An SDN/NFV Platform for Personal Cloud Services

Roberto Bruschi*, Franco Davoli‡, Paolo Lago*, Alfio Lombardo§, Chiara Lombardo*, Corrado Rametta§, Giovanni Schembra§

‡ DITEN - University of Genoa - Genoa, Italy
* CNIT - University of Genoa Research Unit - Genoa, Italy
§ DIEEI – University of Catania – Catania, Italy

Abstract— *In the last few years, network "softwarization" is gaining increasing popularity to achieve dynamicity and flexibility. Cloud computing, as well as the new paradigms of Software Defined Networking (SDN) and Network Functions Virtualization (NFV), are supporting this evolution. However, the need to move services closer to users to guarantee low latency in the service fruition on one hand, and the trend to support personalization of services on the other, are stimulating the migration of services toward edge nodes (in the so-called "fog computing" fashion). This is the target of the INPUT platform, proposed in the INPUT project to support Future Internet personal cloud services in a more scalable and sustainable way, and with innovative added-value capabilities. The INPUT platform enables next-generation cloud applications to go beyond classical service models, and even replaces physical Smart Devices, usually placed in users' homes (e.g., set-top-boxes, etc.), with virtual entities, providing them to users "as a Service." In this paper, we present the INPUT paradigm and discuss a relevant use case – namely, the virtual Set-Top-Box – adopted to prove the feasibility of the softwarized SDN/NFV paradigm jointly with the fog-computing approach for the support of personal cloud services. The INPUT platform is also compared with a legacy approach to evaluate the gain in terms of quality of experience (QoE) for both static and mobile users.*

Index Terms— **SDN, NFV, Network Softwarization, Personal Cloud Services, Fog Computing, Service Chains, Home Entertainment.**

I. INTRODUCTION

In the last years, new ways of interaction, mainly driven by the diffusion of new technological opportunities like social networks and smart devices, have contributed to a fast-paced shift of the users' demands towards the hyper-connectivity paradigm. Apart from being a huge phenomenon from the social science point of view, this request for multiple means of communication is also affecting the evolution of the Internet infrastructure. For example, the diffusion of "smart" devices, like smartphones and tablets, raises some new, special requirements. In fact, contrarily to computers that have traditionally been used to communicate through the Internet, such devices are characterized by limited storage and computational capabilities. These characteristics have led to the development of a significant number of cloud services to provide the needed level of support.

Regarding cloud computing infrastructures, state-of-the-art solutions allow mobile smart devices to access the required computing and storage resources by directing service requests to remote datacenters in the Internet. However, in order to satisfy the low latency levels and bandwidth requirements posed by next generation cloud services, like conversational applications, online gaming, interactive distance learning, video surveillance, etc., there is the need of deeply re-thinking networking and Information Technology (IT) solutions in a holistic fashion [1-4].

Following this line of reasoning, the next frontier to reduce the end-to-end network latency will clearly consist of hosting cloud applications directly into network operators' infrastructures, and of being as close to end users as possible [5-6].

In particular, Software Defined Networking (SDN) [7-8] and Network Functions Virtualization (NFV) [9-10] offer a new opportunity towards a flexible softwarized network paradigm, which overcomes the inadequate and obsolete traditional switching implementations. The definition of new management strategies for these new networking paradigms has become a challenging activity for both Academia and IT industries. Another approach aimed at reducing latency between users and computing resources while offloading mobile applications is the Mobile Cloud Network (MCN) [11-13].

In this perspective, starting from the experience achieved by the authors within INPUT [14], a research project funded by the European Commission under the Horizon 2020 program, this paper aims to propose novel infrastructural solutions to support Future Internet personal cloud services in a more scalable and sustainable way and with innovative added-value capabilities.

To this purpose, this paper presents the INPUT platform, a framework that, leveraging on both fog computing [15-16] and the two networking paradigms of SDN and NFV, allows going beyond classical cloud-computing and mobile cloud network service models, introducing the new concept of *personal network*. This is an extension of the user's home Local Area Network (LAN), achieved by connecting not only physical devices, but also virtual Smart Devices, i.e. virtual images that are able to replace physical Smart Devices (SDs) usually located in users' homes (e.g., set-top-boxes, etc.), and providing them "as a Service" (SDaaS).

The INPUT platform shares this perspective with cloud computing [17], which has been conceived in the view of supporting the Future Internet [18] through the personalization of services; however, the proposed approach overcomes the typical delay problems related to the use of remote data servers, as employed by standard over-the-top providers, by moving cloud computing and network services closer to the edge network nodes.

The potential innovation of the INPUT approach with respect to other previous research initiatives is related to the new dimension of personalization in the services and virtualization of network entities. Moreover, INPUT does not work on proprietary hardware to implement cloud services close to the network edge, but it proposes to use open source configurable devices as edge routers, which can be managed according to an in-network programmability approach. Finally, thanks to the integration of NFV and fog-computing technologies, it is possible to move the complexity of processing and data collection closer to the user, and therefore to "personalize the network to individual users", also taking into account their specific characteristics of traffic generation and mobility, as well as their requirements in terms of Quality of Service (QoS) and Quality of Experience (QoE).

Through SDN and NFV technologies, the network can be fully programmable, in the sense that traffic paths can be customized by software, and any network process can be removed from the device and made running (with custom patches) as application in a remote server [19].

In this paper, we describe the management approach employed in the proposed INPUT platform, and identify the driving forces to flexible and efficient network function deployment explored in its definition. Moreover, we illustrate a relevant use case – namely, the virtual Set-Top Box (vSTB), which embraces some of the NFV use cases defined in [20]. In particular, by focusing on virtualization of home appliances, the vSTB considers ETSI's Use Case 7 on the "Virtualization of the Home Environment", which is also compliant with the DLNA standard [21]. The vSTB also follows the philosophy of ETSI's Use Case 8 on "Virtualization of CDNs", by moving the content streams out of compute/storage nodes closer to the end user, so saving network links and equipment, and allowing to reliably deliver streams with larger bandwidth. However, let us stress that the vSTB, thanks to the underlying INPUT infrastructure, outdistances the above concepts of home environment and Content Delivery Network (CDN) virtualization [22-23], as well as live and stored video streaming through peer-to-peer (P2P) networks [24-26]. In fact, based on the new concept of personal network introduced by the INPUT project, it is able to provide users with a STB service that allows them to receive both stored videos and live streaming with very low latencies, even in mobility.

The rest of the paper is organized as follows. In Section II the INPUT framework is detailed, with a description of both control and data planes. Section III presents the vSTB use case together with its architectural aspects. Section IV introduces a numerical analysis to compare the INPUT approach with legacy live and stored content delivery service in terms of QoE. Finally, in Section V some conclusions are drawn.

II. The INPUT Framework

The INPUT framework presented in this section has the goal of realizing the fog-computing paradigm, by introducing and controlling computing and storage capabilities in the Edge Network. In this respect, the SDN and NFV paradigms are exploited to fit the Future Internet sustainability and scalability requirements: NFV provides the architectural solutions deployed in the Access and Edge networks, especially for application-level services and network-specific functions; SDN allows supporting the control functionalities needed to introduce computing and storage capabilities to edge network devices, and consequently to move cloud services closer to end-users.

This technological approach will promote flexibility and foster the introduction of new services otherwise unfeasible, such as the virtualization of physical SDs – like storage servers, set-top boxes, video recorders, home-automation control units, game consoles, among others. In this paper, we consider the example of a *virtual Smart Device* (*vSD*), which can be used as a SD-as-a-Service (SDaaS), and is able to replace the relevant physical device in the users' home networks. These *vSDs* will be made available to users at any time and at any place, by means of virtual cloud-powered Personal Networks, which will provide users with the perception of always being connected to their home network, independently of their location.

A. Deployment of Personal Cloud Services

A *Personal Network* is a secure and trusted virtual overlay network capable of interconnecting user's SDs (either virtual or physical) with standard layer-2 (L2) protocols and operations equivalent to the ones presently available in the home network of users, independently of their location (inside/outside the user's home) or nature (physical/virtual). In other words, a Personal Network is an extension of the user's physical home network, also including vSDs running in the Telco Operator network, with the possibility of "following" the user, whatever his/her geographical location (inside/outside the user's home).

An example of deployment of a Personal Network is shown in Fig. 1. It is realized by virtualizing both the user's physical home gateway, by replacing it with a virtual home gateway (*vHGW*), and user's physical SDs, by replacing them with *vSDs*. Both *vHGW* and *vSDs* are launched as software instances in commodity computing facilities deployed at the edge of the Telco Operator network.

The vHGW is realized by transferring the network functions that are typically provided by the user's home gateway into software instances, called *Net_Functions*. A *Net_Function* can replace either a single data- or control-plane network functionality (e.g., IP forwarding/routing, firewall, DPI, NAT, DHCP), or a structured network service, constituted by a chain of elementary network functionalities.

A vSD, on the other hand, is obtained as a personal cloud service realized as a chain of one or more User applications (*User_Apps*), Service applications (*Service_Apps*) and Data Center applications (*DC_Apps*), each performing a specific task (e.g., user interface, web server, proxy, content caching, storage, computing, encryption/decoding, etc.). A *Service_App* is a software instance running in a single "execution container" (e.g., a virtual machine) at the edge network, and provides application level services. *User_Apps* run on physical user clients (e.g., smartphones and tablets), and provide users with an interface to access personal cloud services. Finally, *DC_Apps*, running on a remote data center, aim at completing the service with high-computational- and storage-consuming-

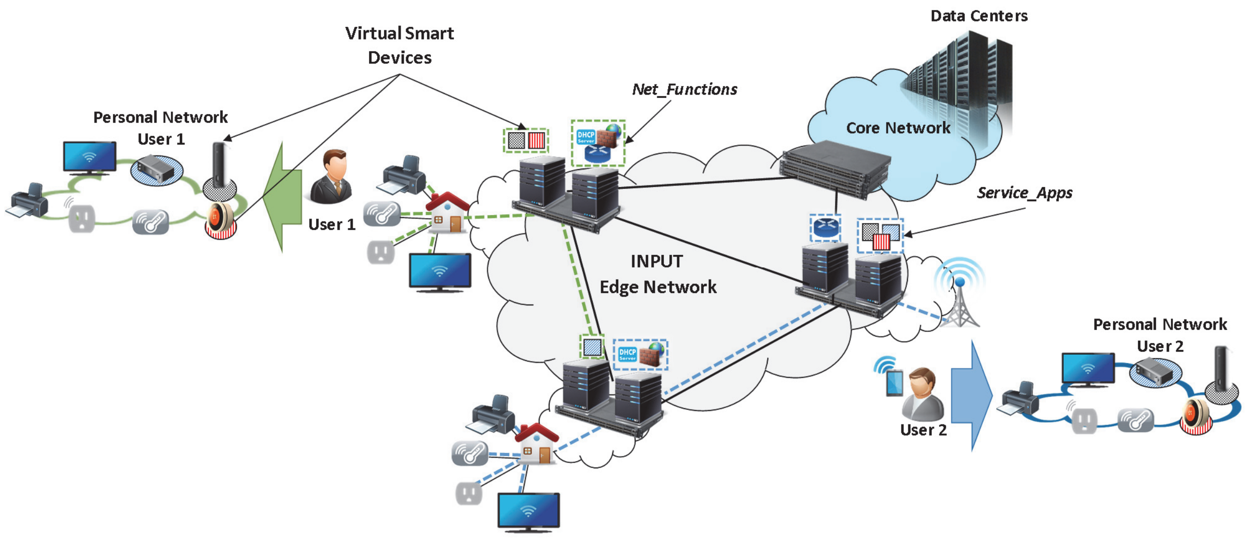


Figure 1: Mapping of the User's Personal Network onto the INPUT Infrastructure.

capabilities.

This modularity allows performing a more fine-grained management of the computing and storage resources and of the QoS/QoE according to the user location, and enables personal cloud service providers to easily upgrade/extend the functionalities of the vSDs. *Service_Apps* and *Net_Functions* can be dynamically migrated from a computing facility to another while guaranteeing service continuity. The purpose of the migration process is twofold: on the one hand, it can be used to put under-utilized servers in low-power idle or standby states, hence reducing the carbon footprint produced by the network operators; on the other hand, it allows the placement of the service chain as physically closer to the user position as possible, in order to reduce the end-to-end latency, and thus improve the overall QoE.

Fig. 2 presents an example of instantiation of a personal cloud service realized with a vSD constituted by the chain of three *Service_Apps*. The *User_App* communicates directly with the *Service_App* 1 connected to the user's Personal Network. This *Service_App* is in charge of realizing the access interface to the personal cloud service. Other service components and functions are provided by further *Service_Apps* (*Service_App* 2 and *Service_App* 3 in the example of Fig. 2), or even by remote *DC_Apps,* which in general can be shared with other services/users for scalability purposes and/or for the nature of the service (e.g., content caching).

The communication and information exchanged among different *Service_Apps* of the same personal cloud service are handled through a Back-End Network (see Fig. 2), i.e. a set of virtual L2/L3 interconnections built between two or more virtual/physical hosts, whose traffic is isolated from the one carried by the Personal Network or other Back-End Networks.

B. *INPUT Functional Blocks and Interfaces*

The most relevant functional blocks of the network control plane and the data path components composing the INPUT architecture, along with the main interfaces towards the stakeholders, are shown in Fig. 3. The INPUT project considers the presence of three main stakeholders: Telco Operator, Cloud Service Providers and End Users. The end users purchase services from one or more service providers, which in turn rent portions of computational and storage resources from the telco operator to deploy the virtual instances of the personal services.

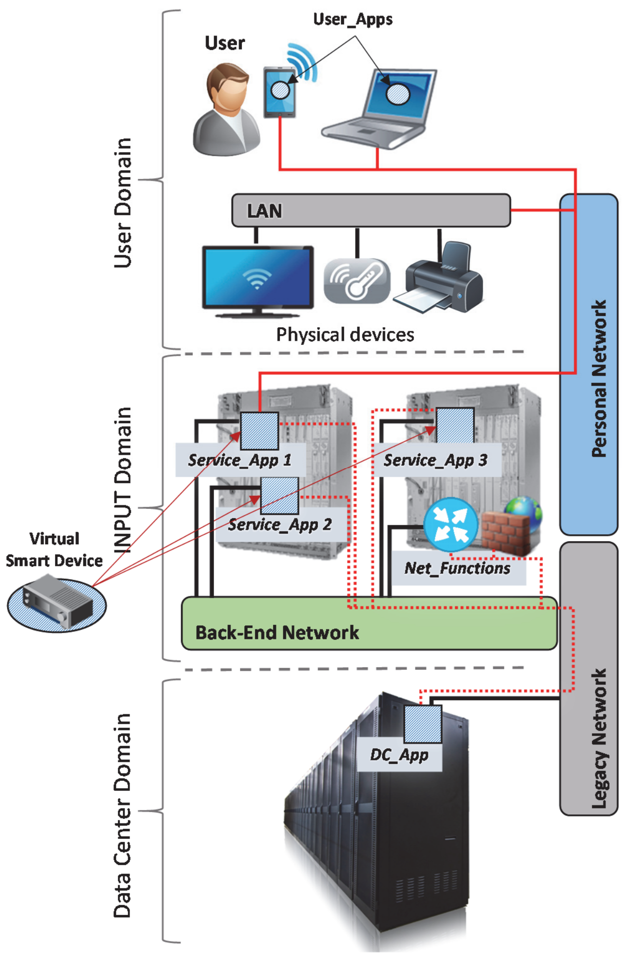


Figure 2: Example of a service deployment in INPUT.

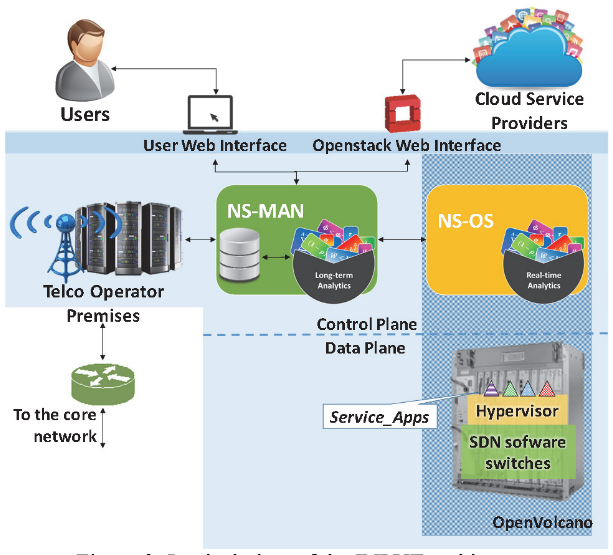


Figure 3: Logical view of the INPUT architecture.

The OpenStack APIs [27] are exploited to provide a trusted and well-known interface to the service providers, enriched with extensions to isolate the physical resources, while guaranteeing the desired levels of QoS/QoE. At the end-user side, a web interface has been designed to provide all the functionalities typically available in a home gateway.

The reference architecture for INPUT is represented by the Open Virtualization Operating Layer for Cloud/fog Advanced NetwOrks (OpenVolcano) [28], an open-source software platform for fog computing, which exploits in-network programmability capabilities for off-loading, virtualization and monitoring. From the physical point of view, the distributed OpenVolcano platform is composed of a number of OpenFlow switches interconnecting a number of computing and storage facilities spread across the Telco infrastructure. As shown in Fig. 3, OpenVolcano covers the data plane and control plane functionalities related to real-time analytics (NS-OS), while decisions on the long-term are provided by an external controller (NS-MAN).

In more detail, the *NS-MAN* is responsible for the long-term configuration of the network, the administrative configuration of the infrastructure, the management of overlaying personal cloud services and Personal Networks. It also has the task of monitoring the resource usage and power consumption of the overall NFV Infrastructure (NFVI), and of reserving/managing/releasing networking and computing resources to properly satisfy bandwidth and quality levels required by the different personal cloud services. Finally, it is in charge of monitoring faults in the system and, by using trend analysis, predicting errors and guaranteeing constant availability of the deployed services. In order to provide these functionalities, it supports big data storage and analytics algorithms to select the proper policy for the optimization of the resource usage. This is achieved by tracing, storing, and correlating the network and computing available resources, the actual user service requirements, and the overall (end-to-end or hop-by-hop) cloud service needs. The data analytics task, on the other hand, allows predicting service demand, planning resource provisioning, preventing congestion and possible failures, and maximizing energy saving.

The *NS-OS* performs the tasks of consolidation, orchestration, and monitoring. This goal is achieved by driving the real-time configuration of the programmable resources and the dynamic instantiation and migration of *Service_Apps* and *Net_Functions* according to the users' locations. The *consolidation* task regards the calculation of the optimal re-configuration of the infrastructure (e.g., the mapping of a Personal Network onto the physical network and the packet matching and action rules to be configured in the SDN switches that are present in the underlying NFVI), in terms of both network paths/overlays and *Service_Apps*/*Net_Functions* locations, with the objective of meeting the required QoE/QoS with the minimum possible level of energy consumption, according to a short-term estimation of workload/traffic volumes. The *orchestration* mechanism takes the re-configured setup coming from the consolidation process as an input, and instantiates/migrates *Service_Apps* and *Net_Functions* to the identified subset of device/hardware resources, by changing the network configuration accordingly, avoiding any service interruption or performance degradation. Finally, the *monitoring* task collects performance measurements and alert messages from the networking and computing facilities of the NFVI, including power-aware performance indexes, like infrastructure- and device-level power consumption, end-to-end latency, and user mobility statistics.

Regarding the data plane, OpenVolcano is in charge of running the software instances providing both L7 services (*Service_Apps*) and *Net_Functions*. To this purpose, each server deployed in INPUT hosts a high-performance OpenFlow software switch based on the DPDK [29] libraries to realize the functionalities of packet capture, buffering and scheduling. These functionalities allow properly exploiting hardware parallelism, which represents a well-known issue in multi-core processors, as discussed in [30]. The internal OpenFlow switch directs traffic flows to *Service_Apps* and *Net_Functions* according to the packet matching/action rules configured by the NS-OS.

Each *Service_App* composing a service chain is implemented as a VM (IaaS services) or a Java VM (PaaS services), and seamless migration is made available for single VMs, as well as in bulk for entire service chains, depending on the consolidation requirements [31]. Regarding *Net_Functions*, as they are considerably "thinner" than *Service_Apps*, the use of VMs would represent a waste of memory and computational overhead. For this reason, and to improve migration efficiency, they are implemented as processes/threads running on the host operating system.

Finally, a *Hypervisor*, controlled by the NS-OS through the libvirt APIs manages allocation and functions' migration.

III. Virtual Set-Top Box Personal Cloud Service

The target of this section is to show how the proposed INPUT framework can be used to provide customers with a personal cloud service regarding multimedia entertainment at home. It consists of the virtualization of a set-top box (STB) device like the physical one provided today by content providers (CPs) that offer services like pay-per-view of live video streaming and video on demand. Thanks to the SDaaS facility offered by the INPUT framework, the STB is not physically present at the

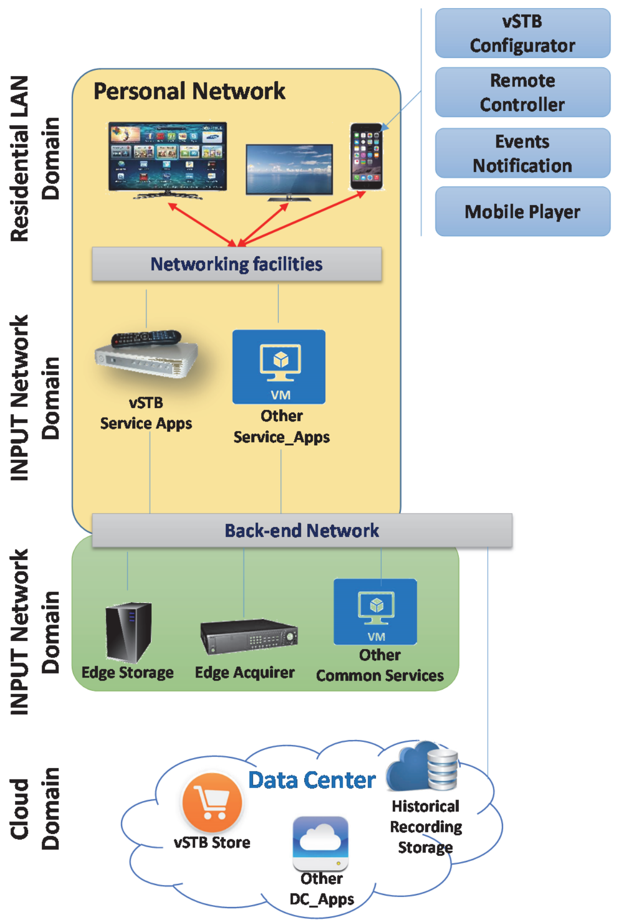


Figure 4: Personal Network when the vSTB service is active.

user's home, but is provided as a vSTB through the OpenVolcano infrastructure.

The vSTB considered in this use case provides the following main capabilities:

1. real-time streaming of multimedia content transmitted from a CP to one or more user home player devices, such as smart TVs, and to one or more user mobile terminals, such as smartphones and tablets;
2. recording of multimedia contents;
3. playout of multimedia contents previously recorded.

These capabilities make the difference between the vSTB and a legacy CDN application. In fact, while a CDN aims at delivering stored video streams by caching them close to the users, the vSTB yields a service like the one provided by a STB, also including live transmission from remote content providers. Only if users decide to record some flow, as we use to do at home with our physical set-top boxes or video recorders, live events are recorded as close to the users as possible, but this is done based on a user's request.

The potential of the STB virtualization becomes evident when the user leaves the user's own home network. In fact, in this case, the user can still exploit the vSTB personal cloud service by means of mobile terminals to watch real-time and stored contents while in mobility.

Fig. 4 shows a high-level overview of the user's Personal Network when the vSTB service is active. As already specified, the Personal Network includes both physical and virtual devices. In more detail, we can distinguish between physical devices at the user side, i.e. in the user's *residential LAN domain*, and virtual devices in the *INPUT domain*, i.e. deployed inside the INPUT platform, but belonging to the user's private domain. In addition to the above components, there are further elements deployed within the network infrastructure in the INPUT platform, which can be considered as lying in the *shared domain*, because they are effectively shared among a set of users, i.e. not belonging to any specific Personal Network.

A vSTB personal cloud service for a given user is realized by integrating the following elements within the same Personal Network:

- user's smart TVs that are assumed to be connected to the physical home LAN of the considered user;
- user's smartphones, tablets and personal computers – in the following referred to as *Smart Clients* – that can be connected to the Internet through either the physical home LAN or a cellular connection provided by the same Telco Operator;
- the vSTB element, realized as a chain of some *User_Apps* running on the user's Smart Clients, some *Service_Apps* running on the server closest to the users's home LAN, and some *DC_Apps* running in a datacenter.

A vSTB instance is created on demand by each interested user. More specifically, a user that wants to activate vSTB logs in the user's virtual Home Gateway via web or through a specific *User_App* installed on the user's smartphone, and selects the required vSTB service. Immediately after that action, the virtual Home Gateway notifies the NS-OS element residing in the OpenVolcano platform of this request, and the service chain needed to realize the vSTB is instantiated on the node of the OpenVolcano infrastructure that is the closest to the user. The NS-OS can implement different placement policies, but this is out of the scope of this paper.

All the vSTB capabilities are managed by *User_Apps* running on the user's Smart Clients. More specifically, a graphic user interface (GUI) running in a specific *User_App* called *Remote_Controller* guides the user across the above-mentioned vSTB functions, allowing the choice of both the action (i.e. playing live contents, recording a live content, and playout of users' contents) and the target device where to finalize the action (i.e. the player, among the devices connected to the Personal Network).

The next subsections will describe the functional architecture of a vSTB and the related service chains.

A. vSTB Architecture

The INPUT platform elements that compose a vSTB, organized as in Fig. 5 according to their typology, are:

User_Apps

- *vSTB Configurator*: it is the mobile *User_App* enabling the user to access the configuration menus of the vSTB, interacting with the vSTB Manager *Service_App* described below.

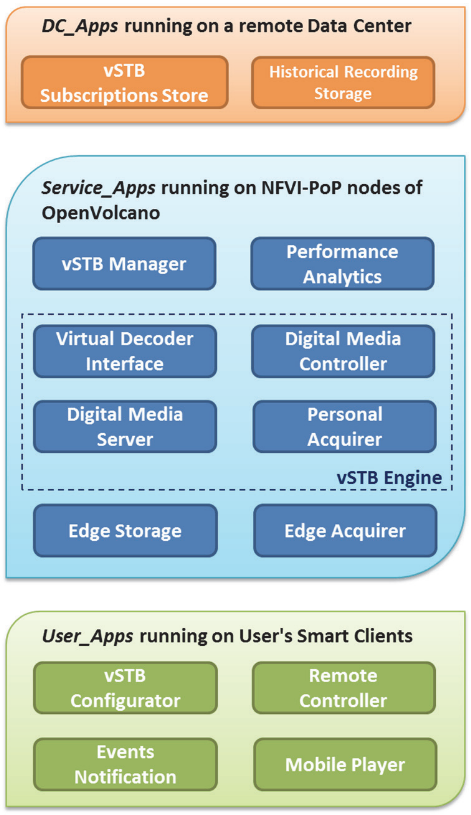


Figure 5: *DC_App*, *Service_Apps* and *User_Apps* organized according to their typology.

- *Remote Controller*: it is a mobile application that the user installs on his/her Smart Client to remotely control the vSTB. It is connected to the VDI *Service_App* to access the CP content schedule, select a real-time or recorded content, and create virtual connections between the vSTB and the DLNA-compliant physical players that are available in the home LAN.
- *Event Notification*: it allows setting up a notification service to alert users when a requested content is available.
- *Mobile Player*: it allows viewing content directly on a smartphone or tablet; it connects directly to the PA and the *Digital Media Server* (DMS) *Service_Apps*, in order to choose a real-time or a stored content.

The above *User_Apps* have been implemented in an Android app called *Ushare*. It allows the User to choose the specific content among the ones transmitted by a selected Content Provider. The content can be reproduced on both the same smart client, and/or any smart TV in the home LAN. In this last case, thanks to the DLNA standard it is possible to control the playout (volume, pause, etc.) from both the *User_App* running on the smart client, and the physical remote controller of the smart TV.

Service_Apps

- *vSTB Manager*: it enables management and configuration of the vSTB device. It is the peer entity that communicates with the vSTB Configurator to modify all the vSTB settings (e.g., the menu and subtitle language, supported encoders, video quality, etc.).

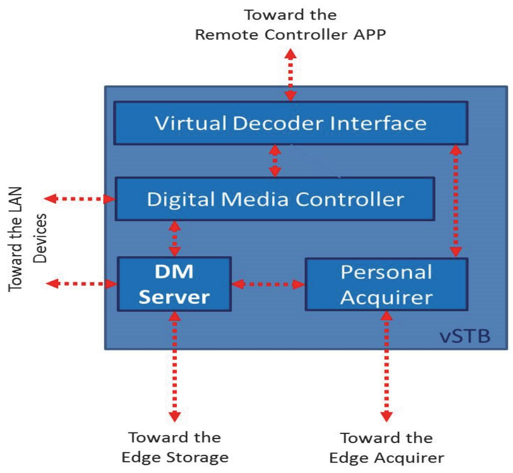


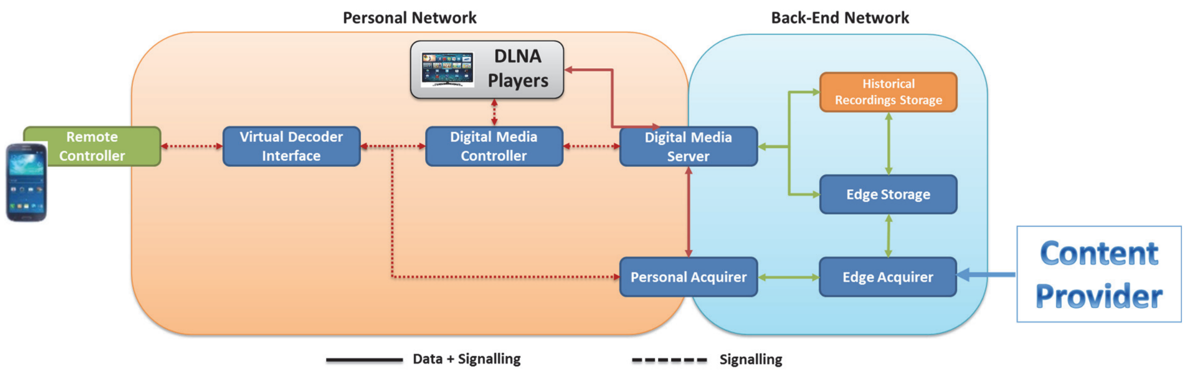
Figure 6: Virtual vSTB Engine architecture.

Moreover, according to the required video quality, and taking into account the peculiarities of the Smart Device and the link bandwidth, it interacts with the Digital Media Server to set the transcoding parameters, that is, the output encoding standard and encoding quality.
- *Performance Analytics*: it gathers the information about the service Key Performance Indicators (KPI) such as latency, delay, delay jitter, packet loss rate, throughput and other parameters allowing the measurements of the user's QoS/QoE.
- *vSTB Engine*: it is the core element of the vSTB, and represents the virtualization of the common physical STB device; it consists of four *Service_Apps*, whose functions will be detailed later:
  - *Virtual Decoder Interface* (*VDI*);
  - *Digital Media Controller* (*DMC*);
  - *Digital Media Server* (*DMS*);
  - *Personal Acquirer* (*PA*).
- *Edge Storage*: it belongs to the shared domain, and is in charge of saving the recently recorded contents.
- *Edge Acquirer*: it is the *Service_App* receiving the data flows from the CP through the dedicated Back-End network. It communicates with the PAs of all the users registered to the server where it is running, in order to publish the content timetable and transmit them the requested multimedia data flows.

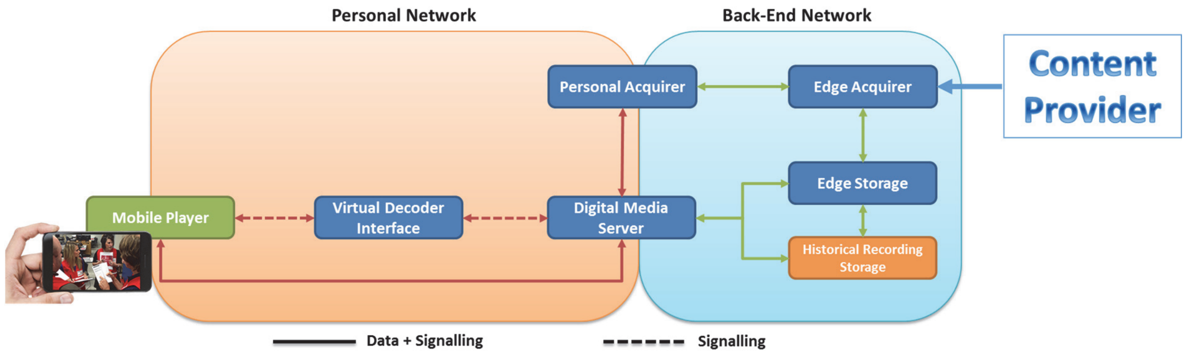
DC_Apps

- *vSTB Store:* it is a cloud service enabling the subscription of services/channels/events proposed by the CP. A user connects to this service every time he/she wants to change the terms of his/her subscription.
- *Historical Recording Storage:* contents saved by users are stored in a storage service within the Telco Operator's premises. When contents become obsolete and, thus, requested less frequently, they are sent to this *DC_App* to be stored in its storage system.

The functional architecture of the vSTB Engine is represented in Fig. 6, where the internal interactions are also depicted. It is compliant with the DLNA standard [21], that is,



(a) Service chain of the action "Watching a content on DLNA Player"



(b) Service chain of the action "Watching a content on a mobile device"

Figure 7: vSTB Service chains.

it is able to communicate with other DLNA-compliant devices like smart TVs connected to the same personal network. It consists of four *Service_Apps*:

- *Virtual Decoder Interface* (*VDI*): this component represents the interface between the Remote Controller *User_App* and the functional components of the virtual decoder. By using the Remote Controller on the user's own Smart Client (Android, iOS, Windows Phone or other frameworks), the user can select an action among watching live contents, scheduling a content recording, and playing a recorded content from the user's own digital library distributed among the Edge Storage running close to the user's location, and the Historical Recording Storage in the data center.
  Furthermore, the VDI communicates with the Digital Media Controller to select the digital media player that the user wants to employ to watch the multimedia content, and with the Personal Acquirer, with the objective of either selecting one of the available contents to be played out, or enabling the recording of a selected event.

- *Digital Media Controller* (*DMC*): according to the DLNA standard, the DMC maintains the list of the DLNA players in the personal network, and communicates with the DMS to know the list of the stored contents.

- *Digital Media Server* (*DMS*): this element is DLNA-compliant, too; it maintains and updates the list of contents exposed by both the PA and the personal digital library in the Edge Storage. Moreover, according to the encoding requirements received by the vSTB Manager, it performs transcoding to adapt the multimedia flows to the players. Let us stress that this is one of the peculiarities that make the difference between the INPUT approach, which allows the Telco Operator to customize the quality for each user, thanks to the adoption of the fog-computing paradigm, and the over-the-top approach, which performs computation on remote clouds, therefore working on classes of users.

- *Personal Acquirer* (*PA*): it receives commands from the VDI to enable the live streaming of a content or the recording of a selected content. It communicates with the Edge Acquirer in the common domain, requiring the multimedia content selected by the user or indicating that a specified event has to be forwarded to the Edge Storage to be recorded.

B. Service Chains of the vSTB

In this section, we present the service chains of *DC_Apps*, *Service_Apps* and *User_Apps* involved during two relevant actions characterizing the considered personal cloud service, that is, watching a content (either live or recorded) on a smart TV or on a mobile device. The *Service_Apps* involved in the first action (refer to Fig. 7.a) are the *Mobile Player User_App*, the *Digital Media Server* and the *Personal Acquirer*, coupled with the *Edge Acquirer Service_Apps*, as concerning the view of a live content from the content provider; the *Digital Media Server* and the *Edge Storage Service_App* or the *Historical Recording Storage DC_App*, as far as the view of a stored content is concerned.

More specifically, as shown in Fig. 7.a, a user that wants to watch a multimedia live content, accesses the *Virtual Decoder Interface* by mean of the user's *User_App Remote Controller*. The VDI *Service_App* is connected to the *Digital Media Controller* and to the *Personal Acquirer Service_Apps*. Thanks to them, it is possible, respectively, to select the destination DLNA player among the devices connected to the user's personal network and to connect to the *Edge Acquirer Service_App* to request the desired multimedia content scheduled by the content provider.
Likewise, a user that wants to watch a recorded content accesses the VDI by means of the User_App *Remote Controller*. The DMC is connected, besides the DLNA Players, also to the DMS to retrieve the recorded content. The latter can be stored in the local *Edge Storage*, if it has been recorded recently, or in the *Historical Recording Storage DC_App* on a remote data center, if it is older than a given time threshold.

Fig. 7.b presents the service chain related to the case when a user wants to receive the service directly on the user's Smart Client that is connected to the user's Personal Network through a cellular connection, under the radio coverage provided by the INPUT network operator. Deployment of the *Service_Apps* composing the personal cloud service in such a way the user can enjoy it according to the subscribed service level agreement is a task of the NS-OS.

IV. Performance Evaluation

In order to evaluate the gain achieved by using the INPUT platform, in this section we test the vSTB service in some situations timely chosen to highlight its power over a legacy approach. A comparative analysis will be carried out with a legacy approach realized by considering an identical service of live video streaming and video on demand, where users receive live video from two remote Italian content providers (in the following referred to as CP1 and CP2) and have the possibility of saving some received contents to be played out later when they want. In Section IV.A, we describe an emulation of the target scenario, while Sections IV.B and IV.C will present the performance analysis with some numerical results considering static and mobile users, respectively.

A. Target Scenario Emulation

The analyzed scenario is shown in Fig. 8. The INPUT network infrastructure is constituted by 7 NFVI-PoP nodes, 4 edge nodes close to the users, and 3 nodes for connection to the external Internet. 200 users populate the network; specifically, 50 of their homes are connected to each edge node, and users of each one generate an aggregate flow, whose trace has been randomly chosen from one of the real traces available in [32], with average bit rates ranging from 10 Mbit/s to 30 Mbit/s. 24 users have activated the vSTB service; specifically, their homes are distributed to the 4 edge nodes, i.e. 6 user home networks for each edge node.

Without losing in generality, in the following we will focus on one of these last users, whose home network is connected to the edge node E_1. A 4G base station is connected to the same edge node to cover an area around the user's home. Moreover, we assume that the edge node E_2 is very close to E_1, with a network latency of about 20 ms, while the edge node E_4 is some

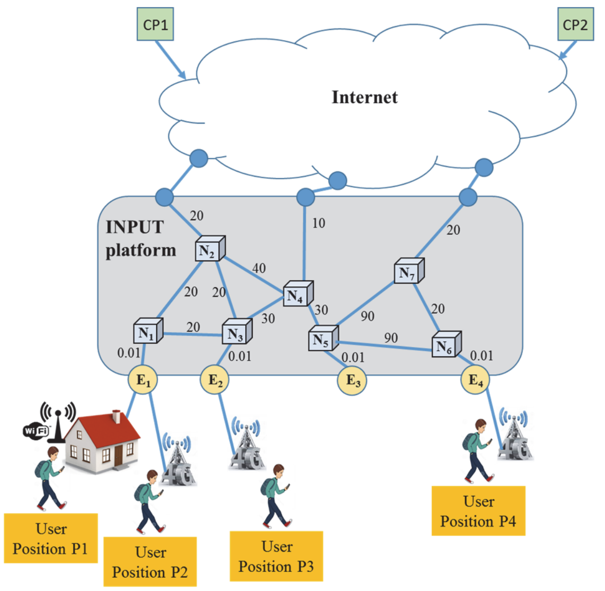


Figure 8: Target scenario: a network constituted by 7 NFVI-PoP nodes, 4 edge nodes close to the users, and 3 nodes for connection to the external Internet

TABLE I: VM Configuration for the vSTB Component.

VM	CPU	RAM	Disk
VDI	2	2 GB	20 GB
DMS	8	16 GB	20 GB
PA	8	16 GB	20 GB
EA	8	16 GB	20 GB
ES	2	4 GB	1000 GB

thousand kilometers away, with a network latency from the edge node E_2 of about 150 ms. The vHGW of the considered user runs on the node N_1, as well as the vSTB at the beginning of our analysis, i.e. when the user is still at home. Each NFVI-PoP node is realized with a DELL PowerEdge R630 server with dual processor Intel Xeon E5-2630L and 384 GB of RAM. OpenStack Ocata and KVM Hypervisor are installed over Ubuntu 16.04 LTS. The VM configurations used to run the vSTB components are listed in Table I.

In Section IV.B, where we will focus on a user at home, we will consider the service chain of the action "Watching a content on DLNA Player", shown in Fig. 7.a, while the service chain of the action "Watching a content on a mobile device", shown in Fig. 7.b, will be considered in Section IV.C to analyze performance when the user is in mobility.

In order to realize the scenario shown in Fig. 8, we connected the real NFVI-PoP nodes described so far with links that introduce the delays shown in the figure, expressed in ms, achieved by using VMs running the Linux Traffic Control tool, timely configured.

In our analysis, we evaluate three different ways to access and use the vSTB: a) vSTB configuration, achieved by interacting with the Virtual Decoder Interface only; b) watching a live video stream transmitted from a remote content provider, both CP1 and CP2; in this scenario, we will consider that the user performs channel surfing jumping from a live channel to another one; c) playout of a pre-recorded event; in this case, we

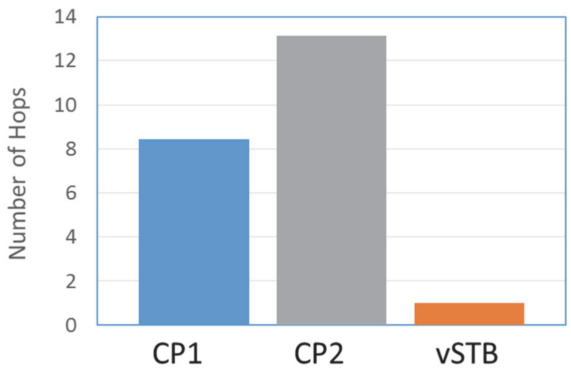


Figure 9: Distance between the smart device at user's home and the video server.

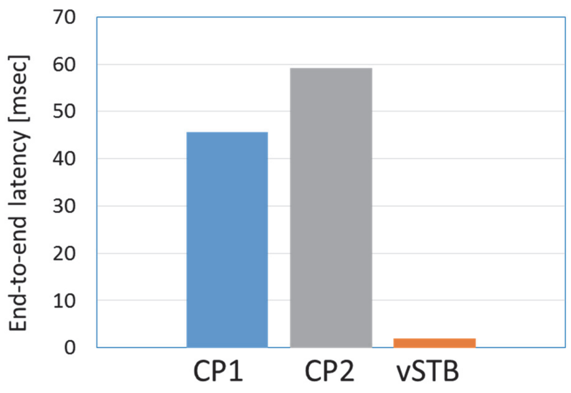


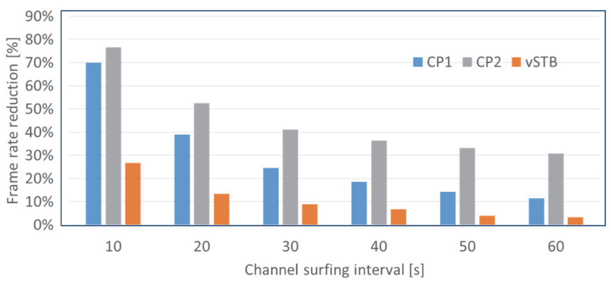
Figure 10: End-to-end latency measured from the user, in respect to the CPs and the vSTB.

will consider that the user performs the action of time jumping on the seek bar to change the playout instant of the movie being watched. In the second and third cases, some frames sent by the content provider during the channel variation period can be lost, causing a QoE degradation while waiting for the new video flow.

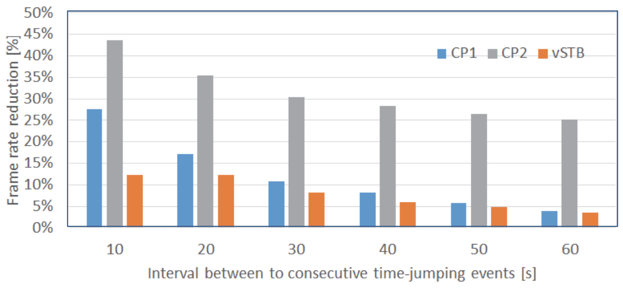
Interaction latency, frame rate reduction and lag time during channel surfing and time jumping on the seek bar will be considered as KPIs. Two important aspects that we will analyze are the impact of the position of the *Service_Apps* in the network in respect to the current position of the user, and the migration of the *Service_Apps* to be run as close to the user as possible. To this purpose, we will start by considering all the *Service_Apps* composing the vSTB chain concentrated in the same node N_1. Then, in Section IV.C, when the user changes access point, we will migrate, one by one, all the vSTB components towards the closest INPUT platform node to the user.

B. *Performance Analysis for static users at home*

In this section, we focus on a user that accesses the vSTB service at home, and present results related to the frame rate reduction and time lag during channel surfing and time jumping on the seek bar.



(a) Channel Surfing events during live video streaming



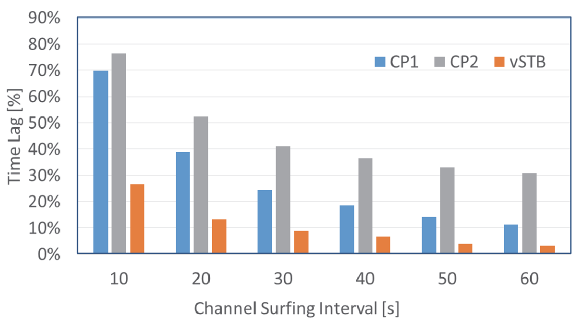
(b) Time-jumping events on the seek bar during a video on demand service

Figure 11: Frame rate degradation vs. the interval between two consecutive surfing events

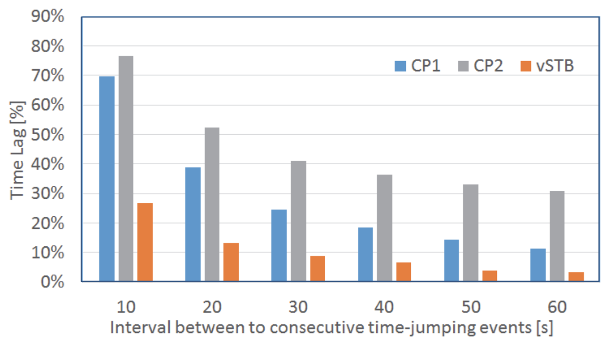
As a first step, we counted the number of hops between the user and the two content providers, CP1 and CP2. Fig. 9 compares these distances with the distance from the vSTB instance that runs on the N_1 node, which is the closest NFVI-PoP node of the INPUT network platform. Moreover, in order to better understand numerical results presented in the sequel, Fig. 10 shows the end-to-end delays measured between the user client device and the two CPs, compared with the delay from the vSTB installed on the N_1 node. Of course, this last graph is strictly related to the distances shown in Fig. 9, but also accounts for the delay performance of the underlying network.

In Fig. 11.a we reported the frame rate reduction due to channel surfing, measured against the mean time interval between two consecutive surfing events. We can observe that the INPUT approach clearly outperforms the legacy one in both cases of CP1 and CP2, providing users with the feeling of having a physical STB ahead in the same room. This is achieved owing to the fact that in the INPUT infrastructure, during channel surfing, channels are all available at a very close Edge Acquirer. On the contrary, without the INPUT facilities, users should connect to the content providers directly to request each required channel, and therefore they experience a higher delay that does not allow them a user-friendly channel surfing. An additional advantage is received by the underlying network because, with the INPUT approach, each flow is replicated for each client by the closest Edge Streamer, rather than by the content providers, making the platform scalable with the number of users.

Fig. 11.b presents the same performance metric in the case of time jumping on the seek bar, defined as a change of the current

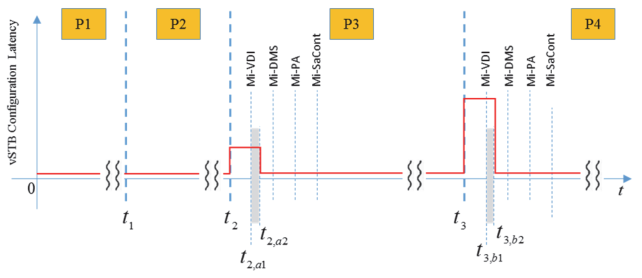


(a) Channel Surfing events during live video streaming

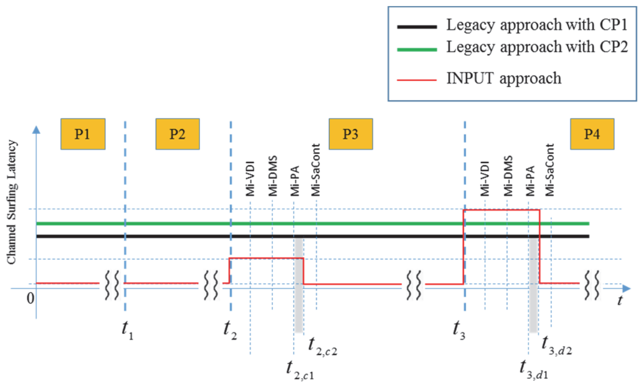


(b) Time-jumping events on the seek bar during a video on demand service

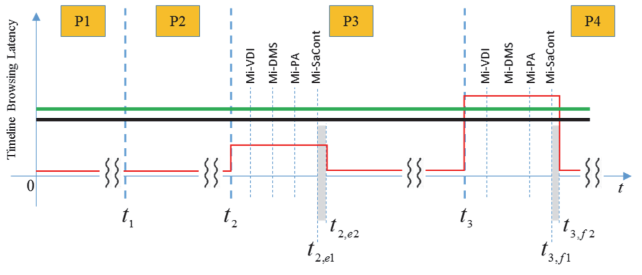
Figure 12: Percentage of service unavailability



(a) vSTB Configuration



(b) Channel Surfing



(c) Time jumping on the seek bar

Figure 13: Result overview for the case of mobile users

playback instant during on-demand video streaming. In addition, in this case, the INPUT approach presents better performance, since recorded events are stored inside the network, close to the users, while in the legacy approach the only way for the over-the-top content providers to store contents is on a remote cloud.

The second benchmark regards time lag, defined as the percentage of time the user is unable to view media contents, in both cases considered so far, that is, channel surfing and time jumping on the seek bar of a recorded event. To this purpose, Fig. 12 shows the percentage of time lag against the time duration of the channel surfing interval and the interval between two time-jumping events on the seek bar, comparing the legacy case to the case when the INPUT approach is applied. From the presented results, we can observe not only the gain achieved by using the INPUT platform, but also the fact that the measured parameters are almost constant, given that they depend on the round-trip delay from the closest Edge Acquirer, whereas they are strongly variable for the legacy streaming from remote content providers.

C. *Performance Analysis for users in mobility*

Let us now present results, in terms of interaction latency, while users change their Internet access point when they are in mobility. To this purpose, we focus on a user that begins to receive a vSTB service from the user's smartphone when at home. Then, the user leaves home for travel, moving towards the city airport; here, the smartphone is turned off, and remains in that state for the whole duration of the flight. Once arrived at the destination, the user switches on the smartphone again, and recommences to receive the vSTB service. More specifically, we highlight the following four situations: 1) the user is connected to the user's home LAN through the home WiFi (User Position P1); 2) when the user leaves home, the network is accessed through some 4G base station connected to the same edge node of the user's home (User Position P2); when the user is far from home, along the route towards the airport, the user accesses the network through some 4G base station connected to edge nodes different from E_1, but close to it (User Position P3); 4) when the user lands at the destination airport, the network is accessed through some 4G base station connected to edge nodes, e.g. E_4, very far from the previous ones.

In order to evaluate the interaction latency with the vSTB service, the analyzed benchmarks regard: 1) the interaction with the VDI for vSTB configuration; 2) channel surfing, achieved by changing channel; 3) time jumping on the seek bar, obtained by changing the playout instant. To this purpose, by using Wireshark, we measured the time occurred between the transmission of the first packet sent by the smartphone to start the action, and the last packet received by the smartphone reporting the successful action completion. In the presented

TABLE II: Numerical Results for the case of mobile users.

(a) vSTB Configuration Latency

User position	Latency Mean Value [ms]	95% Confidence Interval [ms]
P1	8.62	2.32
P2	8.46	2.54
P3 before VDI migr. from N_1	52.37	55.35
P3 after VDI migr. from N_1	8.20	2.41
P4 before VDI migr. from N_3	227.44	0.22
P4 after VDI migr. from N_3	8.39	2.37

(b) Channel Surfing Latency

User position	Latency Mean Value [s]	95% Confidence Interval [s]
P1	1.16	0.51
P2	1.19	0.52
P3 before migrations from N_1	1.30	0.57
P3 after VDI+DMS migration	1.28	0.56
P3 after VDI+DMS+PA migr.	1.27	0.56
P3 after migration completed	1.18	0.52
P4 before migrations from N_3	1.43	0.63
P4 after VDI+DMS migration	1.27	0.56
P4 after VDI+DMS+PA migr.	1.26	0.55
P4 after migration completed	1.17	0.51

(c) Latency due to time jumping on the seek bar

User position	Latency Mean Value [ms]	95% Confidence Interval [ms]
P1	1.03	0.26
P2	1.54	0.38
P3 before saved-content migration from N_1	41.52	0.22
P3 after saved-content migration	1.31	0.21
P3 before migrations from N_3	302.65	0.24
P3 after saved-content migration	1.20	0.22

analysis, the same action has been repeated for twenty times in order to estimate the mean value and the standard deviation of the latencies during the total measurement periods.

The measured interaction latencies are sketched in Fig. 13, where we highlighted the time-variant user position (P1, P2, P3 and P4), and the position variation instants (t_1, t_2 and t_3).

Numerical results are listed in Tables II and III. More specifically, the mean latencies experienced during configuration of the vSTB, channel surfing and time jumping on the seek bar are presented in Table II together with 95% confidence interval, each calculated for all possible positions of the user in the scenario shown in Fig. 8, before and after migration of the involved *Service_Apps*. Likewise, Table III presents results in terms of duration of migration of the relevant *Service_Apps*.

TABLE III: Duration of vSTB Component Migration [s].

From P_1 to P_3		
Migration duration of:	Mean Value [s]	95% Confidence Interval [s]
VDI	35.65	15.62
DMS	90.90	39.84
PA	133.75	58.62
Saved Content	33.70	14.77
From P_3 to P_4		
Migration duration of:	Mean Value [s]	95% Confidence Interval [s]
VDI	136.80	59.96
DMS	481.15	210.87
PA	690.55	302.65
Saved Content	299.25	131.15

With the aim of analyzing the impact of the *Service_App* placement, in the same Fig. 13, we highlighted the time instants when we migrated the Virtual Decoder Interface (Mi-VDI instant), the Digital Media Server (Mi-DMS instant), and the Personal Acquirer (Mi-VDI instant) to the node that is the closest to the current user position. Of course, the Edge Acquirer and the Edge Storage are not migrated to follow the user, because they are shared with other users. Instead, when the PA migration is complete, it is chained with the local Edge Acquirer running on the destination node of the migration. An additional migration considered to have some performance improvement regarding the playout of a pre-recorded event is related to the transfer of the file containing it, from the previous Edge Storage to the one running on the node where the vSTB chains have been migrated. In Fig. 13 it is labeled as Mi-SaCont (migration of a saved content).

As we can observe in Fig. 13, user position variations occurred at the instants t_2 and t_3 determine worsening of all the three interaction latencies, which are restored to the initial low values only after migrating some vSTB components. More specifically, as expected, vSTB configuration latency is reduced after migration of the VDI (at the instants $t_{2,a2}$ and $t_{3,b2}$), channel surfing latency is reduced after migration of the PA (at the instants $t_{2,c2}$ and $t_{3,d2}$), while latency due to time jumping on the seek bar is reduced after migration of the saved content to the Edge Acquirer running on the closest node to the new user position (at the instants $t_{2,e2}$ and $t_{3,f2}$). Gray areas indicate the VM migration periods, whose durations, of course, depend on their hardware configuration.

Fig. 13 also presents a comparison with the channel surfing latency and the latency due to time jumping on the seek bar experienced in the legacy approach case, for both the CP1 and CP2 content providers. We can observe that latencies experienced through the INPUT approach are lower than in the legacy case for almost the totality of time. The contrary can occur only when a user switches on the vSTB service when accessing the Internet from an edge node that is very far from the geographical area where the same service was used the last time; if this is the case, it happens only during the short transient periods when the vSTB components are not fully migrated close to the user. Instead, if the user changes access point moving along contiguous geographical areas (movements from P1 to P2, and from P2 to P3 in Fig. 8), the vSTB follows the user providing constant and very good performance.

V. Conclusions

In this paper, the network "softwarization" approach of the INPUT project and the associated network management framework have been described. In particular, the INPUT paradigm and the relative architecture have been introduced with the aim of embracing a cloud computing approach to support the personalization of services and the migration of the latter at the edge of the network to reduce service latencies. SDN, NFV and fog computing have been key ingredients of this softwarization process.

The INPUT technology enables next-generation cloud applications to go beyond classical service models, and even replace physical Smart Devices, usually placed in users' homes (e.g., set-top-boxes, etc.), with virtual entities, providing them to users "as a Service."

As highlighted in the paper, the INPUT framework gives advantages to both Telco Operators and users. In fact, along the softwarization path that Telco Operators have started to tread to softwarize their networks, the distributed data center they are creating inside the network will also allow them to provide application services at zero cost. In this way, they will be able to compete with over-the-top service providers, but with the advantage of being closer to the users. On the other hand, INPUT will provide users with the new facility of personal network, where personal cloud services can be installed and customized for them.

A relevant use case for the INPUT platform, namely, the Virtual Set-Top-Box, has been identified and evaluated, with the aim of proving the advantages of the proposed platform. Performance evaluation has been carried out in two different scenarios, characterized by a static user at home, and a mobile user that, while changing his/her Internet access point, is followed by the vSTB.

Acknowledgment

This work has been partially supported by the INPUT (In-Network Programmability for next-generation personal cloUd service supporT) project funded by the European Commission under the Horizon 2020 Programme (Call H2020-ICT-2014-1, Grant no. 644672).

References

[1] P. Wuttidittachotti, W. Akapan and T. Daengsi, "Comparison of VoIP-QoE from Skype, LINE, Tango and Viber over 3G networks in Thailand," *2015 Seventh International Conference on Ubiquitous and Future Networks*, Sapporo, Japan, 2015, pp. 456-461.

[2] W. Cai et al., "A Survey on Cloud Gaming: Future of Computer Games," *IEEE Access*, vol. 4, pp. 7605-7620, 2016.

[3] G. Maraviglia, M. Masi, V. Merlo, F. Licandro, A. Russo, G. Schembra, "Synchronous Multipoint E-Learning Realized on an Intelligent Software-Router Platform over Unicast Networks: Design and Performance Issues," *Proc. ETFA 2007*, Patras, Greece, Sept. 2007.

[4] A. Lombardo, F. Licandro, G. Schembra, "Multipath Routing and Rate-Controlled Video Encoding in Wireless Video Surveillance Networks," *Multimedia Systems*, vol. 14, no. 3, pp. 155-165, April 2008.

[5] C. Nguyen Le Tan, C. Klein, and E. Elmroth. Location-aware Load Prediction in Edge Data Centers. In *Proceedings of the 2nd IEEE International Conference on Fog and Edge Mobile Computing (FMEC 2017)*, pp. 25-31, 2017.

[6] E. Ahmed, A. Gani, M. Sookhak, S.H. Ab Hamid, F. Xia, Application Optimization in Mobile Cloud Computing: Motivation, Taxonomies, and Open Challenges, Vol. 52, Elsevier, pp. 52–68, 2015.

[7] White paper on "Software-Defined Networking: The New Norm for Networks," *https://www.opennetworking.org/*.

[8] A. Gelberge, N. Yemini, R. Giladi, "Performance Analysis of Software-Defined Networking (SDN)," *Proc. 21st IEEE Intern. Symp. on Modeling, Analysis & Simulation Computer and Telecommunication System (MSACOTS)*, pp. 389-393, 2013.

[9] White paper on "Network Functions Virtualisation", at *http://portal.etsi.org/NFV/NFV_White_Paper.pdf*.

[10] G. Faraci, G. Schembra, "An Analytical Model to Design and Manage a Green SDN/NFV CPE Node," *IEEE Transactions on Network and Service Management*, vol. 12, no. 3, pp. 435 – 450, Sept. 2015.

[11] G. Karagiannis, *et al.*, "Mobile cloud networking: Virtualisation of cellular networks," in Proc. of the 21st International Conference on Telecommunications, ICT 2014, pp. 410–415, Nanjing, China.

[12] W. Tärneberg, A. Mehta, E. Wadbro, J. Tordsson, J. Eker, M. Kihl, and E. Elmroth, "Dynamic application placement in the Mobile Cloud Network," *Future Generation Computer Systems*, Elsevier, Vol. 70, pp. 163-177, 2017.

[13] G. Incarbone, G. Schembra, "A Business Model for Multimedia Streaming in Mobile Clouds," IEEE Transactions on Network and Service Management, Volume 11, Issue 3, pp. 376 – 389, September 2014.

[14] The INPUT Project, *http://www.input-project.eu*.

[15] L. M. Vaquero, L. Rodero-Merino, "Finding your Way in the Fog: Towards a Comprehensive Definition of Fog Computing", *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, Oct. 2014.

[16] Cisco Fog Computing Solutions: Unleash the Power of the Internet of Things, *http://www.cisco.com*.

[17] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia, "A view of Cloud Computing", *Communications of the ACM*, vol. 53, no. 4, Apr. 2010.

[18] The 5G-PPP Association, "Advanced 5G Network Infrastructure PPP in H2020," Nov. 2013, URL: *http://5g-ppp.eu/wp-content/uploads/2014/02/Advanced-5G-Network-Infrastructure-PPP-in-H2020_Final_November-2013.pdf*.

[19] D. Matsubara, T. Egawa, N. Nishinaga, V. P. Kafle, M.-K. Shin, A. Galis, "Toward Future Networks: A Viewpoint from ITU-T," *IEEE Communications Magazine*, vol. 51, no. 3, pp. 112-118, Mar. 2013.

[20] ETSI GS NFV 001, "Network Functions Virtualization (NFV): Use Cases", 10-2013, available at *http://www.etsi.org/deliver/etsi_gs/nfv/001_099/001/01.01.01_60/gs_nfv001v010101p.pdf*.

[21] DLNA standard, *http://www.dlna.org*.

[22] V. Quintuna and M. Laye, "Modeling and optimization of Content Delivery Networks with heuristics solutions for the Multidimensional Knapsack Problem," *2016 17th International Telecommunications Network Strategy and Planning Symposium (Networks)*, Montreal, QC, 2016.

[23] Z. Feng, M. Xu, Y. Yang, Y. Wang, Q. Li and W. Wang, "Optimizing content delivery in ICN networks by the supply chain model," 2016 IEEE 35th International Performance Computing and Communications Conference (IPCCC), Las Vegas, NV, 2016.

[24] Z. Wang, J. Zhang, H. Li and J. Liu, "A new fast P2P video transmission method applied in asymmetrical speed channel environment," in *Journal of Communications and Networks*, vol. 12, no. 3, pp. 209-215, June 2010.

[25] Y. C. Chen and C. Y. Liao, "Improving quality of experience in P2P IPTV," 2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS), pp. 1-4, Kanazawa, 2016.

[26] G. Incarbone, A. Lombardo, G. Schembra, "Applying P2P to Achieve Real-Time Multimedia Communications: Modeling and Performance Analysis," *IEEE Proc. GLOBECOM 2007*, Washington, DC, USA, November 26-30, 2007.

[27] OpenStack cloud computing software platform URL: *http://www.openstack.org/*.

[28] R. Bruschi, P. Lago, G. Lamanna, C. Lombardo, S. Mangialardi, "OpenVolcano: An Open-Source Software Platform for Fog Computing," *Proc. 1st ITC Workshop on Programmability for Cloud Networks and Applications (ITC PROCON 2016)*, Würzburg, Germany, Sept. 2016.

[29] The Intel "Data-Plane Development Kit," URL:*http://www.dpdk.org*.

[30] R. Bolla, R. Bruschi, C. Lombardo, F. Podda, "OpenFlow in the Small: A Flexible and Efficient Network Acceleration Framework for Multi-Core Systems," *IEEE Transactions on Network and Service Management*, vol. 11, no. 3, pp. 390-404, Sept. 2014.

[31] R. Bruschi, F. Davoli, P. Lago, J. F. Pajo, "A Scalable SDN Slicing Mechanism for Multi-domain Fog/Cloud Services," *Proc. 1st First IEEE International Workshop on Network Programmability - From the Data Center to the Ground (NetFog 2017)*, Bologna, Italy, 2017.

[32] The CAIDA Anonymized Internet Traces 2016 Dataset, at: https://www.caida.org/data/passive/passive_2016_dataset.xml