Sistemas, Portugal, {joao-m-soares, bruno-m-parreira, paulo-c-tavares, jorgec}@telecom.pt
[2]: Instituto de Telecomunicações, Universidade de Aveiro, Portugal, {cgoncalves, jpbarraca}@av.it.pt {ruilaa, susana}@ua.pt

***Abstract*— Deploying Service Functions (SFs) is an essential action for a network provider. However, the action of creating, modifying and removing network SFs is traditionally very costly in time and effort, requiring the acquisition and placement of specialized hardware devices and its interconnection. However, the emergence of concepts like Cloud Computing, Software Defined Networking (SDN), and ultimately, Network Functions Virtualization (NFV) is expected to raise new possibilities about the management of SFs with a positive impact in terms of agility and cost. From a Telco viewpoint these concepts can help to reduce both Operational Expenditure (OPEX) and Capital Expenditure (CAPEX), and open the door to new business opportunities. In this article, we identify how Telcos can benefit with the abovementioned paradigms, and explore some of the aspects that still need to be addressed in the NFV domain. We focus on two problems with the current state-of-the-art: enabling Telco infrastructures to adopt this new paradigm; orchestrating and managing SFs towards Telco-ready cloud infrastructures. The technologies we describe enable a Telco to deploy and manage SFs in a distributed cloud infrastructure. Special attention is given to the way SFs are modeled towards cloud infrastructure resources. In addition, we explore the ability to perform Service Function Chaining (SFC) as one of the fundamental features in the composition of SFs. Finally, we describe a proof of concept that demonstrates how a Telco can benefit from the described technologies.**

Keywords—Service Function, Service Function Chaining, Network Function Virtualization, Software-Defined Networking, Cloud Computing.

I. INTRODUCTION

The emergence of the Cloud concept, its ongoing evolution and the opportunities that it brings, has led many businesses to adapt in order to get the most utility out of it. One can say that the Telco sector is today one of the most active business sectors exploring the opportunities offered by the Cloud. The relationship and inter-dependency between Cloud and Telco can be analyzed from two distinct perspectives:

- **The Telco *supporting* the Cloud**: In a Cloud environment, communication end points are user devices and virtual machines that can be hosted in different physical locations, according to varying conditions. In addition, network capacity requirements are no longer static, but are likely to change as the associated computing and storage resources expand and reduce. This poses a whole new set of challenges to the network, including the data center (DC) and the Wide Area Network (WAN) segments. To provide assured levels of performance to cloud services, cloud and Telco services need to be provisioned, managed, controlled and monitored in an integrated way.
- **The Telco *using* the Cloud**: Today, the establishment, management and composition of SFs (e.g. router, firewall) follows a rigid, static and time consuming process – e.g. resource overprovisioning is usually necessary to cope with estimated peak demand: a fault in a single function can disrupt an entire network, imposing the need for disaster recovery methods, which are slow. A SF is considered a functional block that is responsible for a specific treatment of received packets and has well-defined external interfaces. In practice, a SF can be embedded in a virtual instance or directly in a physical element [6]. In this work we focus on the former case. One of the topics that arise from the combination of SFs is SFC. Briefly, SFC consists of a set of SFs interconnected through the network in a specific order. It can be considered as a particular case of service composition. It requires the placement of SFs and the adaptation of traffic forwarding policies of the underlying network to steer packets through a chain of service components. Furthermore, the lack of automatic configuration and customization capabilities increases the operational complexity. As virtualization technologies reach maturity and are able to provide carrier-grade performance and reliability, it becomes feasible to consolidate multiple network equipment types, traditionally running on specialized hardware platforms, onto industry standard hardware, which minimizes costs, reduces time-to-market and facilitates open innovation. Cloud Computing, along with SDN [1] and ultimately NFV [2], promises to make the SF management process much more agile. Cloud Computing represents a paradigm for Information Technology (IT) services, which can now be delivered in an on-demand and self-service manner. SDN brings new capabilities in terms of network automation, programmability and agility that facilitate the integration with the cloud. With respect to NFV, from a high-level perspective, it envisions accelerating the innovation of networks and services, allowing, among other things, new operational approaches, novel services, faster service deployment (shorter time to market), service assurance and security.

In this article, we explore how Telcos can take advantage of these concepts to improve the management of SFs and potentially build new business models. First, we highlight the Telcos privileged position in this area compared to traditional

cloud providers. We then present Cloud4NFV, a platform for managing SFs in a Telco cloud environment. Later, we focus on SF modeling towards cloud infrastructure resources. Special attention is given to the ability to perform SFC. To emphasize possible application scenarios of the solution presented in this paper, a proof of concept is then detailed. Finally, we point out future work directions and conclusions.

II. CARRIER CLOUD

Traditional cloud infrastructures are far from being suitable for all types of business, especially when referring to network SFs. Network SFs have carrier grade requirements, from guaranteed Quality of Service (QoS) in terms of IT resources and network connectivity, to high-availability and fast fault recovery through redundancy. Telcos, with their already established distributed network infrastructure and hosting centres, are ideally positioned to take the lead in this area, as they can create a compelling end-to-end cloud proposition that integrates their network management capabilities, adapted to a more agile and cloud service-oriented operation model (on-demand, self-service, elastic). Furthermore, they have the experience in providing carrier grade service levels.

We envision a near-future Telco cloud infrastructure that comprises, not only the traditional DC domains, but also the WAN domain. In such scenario, the Telco takes advantage of its already established distributed facilities (sometimes referred as Points of Presence, PoPs) to host small cloud environments. It is also possible for this distributed cloud infrastructure to extend itself into the costumer site. Figure 1 depicts this scenario.

III. CLOUD4NFV PLATFORM

CloudNFV platform builds upon Cloud, SDN and WAN technologies to allow SFs to be managed in an *as-a-Service* basis. The platform is targeted for Telcos to improve the management of SFs within their environment, but can also be used to build new services based on the concept of *Service Function as-a-Service* (SFaaS), in which case SFs or bundles containing a combination of SFs can be offered as a service. This concept is also envisioned in [3].

A. Functionalities

The most relevant functionalities of Cloud4NFV are:

- Automated deployment, configuration and lifecycle management (e.g. instantiation, configuration, update, scale up/down, termination, etc) of SFs.
- Exposure of functionalities such as: service deployment and provisioning; service monitoring and reconfiguration; and service teardown.
- Federated management and optimization of WAN and cloud resources for accommodating SFs.
- Supports composition of SFs through SFC.

B. Architecture

Figure 1 provides an overview of the system, which is organized in four major planes: Infrastructure Plane, Virtual Infrastructure Management Plane, Orchestration Plane, and Service Plane. The Service Plane handles the services that are built on Cloud4NFV, and the Infrastructure Plane comprises all physical resources. Special attention will be given to the Infrastructure Management Plane, since we consider it a major lever for enabling SFC. It is important to note that this architecture is aligned with the ETSI NFV architectural guidelines [4]. This fact is highlighted along the description of the platform.

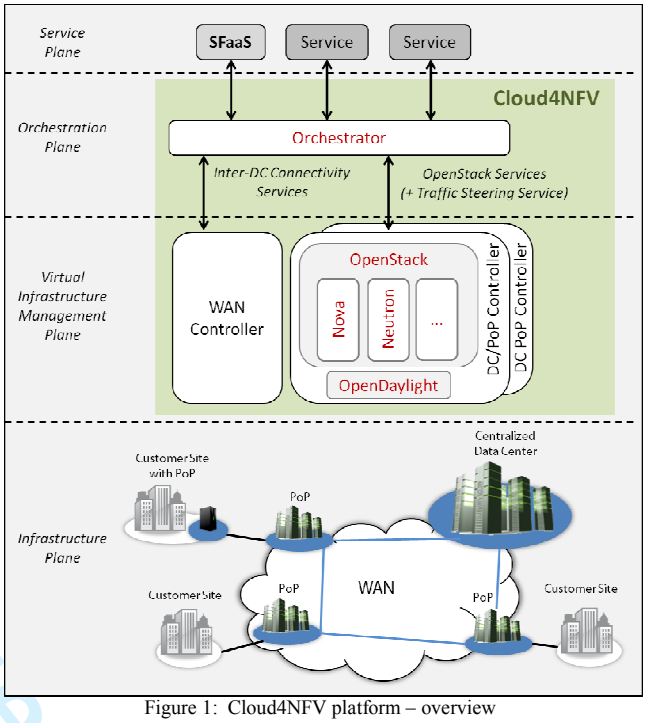

Figure 1: Cloud4NFV platform – overview

C. Orchestrator

The Orchestrator is responsible for the automated provision, management and monitoring of SFs over the virtual infrastructure. It exposes the ability to create and delete SFs, as well as the ability to chain SFs. It relies on the Virtual Infrastructure Management Plane to provision the infrastructure resources where SFs run (virtual machines, virtual networks, etc). Looking to the ETSI NFV reference architectural framework [4], this component considers the *Orchestrator* and the *VNF Manager(s)* entities. The orchestrator has an interface (REST) that exposes the ability to create and delete SFs as well as to chain SFs.

D. Virtual Infrastructure Management Plane

The Virtual Infrastructure Management Plane includes the components for management of infrastructure resources. It includes cloud DC controllers (one per DC) and a WAN controller that is able to establish inter-DC connectivity services. The Virtual Infrastructure Management Plane can be seen as the *Virtual Infrastructure Manager(s)* in the ETSI NFV reference architectural framework [4].

E. Data Center Controller(s)

Although the cloud model may require, to a large extent, the redefinition of SFs and the way they are managed, SFs

also require adaptation from today's cloud solutions to cope with their requirements, especially in terms of networking features. A clear evidence of this fact is the OpenStack[1] project, a reference open-source cloud management platform, which has been witnessing a tremendous evolution of its networking features - in its networking project mostly known by the codename *Neutron*. It is also important to note that *Neutron* provides the network service logics, and relies on different backends called "drivers" to interact with different networking technologies. Among these drivers is a recent one for the OpenDaylight[2] SDN controller. OpenDaylight is today seen as an initiative equivalent to OpenStack in the SDN domain. With this in mind, the DC Controller is based on OpenStack and OpenDaylight.

1) OpenStack

From a networking perspective, OpenStack today allows to create and manage *virtual networks* (L2 network segments) and *virtual ports* (attachment points for devices connecting to networks, e.g. virtual Network Interface Cards (vNICs) in virtual machines). The OpenStack community has been doing a considerable effort on keeping up with users' demand on introducing new *Neutron* network service types - L3 routing, firewall as a service (FWaaS), load balancer as a service (LBaaS) and VPN as a Service (VPNaaS); however, it is unfeasible in the long run to keep up with demands at this pace in a timely manner. Therefore, we argue that OpenStack should also offer the basic tools for network services to be orchestrated at a higher level and be deployed as virtual machines.

With the orchestration and composition of SFs in mind, it is easy to identify the need to fill a gap in OpenStack, the one of being able to steer traffic between OpenStack elements (e.g. virtual machines, routers). We envision a new OpenStack service abstraction that allows steering traffic between *Neutron* "ports" according to classification criteria. The current definition of the abstraction introduces new entities into the OpenStack *Neutron* data model: *Port Steering*, and *Classifier*. Both entities have a set of common OpenStack data model attributes, i.e. *id*, *name*, *description*, and *tenant_id*. The *Port Steering* adds to this common set a list of ports (*ports* attribute), and a list of classifiers (*classifiers* attribute). The former lists the sets of ports that must be targeted of classification and then steered. The *Classifier* entity adds the following attributes: *type*, *protocol*, *port_min*, *port_max*, *src_ip* and *dst_ip*. The purpose of this latter class is further explained ahead. As one will see, this functionality is very useful as it provides the means to realize, among other things, SFC. How the SFC is actually performed is further explained below. More details about the traffic steering functionality can be found in the OpenStack proposal[3], for which we developed a prototype implementation.

2) OpenDaylight

OpenDaylight has a module that integrates with OpenStack *Neutron* for the actual enforcement of services in the infrastructure. This module was extended in order to support and enforce the previously referred OpenStack traffic steering feature. It is important to highlight that this implementation relies on the OpenFlow and the Open vSwitch Database Management Protocol (OVSDB) for the management of network resources.

F. Wide Area Network Controller

The WAN Controller is responsible for managing the operator network, and it exposes connectivity services to the upper layers (in this case the orchestrator). In this context, WAN services are used to support SFs (SF is the client of the WAN service). Point-to-point and multi-point connections with guaranteed network QoS are provided. The details and mechanisms to manage the automatic establishment of connectivity services across different locations are detailed in [5].

IV. SERVICE FUNCTION VIRTUALIZATION

Figure 2 shows how SFs are modeled towards virtual infrastructure resources. Each class is detailed below.

Service Function: represents an instance of a functional block responsible for a specific treatment of received packets that has well-defined external interfaces – *Service Function Endpoints* (SFEs).

Service Function Endpoint (SFE): represents an external interface of one SF instance that is always associated to a SF. Each SFE can have associated information regarding layer 1 (e.g. physical/virtual interface), layer 2 (e.g. MAC address) and/or layer 3 (e.g. IP address), or even regarding higher layers (e.g. HTTP).

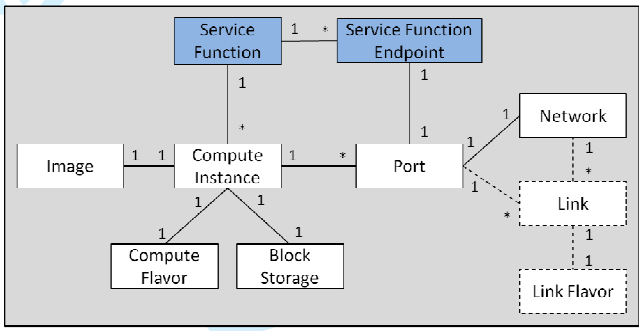

Figure 2: Service Function data model towards a Cloud Infrastructure

From an infrastructure perspective, the resources considered to realize a SF are: *Compute Instance* (i.e. Virtual or Physical Machines), *Image* (disk image), *Compute Flavor* (hardware specification of a compute instance, i.e. CPU, memory and root disk), *Block Storage* (additional disks), *Port* (i.e. network interface), *Network* (a network segment), and *Link* (a connection between two *Ports* from different *Compute Instances*) which has an associated *Link Flavor* (dedicated QoS in terms of bandwidth, delay and jitter). A SF can be associated to multiple *Compute Instances*, while the latter has a single *Image*, a single *Flavor* and can have multiple *Ports*

[1] OpenStack, http://www.openstack.org/
[2] OpenDaylight, http://www.opendaylight.org/
[3] OpenStack Traffic Steering blueprint, https://review.openstack.org/#/c/92477/

and many *Block Storages*. A *Port* can only be associated to a single *Network*; however, it can be associated to multiple *Links*. A SFE is directly associated to a *port*, but not all *ports* need to map to SFEs.

The network QoS, represented in this model by *Link* and *Link Flavor*, is not considered in today's cloud infrastructure systems. However, for a carrier grade cloud this is a must, and OpenStack already has an ongoing project to support it[4].

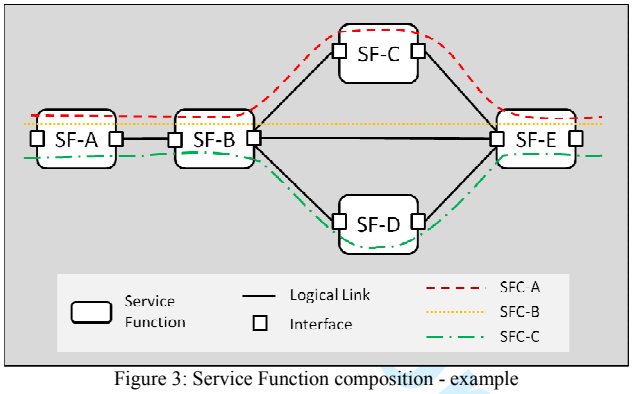

Figure 3: Service Function composition - example

A. *Service Function Composition*

With virtual SFs there will be a greater need to compose and organize virtual SFs dynamically. Figure 3 presents an example of how several SFs can eventually be composed and organized. The relation between SFs presented in this figure resembles what the ETSI NFV refers to as Forwarding Graph (FG). According to ETSI, a FG is a "*graph of logical links connecting network function nodes for the purpose of describing traffic flow between these network functions*" [7]. The FG concept is intimately related to the one of SFC. One could think of a FG modeled by multiple SFCs. Figure 3 shows how the FG can be materialized relying on three SFCs.

V. SERVICE FUNCTION CHAINING

SFC is loosely defined as "*an ordered set of service functions that must be applied to packets and/or frames selected as a result of classification*" [6]. Although there are very good and important contributions (e.g. in IETF and ONF), work is still required, namely when it comes to the details on how to model and actually realize SFCs. But first we highlight the advantages of SFC in the scope of one of the most prominent NFV use-cases [2].

A. *Costumer Premises Equipment Use-Case*

Customer Premises Equipment (CPE) SFs are often pointed out as one of the most suitable candidates for virtualization [2] [8]. If we look to the particular case of the enterprise grade CPEs, SFC will surely play a particular relevant role.

The CPE is a collection of SFs, which can be standard routing nodes with Network Address Translation (NAT) and Firewall (FW) capabilities, but also Voice over IP (VoIP) servers, Virtual Private Network (VPN) servers (possibly extending to the operators network), Wan Optimization Controllers (WOC), Deep Packet Inspection (DPI) or Intrusion Prevention System (IPS). Although these services are combined in a single network infrastructure, they are deployed for different scenarios and not all traffic needs to traverse them, leaving room for optimization through SFC, and its capability of relocating functions across network elements. Below we present a summary of SFC examples for CPE.

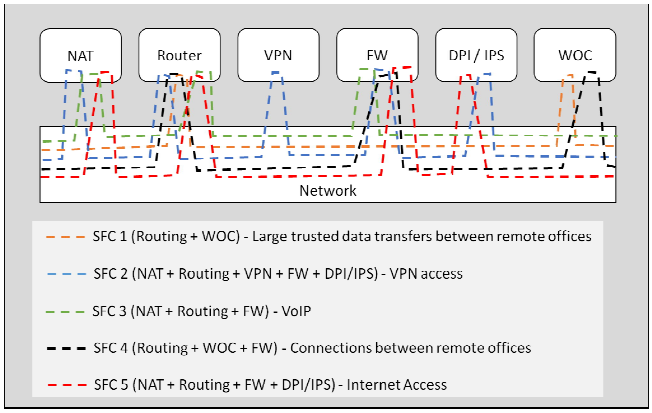

Figure 4: CPE Use-Case

It should be noted that some of the chains can even be temporary (e.g. SFC-1), which states the need for a model that enables the dynamic definition of chains.

B. *Fundamentals*

In SFC two aspects are vital:

Classification: a policy for matching packets (e.g. *HTTP* traffic) used for the identification of appropriate actions (e.g. forwarding). It can be for example an explicit forwarding entry in a network device that forwards packets from one address (e.g IP, MAC) into the SFC. (Re)Classification can also occur at each SF of the SFC independent from the previous SFs. In other words, there can be multiple classification points within one SFC. In this sense, multiple classification policy entries should be allowed in an SFC system.

Traffic Steering: ability to manipulate the route of traffic, i.e. delivering packets from one point to another, at the granularity of subscriber and traffic types [9]. The actual network topology or overlay transports should not be modified to accomplish this.

Moreover, the combination of classification and traffic steering can be done in two ways:

Tagged packet approach: classification can occur only at the initial redirection points to a SFC, if upon this classification packets are tagged. After that, packets are steered to the SFC and routed along it according to the embedded tags.

Non-tagged packet approach: classification occurs not only at the redirection points but also at each hop of the SFC. In this case, packets are not tagged and are subject of classification and steering at each SFC hop.

[4] OpenStack Neutron QoS support
https://wiki.openstack.org/wiki/Neutron/QoS

While the first might look a simpler approach, it is in fact the most complex one. It requires a system to not only have the ability to tag and forward packets, but also SFs have to be able to process the tagged packets. In summary, to adopt such an approach, although feasible, it requires a high adaptation effort. In this sense, in this work we adopt the non-tagged packet approach.

Further aspects should be taken into account when elaborating a SFC solution, such as: i) no assumption should be done on how functions are deployed, i.e. whether they are deployed on physical hardware, as one or more VMs, or any combination thereof; ii) a SF can be part of multiple SFCs; iii) a SF can be network transport independent; iv) a SFC allows chaining of SFs that are in the same layer 3 subnet and of those that are not; v) traffic must be forwarded without relying on the destination address of packets; vi) classification and steering policies should not need to be done by SFs themselves [10].

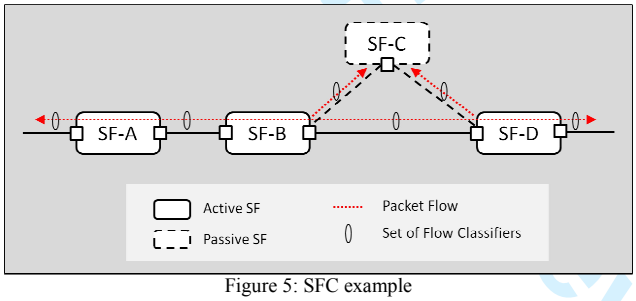

Figure 5: SFC example

C. Service Function Categories

Two categories of SFs have been defined.

Active SFs: those that are in fact part of the main course of a packet, in which case two sub-types are considered: a) functions that may drop packets or forward them, such as a Firewall; b) functions that can actually change packets, e.g. an IPSec VPN server.

Passive SF: are considered to be out of the main course of the chain. These functions mainly inspect packets, e.g. a monitoring system or a DPI. In practice one can think of a SF in a physical device connected to a hub through a single network interface configured in promiscuous mode. Traffic is considered to be duplicated when having to reach a passive function.

These two categories are important because they impose constrains on how classification and steering can be implemented. In short, passive functions can rely on packet characteristics as packets are not modified, while active functions must be integrated at a service level because ingress and egress packets can be completely unrelated (e.g., VPN). If a SFC has active functions that change packets, the classification may differ when passing one of these functions.

D. Service Function Chaining Abstraction Model

Having in mind the considerations made so far, a base data model for SFC (that supports both tagged and non tagged approaches) is now presented. Naturally, other SFC service abstraction proposals may appear in the future, but we consider that this model lays a strong foundation over which other service abstractions can easily be created by extending the model. Figure 6 depicts the model. Five main classes are considered: *Service Function Chain*; *Service Function; Service Function Endpoint; Packet Flow* and *Classifier*.

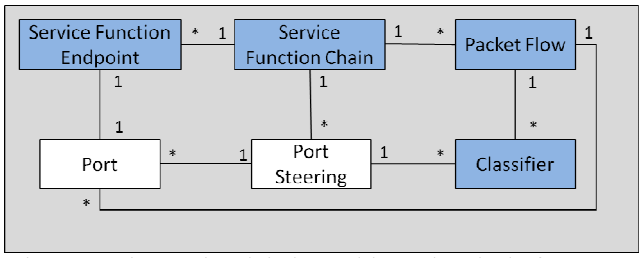

Figure 6: Service Function Chain data model towards a Cloud Infrastructure

All classes have the following attributes: *id*, *name*, and *description*. The *id* refers to an unique identifier able to identify the class instance within the SFC system. The remaining two, *name* and *description*, are attributes that allow a human-readable characterization of the class instance. Below it is provided further detail about each class.

Service Function Chain (SFC): a SFC has a set of *Service Functions* (SFs) associated and an attribute that defines the ordered sequence of functions (*path*). Since a function can have more than one SFE, the *path* attribute is specified by an ordered list of SFEs organized by hops. For example:

- "*path= { hop={SF-A_E2, SF-B_E1}; hop={SF-B_E2, SF-D_E1}, passive={SF-C_E1} }*" where the chain crosses SF-A, SF-B, and SF-D and has SF-C as a passive function between SF-B and SF-D.

Classifier: a *classifier* represents a classification criteria applied to a packet, which determines if the packet matches that specific criteria or not. In this sense, a *classifier* has an attribute *filter* that contains the classification criteria, e.g.:

- "*filter={protocol='6'; port='80-90'; source_IP='192.168.10.20/32'; destination_IP='192.168.10.40/32'}*" - matches all TCP traffic using ports between 80 and 90 with source IP address 192.168.10.20 and destination IP address 192.168.10.40.

Packet Flow: One *classifier* only identifies packets with a certain criteria, while a *packet flow* represents an aggregator of *classifiers*. In this sense, a *packet flow* can have multiple *classifiers*, and a *classifier* can be associated to multiple *packet flows*. Moreover, a *packet flow* has a *source* and *destination*. The former identifies where the initial classification and redirection of the packet flow to the SFC takes place, while the latter identifies where packets are to be delivered after passing through the SFC. The attributes considered so far would be enough if the system realizing the SFC followed a tagged packet approach. For a non-tagged approach, an additional attribute is considered - *sfc_classifiers*. Due to the possibility of (active) SFs to modify packets, the classification initially done may not be the same along all hops

of the SFC, and therefore, the *sfc_classifiers* attribute matches the classification criteria (*classifiers*) at each hop of the SFC. Furthermore, the attribute *direction* is also considered to identify the direction of the SFC which the *packet flow* must traverse – this attribute can assume one of two values: *forward*, *reverse*. We consider that multiple *packet flows* can be associated to a single SFC instance.

Port Steering: this entity refers to the functionality presented above in OpenStack. This feature allows steering traffic between *ports*.

In terms of operations, all classes are considered to allow CRUD (Create, Read, Update and Delete) operations.

VI. Proof of Concept

A proof of concept environment has been deployed to showcase how a Telco can leverage the features described in this work.

The testbed in place resembles the infrastructure depicted in Figure 1. At the core of the operator network there is an IP/MPLS backbone composed of four provider (P) routers and four provider edge (PE) routers. The core routers are connected in full-mesh and in a one to one relationship with the PE routers. The WAN is managed by a proprietary OSS. The core network connects to two DC premises (managed by OpenStack IceHouse release with the traffic steering functionality), where one represents a centralized DC and the other a PoP. Finally, there is a customer premises represented by switching equipment, which is directly connected to the PoP.

A prototype of the Cloud4NFV was developed using the python language, which interacts with both the WAN controller and DC controllers. Details regarding the orchestrator implementation (e.g. RESTful API) can be found in [11].

With this setup we are able to automatically provision SFs (that are in the Cloud4NFV orchestrator repository) across multiple DC locations with guaranteed WAN connectivity.

A. Service Function as a Service (SFaaS)

In order to improve the showcase, at the service layer we implemented a prototype of the SFaaS concept. This is exposed via a web portal[5]. Currently, CPE functions (e.g. Routing+NAT, Firewall, VPN, VoiP server, Video Conference server) are available in the SFaaS, and the ability to perform SFC is not exposed to the end-user. The user requests CPE SFs, that already have a pre-determined relation with other SFs, and associates them to one of his sites. The instantiation and configuration of the SFs is done in a few minutes and the user is able to control them through a dedicated SF management portal. Note that the use of the SFaaS service requires the establishment of a basic business relation between the costumer and the Telco (costumer sites registered and with connectivity services).

[5] SFaaS, http://occi.av.it.pt:3000/, guest credentials: login: *guest@ptinovacao.pt* password: *guest123*

VII. Future Work

Currently, the proof of concept does not support the enforcement of network QoS with the DC domains; this is only supported in the WAN connectivity services. We expect to add this support by the time OpenStack officially releases this feature. Furthermore, runtime management operations (such as scaling and migration of SFs) are yet to be included in the platform. On the WAN domain, we are currently adding a SDN-based network. The purpose is to have both legacy and SDN network technologies in place to better evaluate the advantages and disadvantages of having one approach or the other. Finally, we are working on exposing the ability of performing SFCs to the end-user.

VIII. Conclusions

The orchestration and management of SFs is today a complex task that takes considerable time and effort. However, the concepts like Cloud Computing, SDN and NFV are opening the way to handle SFs in a much more flexible and agile manner. The Telco will play a key-role in this scenario, and we have given some insights on how that will be done. A platform for managing virtual SFs functions in a Telco cloud infrastructure has been presented. Special attention has been given to the modeling of SFs towards cloud resources and to the combination of SFs, i.e. SFC. Finally, we described a proof of concept that showcases how the platform and principles presented can be leveraged in a Telco environment.

References

[1] H. Kim, N. Feamster, "Improving network management with software defined networking," Communications Magazine, IEEE , vol.51, no.2, pp.114,119, February 2013.

[2] ETSI, Network Functions Virtualisation (NFV): Use Cases, Technical Report ETSI GS NFV 001 v1.1.1, Oct. 2013.

[3] G. Xilourisl, E. Trouva, F. Lobillo, J.M. Soares, J. Carapinha, M.J. McGrath, G. Gardikis, P. Paglierani, E.Pallis, L. Zuccaro, Y. Rebahi, A. Kourtis, "T-NOVA: A Marketplace for Virtualized Network Functions", European Conference on Networks and Communications (EUCNC 2014), June 2014.

[4] ETSI, Network Functions Virtualisation (NFV): Architectural Framework, Technical Report ETSI GS NFV 002 v1.1.1, Oct. 2013.

[5] H. Puthalath, J. Soares, et al., "Negotiating On-demand connectivity between clouds and wide area networks," IEEE CloudNet, Paris, 2012.

[6] P. Quinn, Kumar, T. Nadeau, "Service Function Chaining Problem Statement," IETF Internet draft, Informational, December. 2013.

[7] ETSI, Network Functions Virtualisation (NFV): Terminology for Main Concepts, Technical Report ETSI GS NFV 003 v1.1.1, Oct. 2013.

[8] Alcatel Lucent, "Network Functions Virtualization – Challenges and Solutions", White Paper, 2013.

[9] Y.Zhang, N. Beheshti, L. Beliveau, G. Lefebvre, R. Manghirmalani, R. Mishra, R. Patney, M. Shirazipour, R. Subrahamaniam, C. Truchan, Tatipamula, M., "StEERING: A Software-Defined Networking for Inline Service Chaining," Proceedings of. IEEE ICNP 2013, Goettingen, Germany, Oct. 2013.

[10] W. John, K. Pentikousis, G. Agapiou, E. Jacob., M. Kind, A. Manzalini; F. Risso, D. Staessens, R. Steinert, C. Meirosu, "Research Directions in Network Service Chaining," *Future Networks and Services* (SDN4FNS), 2013 IEEE SDN for , vol., no., pp.1,7, 11-13 Nov. 2013.

[11] J. Soares, B. Parreira, M. Dias, J. Carapinha, S. Sargento, "Cloud4NFV: A Platform for Virtual Network Functions", submitted to IEEE CloudNet 2014.