Programming Tools for Rapid NFV-Based Media Application Development in 5G Networks

Ugur Acar
Netas Telekomunikasyon Anonim Sirketi
Istanbul, TURKEY
uacar@netas.com.tr

Refik Fatih Ustok
Netas Telekomunikasyon Anonim Sirketi
Istanbul, TURKEY
fustok@netas.com.tr

Selcuk Keskin
Netas Telekomunikasyon Anonim Sirketi
Istanbul, TURKEY
selcukk@netas.com.tr

David Breitgand
IBM Research
Haifa, ISRAEL
davidbr@il.ibm.com

Avi Weit
IBM Research
Haifa, ISRAEL
weit@il.ibm.com

***Abstract*—The emergence of virtualisation and Infrastructure-as-a-Service (IaaS) have dramatically transformed the telecom industry through network function virtualisation (NFV). A recently introduced cloud-native concept, Platform as a Service (PaaS), ensures to further boost the performance, portability and cost efficiency of the NFV. 5G-MEDIA project proposes the application of a serverless paradigm known as Function-as-a- Service (FaaS) to NFV for the media applications exploiting the 5G technologies. In addition to integration of FaaS, the 5G-MEDIA application/service development kit (SDK) supports microservice-based application development for both hypervisor-based and containerized approaches, specifically supporting Docker, unikernel and LXC. In this paper, we provide an overview of the 5G-MEDIA SDK which is built to support NFV-based next generation media applications and to achieve a development time in the order of minutes. Furthermore, implementations of FaaS Emulation and FaaS command line interface (CLI) tools are also presented.**

***Index Terms*—5G, 5G MEDIA, NFV, SDK**

I. Introduction

Telecommunications industry is witnessing an extraordinary global demand of mobile data volume that is expected to reach 30.5 exabytes per month by 2020 [1]. The 5th generation (5G) wireless systems are expected to satisfy this demand improving the network performance in terms of energy consumption, throughput and latency, thus attribute to the rapidly increasing number of network-connected end devices, internet users with heavy usage patterns and popularity of applications including cloud computing, immersive applications, smart video streaming and gaming applications [2]. The H2020 5GPPP Phase 2 project 5G-MEDIA aims at innovating media-related applications by investigating how these applications and the underlying 5G network should be coupled and interwork to the benefit of both [3]. The project aims to develop the 5G-MEDIA platform which provides mechanisms to flexibly adapt service operations to dynamic conditions and react upon events [4]. FaaS which was introduced by Amazon in 2014 [5] allows event-driven on-demand virtual network function (VNF) instantiation and execution in contrast to traditional virtual machine (VM) approach where virtual appliances are continuously running and thus result in low utilisation. To the best of our knowledge, FaaS has not been applied to NFV prior to 5G-MEDIA. One of the main challenges that we face in design and development of the 5G-MEDIA SDK is harmonization between the existing VM oriented SDK and tools and the novel FaaS approach. In our architecture, we elegantly combine the two approaches within a single framework that addresses all aspects of a media intensive application lifecyce management in 5G networks.

The authors of [6] has discussed the service development kit (SDK) of SONATA project which effectively extends Management and Orchestration (MANO) framework and can be considered as a sandbox to develop and try-out NFV-based network services (NS). 5G-MEDIA SDK provides application/service developers with an open environment for the creation of new Network Apps. SDK supports the new programming model and provide a set of well-integrated open source networking-related and media specific proofing and packaging tools, libraries, repositories and catalogues assisting the service development, emulation, testing and validation process, prior to the deployment phase. SDK supports microservice-based apps development, based on the integration of innovative and open source tools related to serverless computing (e.g. OpenWhisk). In addition, provided services through SDK offers an alternative development and deployment choices that push the necessity of appropriate talent and technological barriers aside and enable rapid service creation.

In this paper, we focus on 5G-MEDIA SDK which is a set of tools that allows for the creation of applications, network services or functions and supports developers in implementing, packaging, deploying, analyzing the software. 5G-MEDIA SDK integrates open-source tools related to serverless computing and thus it aims to achieve a development time in the order of minutes. The innovations that 5G-MEDIA SDK tools brings compared to the existing platforms can be listed as follows:

- 5G-MEDIA SDK includes an all-in-one UI which enables developers access all SDK tools in a single interface,

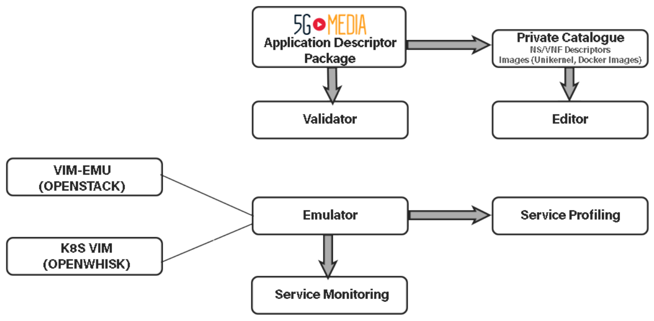

Fig. 1. 5G-MEDIA Overall SDK Interactions

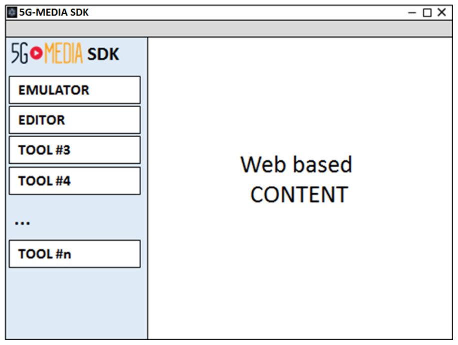

Fig. 2. 5G-MEDIA All in One User Interface

therefore it improves user-time efficiency and provides smooth user experience.

- FaaS Emulation (Lean OW) and FaaS CLI Tools implemented in 5G-MEDIA SDK allows media application developers (network service developers) to quickly develop value added code while relieving them from the infrastructure management concerns.
- VNF/NS Emulation toolkit including monitoring tools provides visualization of pre-defined performance metrics in the emulated multi-vim environment (i.e Openstack and Openwhisk) (emulator). This allows media application developers to test, verify their applications functionality and fine-tune their media application performance before deploying to a live environment.

This document is organized as follows. In Section 2, each SDK tool is described in detail such as all-in-one user interface (UI), editor, validator, emulator and service monitoring. In Section 3, we present an example workflow of 5G-MEDIA SDK dealing with a media application. Finally we conclude this paper with conclusions in Section 4.

II. 5G MEDIA SDK TOOLS

5G-MEDIA SDK provides a set of tools that helps developers easily implement and deploy new media related network applications to the Service Virtualization Platform (SVP). The SDK provides a programming model for application developers by providing several functionalities such as private NS descriptor (NSD)/VNF descriptor (VNFD) catalogue, editor, validator, emulator, service monitoring and profiling tools, which allow defining complex media services consisting of multiple VNFs. 5G-MEDIA overall SDK architecture is given in Figure 1. As demonstrated in this figure, the first step is to validate the descriptors via Validator. If the package is valid, then it is imported onto the Catalogue. Then descriptor which is imported into the catalogue can be instantiated through the Editor.

A. All-in-one UI

Since 5G-MEDIA SDK has a set of tools with several different functionalities, collecting all of them into one interface is helpful for the developers to control the system quickly. The interface consists of a set of navigation items which give fast access to each SDK tool. The proposed interface is a web based GUI embedded into a desktop container (a.k.a desktop application) which can be used by the developers on their personal computers. The interface has links which open each SDK tool as shown in Figure 2. Hereby, the developer can easily use any tool of the SVP (i.e Open Source Management and Orchestration (OSM) CLI [7], Lean OW CLI [8]) by the interface without searching the starter file of the tool. On the other hand, a CLI might be needed for some use cases. Therefore, the navigators of the interface can be modified to execute an external CLI application, e.g. PuTTY, Git, etc. Another advantage of the all-in-one interface is to modify the parameters of external application. By using a configuration file, the interface can forward the external application to a related path or access the related server without entering the extra information for each time and thus provides flexibility to developers. The proposed interface is build using two platforms, i.e. Electron [9] and React [10]. React, which is one of Facebook's first open source projects, is a JavaScript library for building UIs. First, encapsulated components that manage their own state are built, and then they are composed to make complex UIs. As a web-based framework, React increases response speed of the all-in-one UI. However, due to the usage of the SDK toolsets, the UI needs to open an external application from personal computer of the developer. Because of the security level, web-based frameworks like React cannot to open the external application from local; for this reason, Electron is used in combination with React to overcome this problem.

B. Editor

The 5G-MEDIA Editor is a web-based application whose frontend is running on the browser of the media application developer. In fact, the editor runs on developer's local OSM environment so that the developer can manage emulation environments such as vim-emu or lean-OW. It is the main UI for adding new data centres/virtualized infrastructure managers (VIM), designing, validating and onboarding media applications, VNF or NS to the private catalogue by building VNF forwarding graphs (VNF-FG), instantiating or shutting down a NS deployed in emulated data centres. UI also visualizes the topological dependencies or interconnection of involved

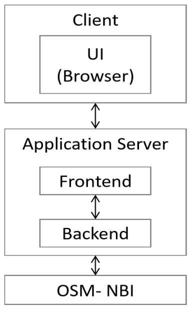

Fig. 3. 5G-MEDIA Editor Architecture

VNFs of VNF-FG and descriptions of individual VNFs. This interface can also be used to conveniently manage lifecycle operations on VNFs and Network Services (NS) such as instantiation or termination of them. 5G-MEDIA Editor is split into a backend and a frontend part that can be executed by different servers to improve its scalability. Figure 3 shows this modularization and how the components can be split between backend and frontend module. In this setup the application logic is encapsulated in the backend module and exposed via a RESTful interface to the frontend parts. The frontend components are executed in the client's browser. The backend module also interfaces with the OSM system via the Northbound REST API based on ETSI NFV-SOL-005 [11].

1) Specifications for FaaS UI: A FaaS button is placed in the composer of the OSM UI. Therefore the developer can list the FaaS VNFs pressing this button and can edit them using the editor. The FaaS VNFs are uploaded on the OW using the update OW API. The UI also includes a button for FaaS accounts under the account sections like SDN or VIM accounts. Hence, the developers are enabled to store access credentials of OW through this menu. These access credentials are used for authorization in any OW API calls related to FaaS specific configurations.

C. Validator

In 5G-MEDIA, media applications (which are NSs) are the chains of one or more VNFs, the orchestration of which realizes the desirable end-to-end functionality. Each chain is constructed and maintained by the Service Orchestrator (SO) using the information which is included in the corresponding NSD. This information can consist of the following

- The VNFs which are composing the service chain
- The Virtual Link Descriptors (VLDs) which describe the resource requirements needed for links between VNFs and endpoints of the network service.
- VNF-FG information which determines the traffic flow and behaviour over the service chain.

In addition to a set of NSD and VNFDs, a 5G-MEDIA service is also characterised by a package descriptor for the overall service. Package descriptor is generally a compressed file with a tar.gz extension which includes VNFD and NSD descriptor files. Each of these descriptors follow a yaml schema language format. 5G-MEDIA Validation tools have been developed

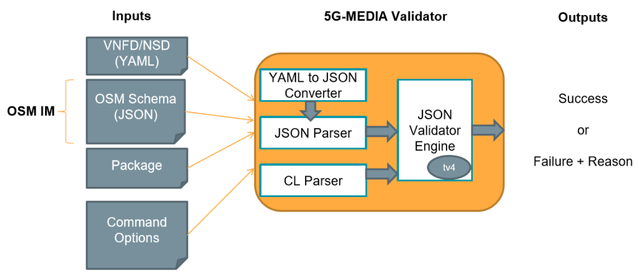

Fig. 4. 5G-MEDIA Validator Architecture

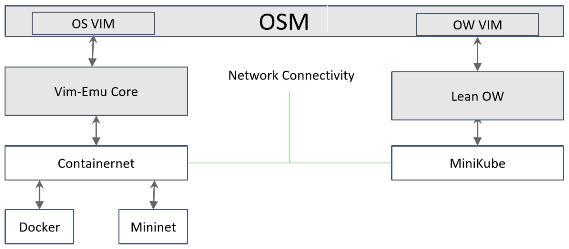

Fig. 5. 5G-MEDIA Emulator High Level Design

in order to check if given descriptors are compliant with 5G-MEDIA schema format. The architecture of 5G-MEDIA validator is given in Figure 4. The inputs of the validator are specified as follows

- *NSD/VNFD yaml files:* NSD files can be validated with a path where referred VNFD files are located.
- *JSON Schema File:* Schema file based on OSM information model is considered.
- *CL Parser:* Node.js opt [12] is considered

D. Emulator

The 5G-MEDIA emulator mimics the architecture presented in [13]. Specifically, it represents a low footprint of SVP with OSM being the entry point for the emulator and OpenStack and FaaS VIMs managing all-in-one development distros of OpenStack and Apache OpenWhisk, respectively. This approach results in both realistic and resource efficient emulation environments. The 5G-MEDIA Emulator supports media application developers to locally prototype and test their NSs in realistic end-to-end multi Point-of-Presence (PoP) scenarios. This platform allows the execution of real network functions, packaged as Docker containers, in emulated network topologies running locally on the developer's machine. The emulation platform not only offers OpenStack-like APIs for each emulated PoP but also provides OpenWhisk APIs via Lean OW, which can therefore be installed to developers personal computer. 5G-MEDIA Emulator also integrates with OSM, which is responsible for deploying and managing the NSs which are tested in an emulated environment. 5G-MEDIA Emulator leverages vim emulator (a.k.a vim-emu/son-emu [14]) and Lean OW to provide an emulated network in the developers environment. The design of vim-emu is based on Containernet [15] which extends the Mininet emulation framework [16]. Furthermore, Containernet supports using standard Docker containers as VNFs within the emulated network. However, lean-ow uses Minikube [17] which is a Kubernetes distribution used as a standard development environment on a single machine. Minikube runs a single-node Kubernetes cluster inside a VM on developers' laptop. As presented in Figure 5, Containernet support using standard Docker containers as VNFs within the emulated network. Furthermore, it also allows adding and removing containers from the emulated network at runtime while supporting the use of the emulator like a cloud infrastructure in which we can start and stop compute resources (in the form of containers) at any point.

1) Specifications of FaaS Emulation: The FaaS emulator which compromises Lean OW and Minikube is not fundamentally different from the SVP, which uses a full clustered installation of Apache OpenWhisk and of a K8s cluster. A FaaS VNF that is to be emulated, should be pre-onboarded into Lean OW in a regular way using a wskdeploy tool . This tool allows to define the OpenWhisk action that implements this VNF as Next, a VNFD should be defined for the VNF and onboarded to the catalog via OSM. Again, this process is not different from that of the regular onboarding. Finally, the VNFD should be instantiated using OSM. The VNF instance and its status are shown in the OSM GUI. The instance can be terminated from OSM on the users discretion as if it was a regular VNF instance.

E. Service Monitoring

Virtualisation techniques such as VMs or containers encounter a significantly growing interest as telecom services which are previously managed by hardware applications now increasingly experience the softwarification [18]. However, these containerization and virtualization techniques require different kind of customizations or configurations. Therefore, they can introduce bugs in the media application definitions, which can slow down the application or network service provisioning time. In order to avoid these bugs and decrease the time required between the development process and the operations side of a developed media application, 5G-MEDIA SDK provides monitoring tools that allow rapid testing and verification of any modified parameter.

The 5G-MEDIA SDK has a set of monitoring tools available for media application developers that can gather and centralize monitored metrics into a local database. Metrics can be queried from either VNFs deployed in the emulator or in the Media Service MAPE of SVP. After the metric data is stored in the local database, further analysis can take place to debug or optimize the performance of the monitored VNF or service. Figure 6 shows the detailed flow of the monitoring tools. Container monitoring tool in emulator (i.e cAdvisor) gathers metrics related to the compute, storage or network and these metrics are exported by starting a query scheduling. This scheduling involves an SDN controller which uses OpenFlow protocol to inquire the packet and byte counters of the virtual network interfaces of the emulated service. In addition to the

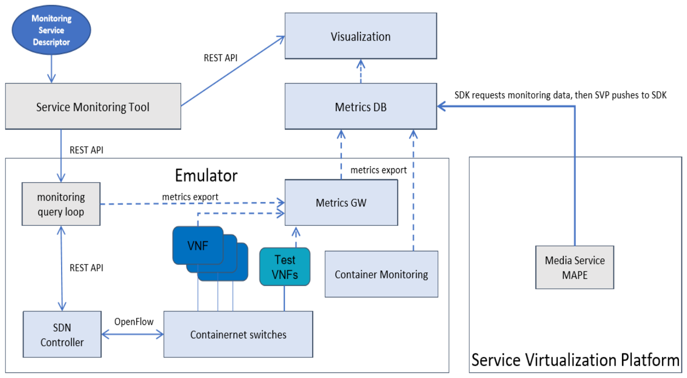

Fig. 6. Service Monitoring Tools High Level Design

interface counters, some particular flow counters can be installed so that a better, fine-grained network traffic monitoring can be provided. Monitoring metrics of the emulator are then pushed to a Metrics Gateway (i.e Prometheus Push Gateway) from where the metrics are pulled by the external Metrics Database (Prometheus database) in the SDK. The emulator has a REST API to control the export of metrics to the Metrics Gateway. This can also be controlled from a Monitoring Service Descriptor (MSD) file that lists all needed metrics. The gathered metrics can be visualized using the visualization tool (i.e Grafana GUI) which visualizes the inquired metrics from the Metrics Database using a web-based GUI. The required metrics for virtualisation and the combination of these metrics on different graphs are described by a Monitoring Service Descriptor (MSD) file. The Service Monitoring Tool CLI then prepares the Virtualisation dashboard by parsing this MSD file.

In 5G-MEDIA SDK, the FaaS VNFs monitoring is fully aligned with the rest of the architecture. Instantiation of the FaaS VNFs via the OSM FaaS Plugin causes VNFs to be offloaded to Minikube (an all-in-one Kubernetes) by Lean OW. Similarly to the monitoring architecture of SVP, in SDK, we leverage the native Prometheus monitoring framework that obtains metrics from cAdvisor (the native metric producer of the Kubernetes) and stores them in the Prometheus TSDB. Grafana dashboards are used to query Prometheus TSDB to obtain compound metrics related to the FaaS VNFs execution.

F. Profiling

5G-MEDIA profiling tool supports load testing under various constraints on NSs which are deployed in the emulator platform. During these tests, a variety of metrics can be monitored and this can help service developers find bugs, detect bottlenecks or investigate problems in their media applications. Profiling tool aims to automate big parts of this workflow and thus support service developers as much as possible. To achieve this aim, 5G-MEDIA profiling tool creates a series of service packages, each with a particular resource limitation and defined functional tests. These packages can be automatically deployed on the 5G-MEDIA emulator or the SVP where the functional tests are executed while metrics are gathered.

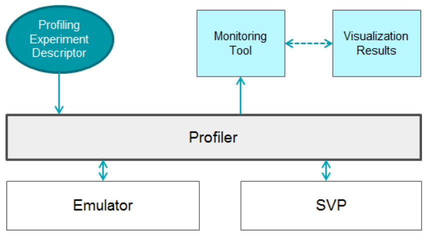

Fig. 7. Profiler logical component diagram

The logical component diagram of 5G-Media Profiling tool is presented in Fig. 7

As the profiler creates descriptors for experiment purposes which results in instantiating VNFs under specific resource allocations and measuring results, for FaaS, the developer needs to keep the same sorts of metrics as for any other VNF. As seen in Fig. 7, the user first creates a profiling experiment descriptor (PED) which consists of all necessary information for the profiling experiment. Especially, it includes a reference for the NS that should be profiled, e.g., a service package or service descriptor. Moreover, it also consists of descriptions of all service configurations which should be tested, such as different resource assignments for the VNFs that are used. A profiling experiment based on a PED is considered to be in active mode. On the other hand, profiling can also be performed on pre-deployed services and in this case it is considered in passive mode.

III. An example workflow of 5G-MEDIA SDK

Referring to Fig. 1, a typical SDK workflow dealing with a media application can be presented. The workflow comprises 6 steps, starting the creation of the VNFDs and an NSD. Then in the next step, the developer uses the validator to validate the descriptors which are exported from the private catalogue. If descriptors are validated, then the media applicatione is instantiated by using 5G-MEDIA editor with the help of a configuration microservice. After instantiation of the media application in the emulated environment completes, a traffic simulator is run to test the action on the Lean OW of 5G-MEDIA emulator. The monitoring tools support media application developers to gather and centralize monitored metrics into a local database, so developers can analyse them through a web-based dashboard. As a result, they can optimize or fine-tune performance of their applications. Furthermore, the developer may want to do some load tests under different resource constraints on NSs deployed on the emulation platform before the live environment deployment. In this case, the developer creates a series of service packages, each with a specific resource limitation and defines functional tests over the profiling tool. Such load tests are executed in the emulated environment.

IV. Conclusions

In this paper, we have presented an overview of the programming tools to efficiently develop NFV-based media applications in 5G networks. The parts of 5G-MEDIA SDK have been presented discussing the open-source frameworks and libraries that are considered for the project. The architecture and usage of the 5G-MEDIA SDK tools such as all-in-one UI, validator, editor, emulator and service monitoring have been explained in detail. Application of the serverless FaaS approach to NFV technology has been presented and the development of the appropriate programming tools to accommodate FaaS paradigm has been given.

Acknowledgment

This work has been realized in context of the 5G-MEDIA project. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 761699.

References

[1] K. Poularakis et al., "Exploiting Caching and Multicast for 5G Wireless Networks," IEEE Transactions on Wireless Communications, vol. 15, no. 4, pp. 2995-3007, April 2016. doi: 10.1109/TWC.2016.2514418.
[2] A. Tzanakaki et al., "Wireless-Optical Network Convergence: Enabling the 5G Architecture to Support Operational and End-User Services," in IEEE Communications Magazine, vol. 55, no. 10, pp. 184-192, October 2017. doi: 10.1109/MCOM.2017.1600643.
[3] S. Rizou et al., "A service platform architecture enabling programmable edge-to-cloud virtualization for the 5G Media industry," in EasyChair Preprint no. 149, 2018 doi = 10.29007/bn68.
[4] D. Breitgand, "Towards Serverless NFV for 5G Media Applications," Proc. 11th ACM Int. Systems and Storage Conf. p: 118, 2018.
[5] I. Baldini et al., "Serverless Computing: Current Trends and Open Problems," In: Chaudhary S.,Somani G., Buyya R. (eds) Research Advances in Cloud Computing. Springer, Singapore, 2017.
[6] S. Rossem et al., "A Network Service Development Kit Supporting the End-to-End Lifecycle of NFV-based Telecom Services," Proc. of IEEE Conf. on Network Function Virtualization and Software Defined Networks, 2017.
[7] Open Source NFV Management and Orchestration: https://osm.etsi.org/, Accessed on the 8th Aug, 2018.
[8] Lean Openwhisk: https://github.com/kpavel/incubator-openwhisk/tree/lean, Accessed on the 8th Aug, 2018.
[9] Electron, Build cross platform desktop apps with JavaScript, HTML, and CSS: https://electronjs.org/, Accessed on the 8th of August, 2018.
[10] React, A JavaScript library for building user interfaces: https://reactjs.org/, Accessed on the 8th Aug, 2018.
[11] ETSI, "Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; RESTful protocols specification for the Os-Ma-nfvo Reference Point," White Paper ETSI GS NFV-SOL 005 V2.4.1, 2018.
[12] Node js opt: https://www.npmjs.com/package/opt, Accessed on the 8th Aug, 2018.
[13] S. Rizou et al. "A service platform architecture enabling programmable edge-to-cloud virtualization for the 5G Media industry." 2018 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB). IEEE, 2018.
[14] T. Soenen et al., "Insights from SONATA: Implementing and integrating a microservice-based NFV service platform with a DevOps methodology," Proc. of IEEE/IFIP Network Operations and Management Symposium, pp:1 -6, 2018.
[15] M. Peuster, H. Karl, and S. v. Rossem: MeDICINE: Rapid Prototyping of Production-Ready Network Services in Multi-PoP Environments. IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Palo Alto, CA, USA, pp. 148-153. doi: 10.1109/NFV-SDN.2016.7919490. (2016).
[16] Mininet An Instant Virtual Network on your Laptop (or other PC): http://mininet.org/, Accessed on the 14th Aug, 2018.
[17] Minikube: https://kubernetes.io/docs/getting-started-guides/minikube, Accessed on the 8th Aug, 2018.
[18] S. Rossem et al., "Monitoring and debugging using an SDK for NFV-powered telecom applications," Proc. of IEEE Conf. on Network Function Virtualization and Software Defined Networks, pp:1-5, 2016.