Realizing services and slices across multiple operator domains

EU H2020 5GEx Consortium: Janos Czentye[†], Molka Gharbaoui[‡], Giovanni Giuliani[§], David Haja[†], Janos Harmatos[¶], David Jocha[¶], Juhoon Kim[‖], Barbara Martini[**], Javier Melián[††], Paolo Monti[‡‡], Balazs Nemeth[†], Wint Yi Poe[*], Aurora Ramos[††], Andrea Sgambelluri[‡], Balazs Sonkoly[†], Laszlo Toka[†], Francesco Tusa[x], Ishan Vaishnavi[*], Carlos J. Bernardos[xi], Robert Szabo[¶]

[†]Budapest University of Technology and Economics (BME), [‡]Scuola Superiore Snat'Anna Pisa, [§]HPE Milan, [¶]Ericsson Hungary, [‖]Deutsche Telekom AG, [**]CNIT Pisa, [††]ATOS Spain, [‡‡]KTH Sweden, [*]Huawei Technologies Germany, [x]University College London, [xi]UC3M Spain

***Abstract*—Supporting end-to-end network slices and services across operators has become an important use case of study for 5G networks as can be seen by 5G use cases published in 3GPP, ETSI as well as NGMN. This paper presents the in-depth architecture, implementation and experiments on a multi-domain orchestration framework that is able to deploy such multi-operator service as well as monitor the service for SLA compliance. Our implemented architecture allows operators to abstract their sensitive details while exposing the relevant amount of information to support inter-operator slice creation. Our experiments shows that the implemented framework is capable of creating services across operators while fulfilling the requirements of the network functions that form that service.**

***Index Terms*—Network function virtualization, Multi-operator Orchestration, 5G slicing**

I. Introduction

The next generation of telecommunication networks is driven by a change in the business paradigm. Due to limited incomes from end-customers, operators are now looking for newer business models that may be of interest for newer types of customers, such as the the vertical industries whose requirements may note match the existing coverage and the technical skills of telecom operator. As an example, the requirements of a connected-car company may extend to multiple countries across the globe, going beyond the coverage capabilities of a single operator. Hence, to create a uniform set of services for the verticals' end-customer, vertical industries would need to create the same service supported by the same slice over multiple operator networks. The creation of global services for the vertical will likely be provided by a single operator who, in turn, can leverage on existing relationships with other operators. A single operator, in an *automated way*, is then able to establish symbiotic relationships with other operators to create a multi-operator service/slice, as shown in Figure 1.

A vertical customer (e.g. car company) requests services across multiple operator domains. The service in 5G will be supported over a typically virtualized and possibly isolated infrastructure as well as network functions, currently called a slice. Over interface 1 the car company will be able to specify the requirements for the service, including the QoS and the coverage area. The operator's multi domain management and orchestration system (MdO) is responsible to create the slice that would support this service. To do so it may interact with another operator's MdO (over interface 2) or its local technical domain orchestrators (over interface 3). Once the slice is created, the end-customer (car owner) will be able to use the service (connected car) over the coverage area specified by the vertical customer via interface 1. The end-customer can then expect similar levels of service across different geographies. This work proposes the in-depth architecture and implementation of the MdO and also describes the deployment process of sample initial services.

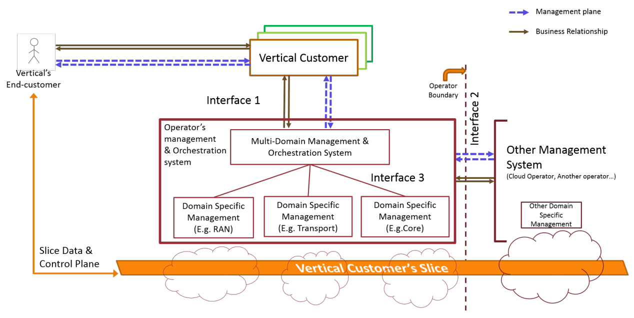
Management plane
Business Relationship
Vertical's End-customer
Vertical Customer
Interface 1
Operator Boundary
Interface 2
Operator's management & Orchestration system
Multi-Domain Management & Orchestration System
Interface 3
Domain Specific Management (E.g. RAN)
Domain Specific Management (E.g. Transport)
Domain Specific Management (E.g.Core)
Other Management System
(Cloud Operator, Another operator...)
Other Domain Specific Management
Slice Data & Control Plane
Vertical Customer's Slice

Fig. 1. The scenario for multi-operator Services

II. Related Work

The initial work on the concept and design of the MdO began with the use cases and requirements expressed in [1]. More, in depth, details on how to place VNF in multi-domain architectures was researched in a number of previous studies [2], [3], [4]. An in depth analysis of multi-domain orchestration frameworks is in our previous work [5].

There are two main works that largely contributed to the development of the MdO framework presented in this paper: the UNIFY [6] and the TNOVA [7] projects. The UNIFY project [6] uncovers the advantage of SDN-enabled network infrastructure that is built for the full network virtualization and agile service creation. The MdO framework presented in

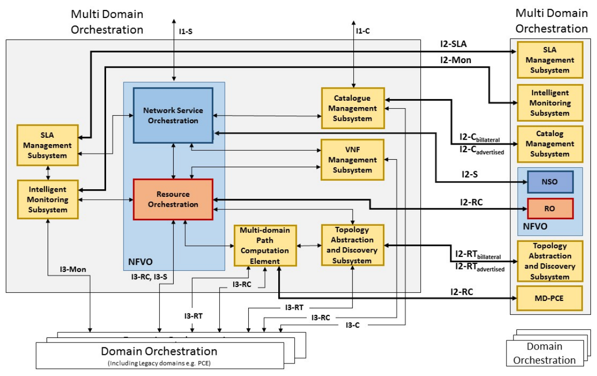
Multi Domain Orchestration
I1-S
I1-C
Multi Domain Orchestration
I2-SLA
I2-Mon
SLA Management Subsystem
Intelligent Monitoring Subsystem
Network Service Orchestration
Catalogue Management Subsystem
Catalog Management Subsystem
I2-C billateral
I2-C advertised
VNF Management Subsystem
I2-S
NSO
Resource Orchestration
I2-RC
RO
NFVO
Multi-domain Path Computation Element
Topology Abstraction and Discovery Subsystem
I2-RT billateral
I2-RT advertised
Topology Abstraction and Discovery Subsystem
NFVO
I3-Mon
I3-RC, I3-S
I3-RC
I2-RC
MD-PCE
I3-RT
I3-RT
I3-RC
I3-C
Domain Orchestration
(Including Legacy domains e.g. PCE)
Domain Orchestration

Fig. 2. Multi-domain Orchestration system architecture

this paper extends and details UNIFY's concept for network slicing over multiple service providers. The T-NOVA project aims at designing and implementing a NFV management and orchestration framework that enables customers selecting VNFs directly from a platform called NFV Marketplace. This feature is commonly referred to as a Network-Function-as-a-Service (NFaaS). Concepts from both UNIFY and TNOVA project are re-used in the design of our orchestration system in the 5GEx project.

III. Architecture

This section discusses the system design architecture of the Multi-domain Management and Orchestrator (MdO) of Figure 1. A MdO coordinates resource and service orchestration across multiple domains (multi-technology or multi-operator). Components of the MdO architecture and inter-and intra-MdO interfaces are shown in Figure 2.

The generic slice deployment process in the MdO is supposed to consist of the following four steps:

- Capability detection: This is the pre-step to service deployment where the MdO from one operator (home MdO) can detect the capabilities of other operator as exposed by the respective MdO. Note that the other operators MdO may expose only a limited set of *abstracted* capabilities to hide internal details of the operator. This abstraction presents the main challenge for the other steps.
- Deployment: In this step the request arrives over interface 1 from the vertical, the MdO evaluates the request and splits it into parts that are to be deployed by each domain orchestrator (DO) or another MdO.
- Assurance: Once the service is deployed the operator is responsible for verifying the SLA towards the customer. To do this it must monitor the SLA KPIs across all the domains and the operators of the deployed service.
- Reconfiguration: In the case of failures or SLA violations, the Service or the slice that supports it will need to be reconfigured by the home MdO.

The components of the MdO can be classified based on these steps into three main groups representing the core functionalities of the MdO.

Components that gather resources and services related information: The Catalogue Management Subsystem is responsible for maintaining the VNF Catalogue (list of available functions a Service Provider can utilize to compose Network Services), Network Service Catalogue (list of services that a certain MdO provides to a Customer) and takes care of the catalogue synchronization among MdOs via I2-C. Topology Abstraction and Discovery Subsystem (TADS) is in charge of maintaining a database about the networking, compute and storage resources available for the Resource Orchestrator (RO) and Network Service Orchestrator (NSO). Furthermore, there is a topology information exchange between the TADS and the Multi-Domain Path Computation Element (MD-PCE); TADS acquires the intra-domain connectivity topology from MD-PCE, while informs MD-PCE about the Multi-domain topology gathered from the TADS components of other MdOs.

Components that relate to the deployment procedure: When the MdO receives a request from the customer, the deployment operation is carried out by the Network Service Orchestration (NSO), Resource Orchestration (RO) and the MD-PCE. The NSO is responsible for handling the Network Services (NS), requested by the customers or the NSO of another MdO. RO is in charge of embedding the resource requests to the available domain resources offered by the TADS. MD-PCE is responsible for managing the traditional network connectivity services (e.g MPLS). In the case of VNF as a Service (VNFaaS) request, the VNF Management Subsystem is invoked to handle the Lifecycle Management (LCM) operations for the service, such as service topology evolution, management of the service component dependencies and the component LCM. VNFM and RO are working in cooperation, however, clear separation between their functionality is kept: while VNFM is aware of service component LCM, unaware of the embedding procedure, which is handled by the RO.

Assurance components for the deployed service: When the service is deployed, the NSO/RO passes information on the SLA and resource entities to be monitored to the assurance components of the architecture, namely SLA Management Subsystem and Intelligent Monitoring Subsystem (IMoS). IMoS is in charge of the coordinated deployment and management of probe-based measurement methods for different domains within a provider or across multiple providers by interworking with IMoS component of other MdOs. The SLA Management Subsystem is responsible for evaluating service KPIs belonging to a certain running instance by using the performance measurement reports provided by the IMoS.

A. Interfaces

On interface I1 customers can specify their requirements for a service by using service specification customer-to-business APIs. Each MdO is aware (over I2) of the service capabilities and resources of other administrative domains in order to make service and resource orchestration decisions. The MdO interacts with other MdOs via I2 business-to-business interfaces to exchange information as well as to request and orchestrate resources across administrative domains. Two types of interfaces are distinguished for I2: by using the advertisement-based option an MdO can announce its capabilities, whereas using bilateral type exposure the MdOs can exchange further details such as abstracted topology, resource, service, pricing related information. The MdO interacts with the corresponding domain orchestrators using I3 interface APIs. Each of I1, I2, I3 is further split in Figure 2 based on the functionality of the component it supports. Initial requirements over the interface I2 have already been published in standards [8] as a direct contribution of this work. A detailed specification of the implemented interfaces is expected to be published soon.

B. Security Aspects

The security aspects in the overall architecture have been looked into in depth. Best practices clearly suggest that the SW that implements the MdO needs to be deployed inside a Trusted Zone of the operator data center. Interfaces $I3-*$ are used by actors belonging to the same administrative domain, inside trusted zones of the same security level. The security aspects of the other interfaces follow.

1) Securing $I1-*$ Interfaces: All the end-customers (indirectly through some SW client application like a Dashboard, or other SW using $I1-*$ APIs) using any of the $I1-*$ interfaces need to be registered into a centralized exchange point. Following the administrative domain policies as registered end customers their credentials are internally accessible to the MDO. All $I1-*$ interfaces use JSON/REST protocol secured by HTTPS with MDO server certificate (*Communication Privacy*): at the beginning of each session (login), the users be correctly identified and authenticated by using the stored credentials and will be returned a signed time based token (in *JSON web token (JWT)* format, signed by MDO itself) to be used at the next requests. In any successive request the JWT will always be provided to grant access to the interface (*Identification and Authentication*) based on the role of the user (*Authorization*)

2) Securing $I2-*$ interfaces: The collaboration among different MDOs requires mutual authentication, therefore, each MDO needs to have a valid X.509 certificate issued by a commonly accepted Certificate Authority (CA). All $I2-*$ interfaces will need to comply to these rules: administrative domains becoming business partners in 5GEx, need to register in a peer-to-peer mode (including the exchange of public certificates). TLS (SSL) protocols with both client and server authentication (e.g. using JSON/REST protocol on HTTPS connections with client and server verification) need to be used to guarantee the identity of both partners of the collaboration (*Communication Privacy, Identification, Authentication*). Moreover, the identity of the MDO with respect to the registered partner list will be checked (*Authorization*). Finally, a mechanism to support *non-repudiation* of request needs to be setup, e.g. trusted audit logs and request signing, to resolve inter administrative domains issues also from legal perspective.

C. Business and Pricing Aspects

Pricing information is assigned to a service by means of the information model that is used to describe the items for sale in the catalog. A pricing offering for a specific service will be stored in the catalog as part of the offering. The currently implemented pricing model for VNFaaS case is pay-as-you-go model, following the approach of cloud services [9], [10]. The information model used for this is a part of the SLA and is shown in Table I Other billing models are still being studied currently for other types of services, e.g. connectivity services, along with pricing negotiation.

TABLE I
PRICING MODELS FOR THE SERVICE

NSD:SLA:billing	
type	< Billing Model (PAYG)>
period	<Period>:<Number of Periods>:<Period unit (Hours, days)>
Price	< numeric amount for the period>
NSD:SLA:billing: price	
currency	< currency type (EUR)>
setupCost	<Numerical setup cost>
pricePerPeriod	<charge per period>

IV. Implementation Details

The following sections present the implementation details of the components presented in the architecture, Figure 2.

A. Information Gathering Components

1) Catalog management - Multi-domain Catalog : The MdO operator decides which items (NSs or VNFs) are shared and with whom, establishing sharing policies after going through a process of testing, validation, and adaptation (in terms of SLA and pricing). These are stored in the Multi-domain catalog (MDC). The local MDC retrieves a list of shared items from remote MDCs using REST API based on the sharing policies established in the remote domain.

In communication with the TADS module via REST API, the catalog subsystem also keeps a record of the registered domains within the system to know which domains are available at any time and take care of catalog inconsistencies among the peers, ensuring that the list shown is only of available items.

Via the 5GEx dashboard, the Service Provider can create NSs for the customers that are composed of VNFs and (recursively) other NSs either local or remote that have been added previously to the local catalog.

2) TADS: The Topology Abstraction and Discovery Subsystem (TADS) discovers other MdOs and then retrieves information about the type and quantity of resources available (including both networking and IT). The retrieved information is maintained in local database (TED) and is then provided to the other modules of the MdO. Local information (i.e., the URL of MdO entry point, the updated view of its abstracted network resources, the abstracted view of its overall availability of IT resources and MD-PCE entry-point) is kept up-to-date in the DB and is exchanged with the other MdO TADS.

The internal architecture of TADS is shown in Figure 3. The Topology Module (TM) provides the NFVO, the Catalogue subsystem and MD-PCE with the information related to the abstracted view of all the resources available at each (discovered) MdO. The communication with the NFVO (intra-MdO)

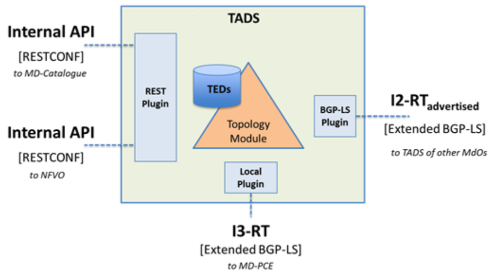

Fig. 3. TADS architecture

is done by using the REST Plugin that presents a dedicated API. The NFVO provides an up-to-date abstracted view of local IT resources to the TADS, while the TADS provides to the NFVO the retrieved abstracted information collected from other MdOs. In order to support the catalogue exchange, the REST Plugin provides also a unidirectional API to the Catalogue subsystem, where the list of the registered Operators and their related entry-point is provided. The Local Plugin has been introduced to enable the dynamic exchange of topology information between TADS and MD-PCE. In particular, the TADS receives the description of local connectivity topology from the MD-PCE, while it exports to MD-PCE the full Multi-domain topology received by the TADS of other MdOs.

The BGP-LS Plugin is at the heart of the TADS discovery process. The BGP-LS Speakers of different MdOs are able to communicate through the Interface 2-Resources and Topology advertised (I2-RTadvertised interface) for exchanging the network topology, overall IT resources information and MD-PCE entry-point. In particular, the BGP-LS Plugin is responsible for exporting the abstracted view of local resources, and for importing the abstracted views of the other connected MdOs. The solution adopted for the I2-RTadvertised interface is an extended version of the BGP-LS protocol [11] that supports the advertisement of TE metrics [12] and of IT resources represented using the UNIFY data model [13], [14].

B. Orchestration Components

1) VNF Manager: The VNF Manager (VNFM) subsystem handles the component resolution and the Lifecycle management (LCM) belonging to a particular service. A given service description is stored in the *Service Catalogue* in the CMS (Section IV-A1) and may contain multiple embedding states (represented by one or multiple blueprints), inputs and triggers that may cause transitions among states. When a service deployment request is received by the RO, the VNFM will be invoked and the *Deployment Logic* will check the requested service creation in the *Catalogue* verifying its initial state and the called workflow (e.g. start, stop, delete, etc). Then a Deployment state is created and sent to the RO by using REST API. The RO handles the current service configuration offered by the VNFM and performs its orchestration steps that might result in success or embedding error. In the latter case, the VNFM will be invoked again and will start the iteration among the stored embedding states.

2) NSO: The high-level software architecture of the 5GEx NFVO encompassing the NSO and RO is shown in Fig. 4. The entire functionality of the Network Service Orchestrator

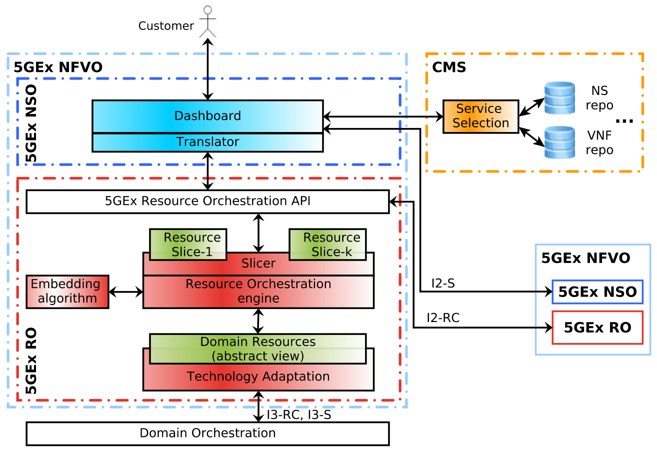

Fig. 4. Software architecture of 5GEx NFVO

(NSO) is implemented by three different co-operating components including the Catalog Management Subsystem (CMS). First, the Dashboard is used by the Service Providers (SPs) to create Network Services (NSs). An SP i) chooses the constituent VNFs from the list of available ones, ii) gives SLA requirements, ii) defines the VNFFG containing virtual links between VNFs, and iv) defines additional parameters, such as assurance parameters and billing information. Second, the Service Selection module (part of the CMS) is responsible for managing a service instantiation request. During this process, it interacts with both the NS and VNF repositories. VNFDs of constituent VNFs are retrieved from the VNF store while the created/purchased NS is stored/retrieved in/from the NS store (NSDs). Third, a Translator module converts the customer's service instantiation request (service selection) to a deployment request inter-operating with the RO. This component enables basic network service management, such as starting and terminating NS instances. This module is in charge of storing and providing assignment of VNFs to NS instances. By these means, e.g., IMoS can request a list of VNFs encompassed by an NS instance.

The current version of the NSO also provides a versatile traffic steering control over virtual links; VNFFG deployment including supports delay and bandwidth requirements on virtual links of the VNFFG; and VNF on boarding as well.

3) RO: Resource Orchestrator (RO) of our 5GEx NFVO is in charge of two key tasks. On the one hand, RO is responsible for joint virtualization and control of compute/storage and network resources. On the other hand, it calculates (a close to optimal) embedding of the NS request and adapts to different technological domains. The software architecture of the RO is depicted by the lower part of Fig. 4.

RO gathers resource information from local and remote domains via TADS and maintains a global abstract resource view. On this view, it can construct different resource slices (virtual views) exposed towards different consumers or other operators. The creation and configuration of resource slices is triggered from the (local or remote) NSO level. We have implemented a novel interface to enable slice configuration for remote providers. The new SlaaS interface, which is implemented at MdO and DO level as well, is based on the Virtualizer data model [13], [14] (also used at interfaces I2-RC, I2-RTbilateral, I3-RC, and I3-RT) and it supports the management of multiple Virtualizer objects corresponding to different resource slices. A special purpose infrastructure slice has been introduced which can expose the low-level details of the given domain providing privileged control over the resources. Physical resources can therefore be bound to virtual resources, rendering an VNF running in a given slice controllable from another slice.

NSO sends an NS request to RO described for a given resource slice. The request includes VNFs, a VNFFG and requirements. We support different types of requirements such as, delay and bandwidth on virtual links of the VNFFG, end-to-end delay, affinity and anti-affinity constraints. The embedding engine of the RO calculates the best placement of the VNFs and optimizes the forwarding overlay based on a fast and efficient greedy heuristic. At this operation level, the resource slice is handled in an abstract way and the RO does not differentiate between local and remote domains. The result, which is a resource slice configuration describing the deployment, is split according to the involved domains. As a final step, the corresponding parts of the deployment result are sent i) to the local domains via interface I3-RC which is responsible for technology adaptation, and ii) to remote MdOs via interface I2-RC. In the latter case, the remote RO performs similar steps and further orchestrate the received request but this operation is hidden from the local MdO.

C. Assurance Framework

1) IMOS: Intelligent Monitoring Subsystem (IMoS) implements the intelligent and coordinated end-to-end monitoring required for the management and orchestration of services and resources in multi-technology, multi-domain 5G networks. IMoS has been implemented following the RO hierarchical structure that enables measuring the performance of a service instance either belonging to a single provider, or spanning multiple administrative domains. The IMoS instance running on the MdO that receives the service request, is responsible for triggering the collection of monitoring information from all the (technological and administrative) domains implementing that service instance. Therefore, IMoS supports mechanisms that enable the orchestrated deployment and configuration of probes both on heterogeneous resource domains by interacting with the respective Local Monitoring Systems (LMSs), and on external administrative domains by cooperating with the remote IMoS instances running on there.

In order to optimize the resource utilization related to the deployment of probes for KPI monitoring as well as reducing possible communication overheads, an ILP model is implemented in IMoS using the Gurobi solver library to determine the best probe embedding strategy. The interfaces I3-Mon and I2-Mon are designed and implemented to support the monitoring functions across multi-technology and multi-administrative domains respectively. The former is fully supported by Lattice through the REST API implemented by the Lattice Controller, which allows the on-demand instantiation and configuration of Lattice monitoring entities. Those entities natively provide monitoring mechanisms for physical hosts and KVM VMs and are also able to work as proxy probes that collect monitoring information from already available LMS (e.g., Docker monitoring and Openstack Ceilometer). The monitoring data collected by the aforementioned probes will be stored in a central database in time series format, and will be processed/aggregated across different administrative domains in order to support the SLA Manager in verifying the global SLA fulfillment of a service.

2) SLA Manager: The SLA Manager is in charge of evaluating the selected KPIs for each of the running instances (slices, Network Services, VNFs, etc.) according to an agreement that is established and signed upon the service instantiation. The SLA Manager subsystem is composed of two main modules:

- *S*LA Evaluator: This module is in charge of receiving the contract agreements from the user interfaces that relates vertical customer (over interface 1) as well as the another operator (over interface 2). Each one of these agreements includes a set of KPIs and thresholds that need to be meet in order to fulfil a Service Level Objective. The way of evaluating these metrics (KPIs) is by means of its monitoring information that tells about the behaviour of the instance we want to monitor and evaluate.
- *S*LA Aggregator/Proxy: This module receives the request from the SLA Evaluator and looks for the appropriate monitoring information in the local monitoring database. To be aware of the location of the metric, this module is informed about it on provisioning time. Metrics can be either simple, that do not require extra effort to gather the monitoring information or complex, which are calculated based on a formula containing simple metrics and/or recursively, other complex metrics.

Each local SLA Evaluator requests monitoring information for all of the KPIs to be evaluated to the local SLA Aggregator. The later collects all the simple metrics monitoring information from the remote locations, compress the sample to reduce traffic overload across domains, and finally aggregating the information and transfer the results to the original SLA Evaluator. The detailed monitoring values if required by one MdO from another MdO are requested through the IMoS, the SLA aggregator provides the service level metrics.

D. Integration with SDN domain manager

This section describes an implementation of interface 3 between the MdO and an SDN Domain Orchestrator. The SDN DO allows the programmable provisioning of data delivery paths through a sequence of virtual functions, i.e., forwarding graphs (FGs) [15]. The application layers (i.e., the MdO),

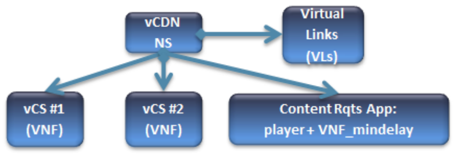

Fig. 5. The virtual CDN (vCDN) Service Graph

through a REST API, can require the set-up and tear-down of service delivery paths by just specifying the Service Access Points (SAPs) to connect and, in case, a set of VNF instances to traverse, thus, using an application-oriented semantic rather than dealing with technology-specific low-level details. The DO performs mapping operations to decide the path of each segment connecting the source SAP or a VNF to the next VNF or the destination SAP. Then it enforces a set of forwarding rules to establish every path segment.

Finally, the DO offers interface for adaptation capabilities for the established paths to recover from congestion events (e.g., service outages or degradation events) that are possible when a concurrent resource usage takes place according to IETF SFC guidelines. The adaptation capabilities involve the continuous monitoring of the paths status (via interfaces exposed to IMoS over I3-Mon) and the collection of statistics that are then made available to the MdO to enhance its resource orchestration capabilities. In this way, the MdO dynamically controls the status of the deployed entities.

V. The vCDN experiment

This experiment demonstrates the end-to-end MdO lifecycle for a virtual content delivery network service (vCDN) service composed by 2 non-collocated virtual Content Servers (vCS1 and vCS2).

1) Scenario: The scenario includes the following roles:

- vCDN-ProviderDomain1: creates vCDN service offerings stored in the service catalog for advertisement, and outsource a resource slice in domain 2 to deploy vCS 2.
- NFVI-ProviderDomain2: provides a resource slice in a second MdO domain that is used for vCS 2 deployment.
- Customer-vCDN consumer (e.g. content provider): requests the partially configured vCDN NS.
- End User-content consumer: requests the content to the already deployed vCDN service entry point.

2) vCDN service specification and implementation: The vCDN Service in this experiment is formed by 2 virtual content server and 1 content request application that is composed by 2 VNFs: a player and a VNF that enablers to select the vCS that will imply the minimum delay on the content delivery to the end user, see Figure 5.

- vCSs are implemented using Wowza Streaming Engine, which is containerized to act as another VNF and included within the VNF descriptor (VNFD) meta-model which is ETSI NFV compliant.
- The content request app is formed by: 1) a player that reproduces the media content and 2) a VNF that selects the vCS minimizing mean delay.

Some of the most relevant fields in the VNFD for these VNFs are: hardware requirements, for an optimal creation of the resource slice; container image, which the Docker orchestrator would download, provision and deploy; appropriate environment for the streaming software;list of ports that each VNF uses and have to be opened and mapped to host ports; virtual links that will be exposed and visible to create a VNFFG to compose a complex NS; SLA parameters or list of metrics that can be monitored and evaluated; and starting scripts to be executed to start services inside the container for an optimum performance of the Wowza server, e.g. adding the media cache source, etc.

3) Experiment steps, architectural components involved and results: The steps of the experiment follow the steps mentioned in Section III-C.

- Pre-requisites:
  - VNFs specification: VNFDs are created and uploaded to the MdO CMS together with VNF images
  - Neighbor domains are discovered by TADS.
- Creation of the NSD including the VNFFG among the vCSs: the provider creates an NS including business offering.
  - Syntax validation of the descriptor
  - The new NS is stored in the CMS
- NS instantiation request takes place by the selection of the service in the CMS and it is sent to the NFVO (NSO+RO). The NSO in multiple administrative domains includes the NSs on-boarding (referenced VNF images loading), NS instantiation including placement information, NS monitoring and NS termination. As part of the RO a slice request is performed to a remote domain so the multi-domain resource orchestration of compute and networking resources takes place.
  - For each of the vCS, a resource slice is created: one in domain 1 and the other in domain 2.
  - Each of the containers are downloaded and deployed in each of the resource slices.
  - The vCSs external links are linked based on VNFFG.
  - Service Provisioning and deployment takes place.
- The monitoring info is received in dashboard via IMoS.
- The vCDN is working properly when the content request is forwarded to the vCS that with the minimum delay and the end-user is able to see the video content.

VI. Conclusions

This document provides a description of the MdO prototype that enables the deployment of services across operators using the concept of resource slicing. We presented the in-depth architecture of each of the components that compose the MdO and their functionalities and implementation aspects. We presented just one of the many services that can be deployed over testbeds across the EU that emulate multiple operators. Pilot test with real end-users of the said services are underway.

VII. Acknowledgment

This work has been supported by the European Community through the 5GEx project (grant no. 671636) within the H2020 program.

References

[1] D. Perez-Caparros, I. Vaishnavi, S. Schmid, and A. Khan, "An architecture for creating and managing virtual networks," in *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Sept 2013, pp. 2984–2988.

[2] I. Houidi, W. Louati, W. B. Ameur, and D. Zeghlache, "Virtual network provisioning across multiple substrate networks," *Computer Networks*, vol. 55, no. 4, pp. 1011 – 1023, 2011, special Issue on Architectures and Protocols for the Future Internet. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128610003786

[3] I. Vaishnavi, R. Guerzoni, and R. Trivisonno, "Recursive, hierarchical embedding of virtual infrastructure in multi-domain substrates," in *Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft)*, April 2015, pp. 1–9.

[4] N. M. M. K. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," in *IEEE INFOCOM 2009*, April 2009, pp. 783–791.

[5] R. Guerzoni, I. Vaishnavi, D. Perez Caparros, A. Galis, F. Tusa, P. Monti, A. Sganbelluri, G. Biczók, B. Sonkoly, L. Toka *et al.*, "Analysis of end-to-end multi-domain management and orchestration frameworks for software defined infrastructures: an architectural survey," *Transactions on Emerging Telecommunications Technologies*, vol. 28, no. 4, 2017.

[6] R. Szabo et. al., "D2. 2: Final architecture," *UNIFY Project, Deliverable 2.2*, 2014.

[7] G. Xilouris, M.-A. Kourtis, M. J. McGrath, V. Riccobene, G. Petralia, E. Markakis, E. Palis, A. Georgios, G. Gardikis, J. F. Riera *et al.*, "T-nova: Network functions as-a-service over virtualised infrastructures," in *Network Function Virtualization and Software Defined Network (NFV-SDN), 2015 IEEE Conference on*. IEEE, 2015, pp. 13–14.

[8] "3gpp tr 28.801: Telecommunication management; study on management and orchestration of network slicing for next generation network." "www.3gpp.org/DynaReport/28801.htm".

[9] 5GEx Consortium, "5gex initial system requirements and architecture." [Online]. Available: http://www.5gex.eu/

[10] TNOVA Consortium, "Slas and billing." [Online]. Available: http://www.t-nova.eu/

[11] S. Previdi et.al., *BGP-LS Traffic Engineering (TE) Metric Extensions*.

[12] H. Gredler, J. Medved, S. Previdi, A. Farrel, and S. Ray, "North-bound distribution of link-state and traffic engineering (te) information using bgp," Tech. Rep., 2016.

[13] EU Project, *UNIFY Deliverable D3.2a, Network Function Forwarding Graph specification*.

[14] R. Szabo, Z. Qiang, and M. Kind, *Towards recursive virtualization and programming for network and cloud resources*.

[15] B. Martini, F. Paganelli, A. Mohammed, M. Gharbaoui, A. Sgambelluri, and P. Castoldi, "Sdn controller for context-aware data delivery in dynamic service chaining," in *2015 IEEE Conference on Network Softwarization (NetSoft)*, April 2015, pp. 1–5.