A Marketplace-based Approach to Cloud Network Slice Composition Across Multiple Domains

Paulo D. Maciel Jr.*, Fábio L. Verdi*, Polychronis Valsamas†, Ilias Sakellariou†, Lefteris Mamatas†, Sophia Petridou†, Panagiotis Papadimitriou†, David Moura‡, Asma Islam Swapna‡, Billy Pinheiro§, Stuart Clayman¶

* *Department of Computer Science*
Federal University of São Carlos - Sorocaba, Brazil
paulo.maciel@ifpb.edu.br, verdi@ufscar.br

† *Department of Applied Informatics*
University of Macedonia - Thessaloniki, Greece
{xvalsama, iliass, emamatas, spetrido, papadimitriou}@uom.edu.gr

‡ *Department of Computer Engineering and Industrial Automation*
University of Campinas - Campinas, Brazil
{dfcmoura, aiswapna}@dca.fee.unicamp.br

§ *Department of Computer Science*
Federal University of Pará - Belém, Brazil
billy@ufpa.br

¶ *Department of Electronic and Electrical Engineering*
University College London - London, UK
s.clayman@ucl.ac.uk

***Abstract*—Cloud network slicing can be defined as the process that enables isolated end-to-end and on-demand networking abstractions, which: (a) contain both cloud and network resources, and (b) are independently controlled, managed and orchestrated. This paper contributes to the vision of the NECOS project and relevant platform, that aim to address the limitations of current cloud computing infrastructures to accomplish the challenging requirements of the slicing approach. The NECOS platform implements the *Slice-as-a-Service* model, enabling the dynamic creation of end-to-end (E2E) slices from a set of constituent slice parts contributed from multiple domains. A challenging issue is to define the facility that implements dynamic slice resource discovery, aligned to the requirements of the slice owner or tenant, over different infrastructure providers. Here, we propose a Marketplace-based approach implementing relevant federated interactions for the resource discovery and we detail its architecture, workflows, and information model. We also present its initial implementation details and provide both quantitative and qualitative experimental results validating its main operation.**

***Index Terms*—Cloud Network Slicing, Slice-as-a-Service, Slice Resource Discovery, Marketplace**

I. Introduction

The need for supporting a variety of vertical industries, such as automotive, health-care, energy, and entertainment, is one of the main drivers of 5G systems. Different industries, with distinct requirements, will create various use case scenarios, in a broader range of offerings than existing services nowadays. Along these lines, *network slicing* is one of the key features of future 5G networks and a main enabler for the above challenging requirements. Network slicing has various definitions, but overall a network slice can be simply defined as end-to-end logical on-demand networks, relying on a common underlying infrastructure, comprised of physical and/or virtual resources, with independent control, management and orchestration. These end-to-end (E2E) self-contained networks must be mutually isolated from each other, and flexible enough in order to accommodate these simultaneous business-related use cases from different tenants on a shared infrastructure. Here, we use the term *cloud network slicing* to emphasize that slicing spans over both data center (i.e., cloud) and network resources.

This work is partially supported from the European Union's Horizon 2020 for research, technological development, and demonstration under grant agreement nº 777067 (NECOS - Novel Enablers for Cloud Slicing), as well from the Brazilian Ministry of Science, Technology, Innovation, and Communication (MCTIC), through RNP and CTIC. This work is also supported by the MESON (Optimized Edge Slice Orchestration) project, co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH CREATE - INNOVATE (project code: T1EDK-02947); the 4th open call scheme of the FED4FIRE+ (grant agreement nº 732638); and by a technical collaboration between the Federal University of São Carlos and the Federal Institute of Education, Science and Technology, Paraíba State, Brazil.

978-1-5386-9376-6/19/$31.00 ©2019 IEEE

The NECOS platform [1] aims to address the limitations of current cloud computing infrastructures to align to the *cloud network slicing* paradigm. *Slicing* is intended to be the foundation for a variety of scenarios, from the adoption of cloud computing in large networks of Telco Service Providers, to the utilization of edge clouds to support mobile devices with low computation and storage capacity. The NECOS platform incorporates several distinguishing features, the main ones being: the *Slice-as-a-Service* model for grouping sliced resources which are managed as a whole, and a Marketplace solution for dynamic slice resource discovery. The NECOS solution is based on the concept of *Lightweight Slice Defined Cloud* (LSDC), which extends the virtualization and VIM on-demand [2], [3] paradigms to all the networking and data center resources.

The LSDC system allows for the dynamic and run-time creation of complete E2E slices from a set of constituent slice parts. In the NECOS perspective, a slice is a full set of federated resources, so the platform should be able to locate slice parts from different infrastructure providers to build up a slice, while satisfying particular service constraints defined from the slice owner or tenant. However, instead of a pre-determined set of federated providers, a more flexible model from which slice parts can be provisioned is highly desired to support the challenging requirements of the vertical industries in 5G. Utilizing this dynamic approach, resource providers can decide *how* or *when* or *where* to supply resources required for the slice parts, in the form of data center resources (e.g., servers, storage, and network resources), or any other necessary resources (e.g., Internet of Things devices). This way, each provider can be capable of providing slice parts for E2E slices.

In this work, we detail a Marketplace-based slice resource discovery approach as an essential feature of the NECOS platform. In our proposal, different infrastructure providers participate in a slice resource Marketplace, offering resources dynamically to slice owners or tenants specifying particular service or cost requirements. The Marketplace utilizes the NECOS information model that includes a pricing model for the slice parts. The model enables interaction between the different stakeholders in the NECOS ecosystem and, therefore, it is an important resource discovery feature that should be accomplished. This process requires resource information to be exchanged between the slice composition components of NECOS and the resource providers. More specifically, we detail:

- the architecture of the proposed Marketplace and its main building blocks;
- the required unified information model that bridges the requirements of the slice owner or tenant, with the general slice specification and the alternative slice resource offerings from the infrastructure providers; and
- the initial implementation details of our proposal as well as experimental results validating its main technical directions.

The remainder of this paper is structured as follows. Section II presents a state of the art investigation to underline the value of our contribution. A short overview of the NECOS architecture and its Marketplace components are given in Section III. Section IV describes the main ideas from the Marketplace point of view for the slice resource discovery. Section V presents our experimental evaluation scenarios and preliminary results. Section VI concludes the paper by presenting final remarks and future directions.

II. State of the Art

The idea of the marketplace has been entertained in diverse contexts in the cloud and networking research. For instance in [4], a marketplace ecosystem of VNFs is described, where users can discover and execute VNFs and compose service function chains (SFCs). The main mechanism for VNF discovery presents similarities with that of Google Play and Apple Store, i.e., a trusted (reviewed) repository of VNFs that users instantiate and execute on the provided infrastructure layer. In [5] the notion of a marketplace is used in a very similar vein, i.e., a repository where a variety of developers can publish their VNFs offered to customers for selection. Both approaches provide the infrastructure necessary to locate and execute VNFs. The NECOS Marketplace aims at a different level: it discovers physical resources with meeting specific compute and connectivity constraints to create a slice to deploy tenant services.

5G!Pagoda [6] proposed a holistic approach, to the creation of a network slice in a hierarchical fashion. Slice creation considers any specific application/service requirements described in the tenants request, while slice provisioning utilizes major technological enablers (i.e., SDN, NFV, cloud computing). The architecture is service oriented, aiming at mapping different slices to virtual resources, however dealing with a dynamic set of physical resources is not addressed, to the best of our knowledge.

The 5G NORMA project [7] proposes a business model of network slicing with an optimized 5G infrastructure market [8]. The project introduces an admission control schema accepting new slices to the market based on the resource availability, while bringing network infrastructure providers and the network slices' tenants under a marketplace ecosystem [9]. The marketplace platform of 5G NORMA follows a vertical marketing model and constitutes of two players: (i) the infrastructure providers (IPs) offering on-demand the slice resources; and (ii) the tenants with rights to acquire requested network slices [7]. The concept of 5G NORMAs business model focuses on resource allocation mechanisms by controlling market players, i.e., IPs and Tenants, admissibility into the market with a view to enhance performance and revenues of both of the parties. 5G NORMA introduces the notion of a brokerage model residing at the infrastructure providers, multi-tenancy, multi-service support and a controlled access to the market. Our Marketplace proposal enables dynamic slice deployment and elasticity over multi-domain resources dynamically offered from different infrastructure providers,

based on the service and cost requirements of the slice user or tenant.

ONAP [10] introduces a reference implementation of a Marketplace in the form of VNF, SDKs and APIs in providing policy-driven orchestration and automation of physical and virtual network functions. The model incorporates two actors inside the marketplace platform: i) the vendors, ready to provide VNF and network services; and ii) the operators, capable to perform queries on VNFs already available in the Marketplace. Once uploaded by the vendors (or suppliers), each VNF is validated within the Marketplace platform automatically and therefore is enlisted as available in the market for purchase [11]. The Marketplace provides an on-boarding report for each VNF, for the operators to test and validate them before the purchase. The orientation of ONAPs marketplace architecture is rooted on a catalog model, as the orchestration platform provides an Active and Available Inventory (AAI) model based on resource cartography. AAI stores information regarding the resources and services of a single provider and tenant present in the marketplace [12]. Unlike ONAP, the NECOS marketplace considers resource requests and their matching resources from a diverse set of tenants and providers while being an essential part of the slice deployment and operation process.

The 5GTANGO [13] project proposes an integrated vendor-independent platform, where a packaged NFV forwarding graph of composed services is automatically tested and validated. In 5GTANGO the Marketplace appears as the concept of a Store with a customizable orchestrator, network slice manager and slice-to-network-service-mapper, compatible with common existing Virtual Infrastructure Managers (VIM) and slice controllers [14]. Taking into account vertical application requirements and defined SLAs, 5GTANGO follows complex resource allocation schemes, fitted into dedicated network slice blueprints that are suitable for vertical industries. The primary players are indicated as the Service/Infrastructure Provider, the end-user that could be a developer of VNFs or network services (NSs), or/and the service consumer [15]. The proposed framework employs artificial neural networks (ANNs) and acts as a mediator between the service providers and the end-users [16]. In our case, the NECOS Marketplace approach implements a resource discovery schema leveraging a brokerage model inside the platform.

The [17] project introduces a service platform for both providers (i.e., network operators) and developers through an integrated DevOps model to locally prototype and test complete service function chains [17]. The tight integration between the two main building blocks of SONATA architecture (i.e., the service platform and the software development toolkit) offers flexibility and convenience in adding management and orchestration functionalities on-the-fly. The SONATA service platform introduces a brokerage system that exchanges information between the loosely coupled components of the platforms (i.e., aligned to the Micro-services paradigm) [18]. The status information on the running network services and functions is held in a catalog system accessed by

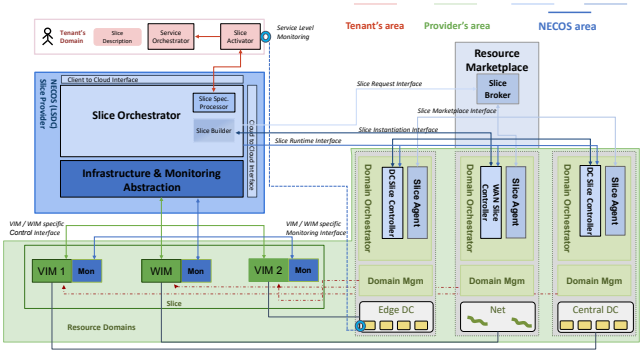

Fig. 1. Abstract view of the NECOS architecture.

the brokers and stakeholders of the service platform. However, the idea of a generic Marketplace and the direct communication between the network operators and the developers are beyond the project's scope [18].

T-NOVA [5], [19] designed and implemented an orchestration platform for virtualized network functions (VNFs). In this context, T-NOVA developed a Marketplace, allowing clients to browse and select virtual appliances, facilitating the composition of network services out of selected VNFs. To enable the submission of service requests from clients, T-NOVA defined a description model for VNFs and network services. A brokerage module further carries out the selection of VNFs across multiple Points-of-Presence from a single NFV provider. Since T-NOVA does not account for multi-provider NFV deployments, the applicability of its marketplace has a limited scope compared to the proposed Marketplace of NECOS.

III. The NECOS Architecture for Slicing

The NECOS architecture was defined and encompasses the components necessary for the provisioning of E2E slicing in a multi-provider and multi-technological federation of domains. An abstract view of the architecture with particular emphasis on the *Resource Marketplace* component is shown in Fig. 1.

To facilitate the comprehension of the NECOS architecture, which is extensively described in [20], in Fig. 1 we define three distinct areas, i.e., the tetant's rose area, the providers' green area and the core NECOS blue area. The last one is further separated to emphasize the three main functional blocks of NECOS. Architecturally, NECOS consists of: (i) the *Slice Orchestrator* block in light blue, (ii) the *Infrastructure & Monitoring Abstraction (IMA)* block in dark blue, and (iii) the *Resource Marketplace* block in gray blue. A set of sub-components inside the *Slice Orchestrator* and the *IMA* blocks are defined to provide the functionality needed to support the E2E slice lifecycle, e.g., slice orchestration, management and monitoring [20].

Briefly, the *Slice Orchestrator* performs operations to deal with slices, i.e., (i) requests from the *Resource Marketplace* the different slice parts that will be included in an E2E slice via performing an initial orchestration phase, in order to work

out a subset of domain that could be used as candidates for the E2E slice creation; (ii) stitches the allocated slice parts into a single aggregated slice on which it holds the overall end-to-end view and manages the lifecycle of each individual slice part. In addition, it is also responsible for orchestrating the service elements across the slice parts that make up the full E2E slice as it performs the actual placement and embedding of VMs into the resource domains. Furthermore, it takes care of the lifecycle management of the services running on the slices based on the information retrieved from the *IMA* component.

Indicated by its name, this component provides a uniform abstraction layer above the heterogeneous Virtual Infrastructure Manager (VIM) / Wide-area network Infrastructure Manager (WIM) and monitoring subsystems that are part of an E2E slice. Since different slice parts will constitute an E2E slice, each part can potentially rely on a different technology (e.g., specific VIM / WIM and monitoring subsystem implementations). Thus, *IMA* plays a focal role in the NECOS architecture offering an abstract northbound interface which allows the *Slice Orchestrator* to perform its functions while the slice parts details remain agnostic to it, e.g., collecting information about the resource topology and resource monitoring information regarding each slice part; monitoring and verifying the status of the virtual elements allocated to each slice part; gathering resource facing KPIs (such as CPU, memory and storage) for the virtual service elements running on the slice parts. To achieve that, multiple adapters are allocated which, in fact, translate the requests coming through the northbound interface into the particular VIM or WIM APIs. This way they hide the slice parts' heterogeneity at the southbound of the *IMA*.

Supposing that the *Slice Orchestrator* requests and initiates slices on behalf of the tenants, while the *IMA* contributes on their management and monitoring once the slices are instantiated, in-between lies the need to discover the resources' that get compose an E2E slice. This task is assigned to the *Resource Marketplace* component which is a fully-distributed system responsible to locate suitable slice parts from a set of participating resource domains. As depicted in Fig.1, the marketplace consists of two main NECOS sub-components, namely the *Slice Broker* and a number of *Slice Agents*. The first interacts eastbound with the *Slice Builder*, which could be considered the NECOS sub-component between the *Slice Orchestrator* and the *Resource Marketplace*. The *Slice Builder* actually transits the slice request to the broker and receives back relevant offers for slice parts. The *Slice Broker* discovers these slice parts by interacting with a set of *Slice Agents* hosted by the involved resource domains. We further elaborate on the *Resource Marketplace* right afterwards.

IV. Marketplace for resource discovery

The resource discovery process is essentially composed of a set of workflows for information exchange including slice requests, resource requests, and resource offerings. Several NECOS components are involved in the discovery process, such as the *Slice Builder*, the *Slice Broker*, and the *Slice Agents*. In this section, we present: (i) the main requirements that must be accomplished for resource discovery in the Marketplace; (ii) the proposed information models for slice specification and infrastructure description; and (iii) the Marketplace components and associated workflows.

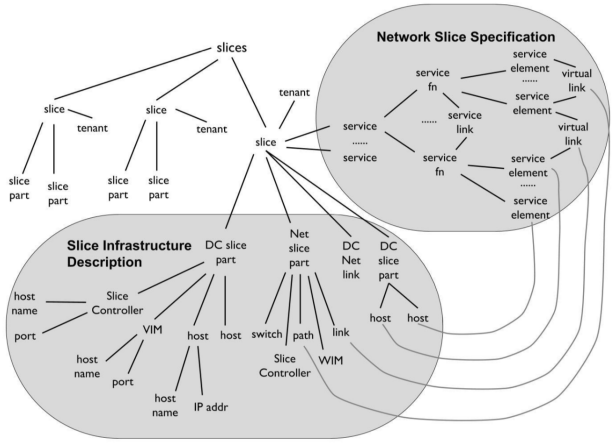

Fig. 2. Overview of NECOS information model [21].

Based on the context of the Marketplace, we consider the following requirements for the slice resource discovery framework. The *Slice Agents* (in coordination with *DC/WAN Slice Controllers*) should be able to evaluate the feasibility of hosting a part of the overall slice, as well as have the capability of scaling allocated slice parts in response to evolving service demands. Furthermore, the *Slice Builder* should be able to convey topology and resource slice related requirements to the *Slice Broker* using predefined message templates.

The *Slice Builder* and *Slice Broker* should have the capability to expose the slice control and configuration services to the authorized consumers (i.e., infrastructure providers / tenants), as a way to resolve potential conflicts. Special *Slice Broker* messages may be addressed to specific *Slice Agents*, for the purposes of fault, performance, provisioning, and access management.

A. Information model

The NECOS information model aims at providing a unified description of all information regarding a slice, that will be used for slice specification (i.e., from the tenant), provisioning and run-time management. Thus a detailed description of: (i) slice parts, allocated infrastructure resources, and their properties; and (ii) services decomposed to service elements along with the necessary resource demands, deployed to these parts. It should be noted that the model acts as a "blackboard" during the slice lifecycle, where different components "fill in" the missing details of the slice. For instance, the *Slice Builder* will fill in the details of a selected *DC Slice Controller* processing alternatives returned by the Marketplace, while the *Slice Specification Processor*, refines the tenant's request for hosting a service, by mapping service elements to a desired slice part. The model offers a very flexible slice request

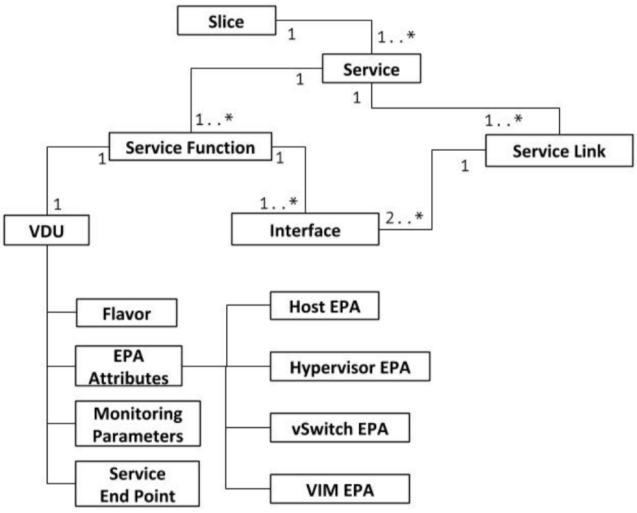

Fig. 3. Slice specification with the NECOS information model [21].

specification on the tenant's side: the tenant might specify both the number of slice parts and their associated requirements, or specify the service elements along with general resource requirements; letting the NECOS system decide on the necessary infrastructure that could host the service, or a mixture of both. The proposed information model has been inspired by relevant information models, such as COMS (Common Operations and Management on network Slices) [22] and ETSI NFV MANO [23].

As depicted in Fig. 2, a slice description consists of the *Slice Infrastructure Description* and the *Network Slice Specification*. The former supports fields storing provider-specific information for DC and WAN slice parts; necessary for the slice instantiation and management (such as the IP addresses of the *Slice Controllers*), i.e., the slice infrastructure graph. Each slice part is linked to one or more service elements it hosts in the *Network Slice Specification*.

The *Network Slice Specification* stores the specification of the service to be hosted by the slice. This includes a decomposition of the former to service elements and the necessary links among them, as depicted in Fig. 3. The *Virtual Deployment Unit (VDU)* object associates a service function with resource requirements, a range of Extended Platform Awareness (EPA) [23] attributes, and monitoring parameters. For instance, the *Host EPA* defines resource parameters such as minimum storage requirements ("storage-gb") and memory (RAM), whereas the *VIM EPA* allows the tenant to express preferences for the VIM deployed (e.g., OpenStack or Kubernetes). A more detailed description of the objects and attributes of the information model for the slice specification escapes the current context of the paper, however it is available at [21].

Thus, the model offers great flexibility in defining the characteristics of the desired infrastructure where the services need to be deployed. The role of the Marketplace is to process this specification, discover and report alternative providers that are able to host each slice part.

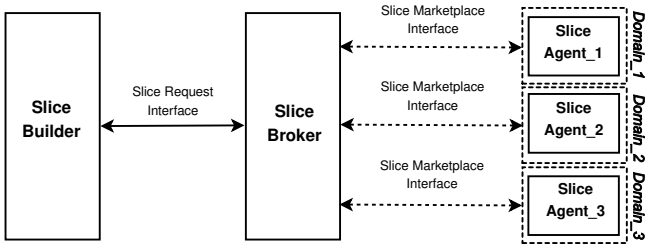

Fig. 4. An abstract view of the resource discovery workflow.

B. Marketplace components and workflows

The resource discovery framework proposed in the NECOS architecture [20] is responsible for locating the appropriate resources that compose a slice, i.e., slice components that correspond to service functions and service links, according to the information model discussed previously. In a nutshell, a *Partially Defined Template (PDT)* message defines the general slice requirements and acts as the input to the resource discovery framework. This message actually originates from the *Slice Specification Processor* (Fig. 1) and is passed to the *Slice Builder* that is closely coupled with the former. The *Slice Broker* is responsible for locating resources from both DC and WAN providers to fulfill the slice requirements and prepare a corresponding response, namely a *Slice Resource Alternatives (SRA)* message. In practice, the SRA message annotates the PDT message with alternative slice component options.

More precisely, the aforementioned process employs the following three architectural functional components of NECOS, depicted in Fig. 4:

- the *Slice Builder*, which is responsible for initiating the slice resource discovery process, by forwarding the PDT message to the *Broker*, and selecting the most appropriate slice components, among alternatives returned by the latter in the form of an SRA message. The PDT consists of DC slice parts, annotated with computing resource constraints, and WAN slice parts that describe the desirable connections among providers.
- the *Slice Broker*, which decomposes the template it receives from the *Builder* and creates a different query for each slice part. Given the structure of the PDT message, such decomposition can be performed easily, since each slice part corresponds to a different resource provider. The *Slice Broker* has all the necessary information to form query messages that contain all the constraints/preferences/resources needed for the component request message. Once the PDT has been decomposed and submitted, the *Broker* proceeds with collecting all alternative responses and responding back to the *Builder*. Responses for each alternative slice part (both dc-slice parts and network-slice parts) are, in fact, lists of al-

ternative resources originating from the providers' *Slice Agents*. Since each agent supplies references to the offered slice parts, along with cost and other information, the *Builder* is in position to select the configuration of the slice that best matches the tenant's needs.
- a set of *Slice Agents* that reside on the providers' domains. The role of an *Slice Agent* is to answer requests for resources that originate from the *Slice Broker* and regard specific dc- or wan-slice part. This message is translated in a form that the corresponding DC or WAN provider can process (i.e., to lookup the requested resources through its own provider-specific resource directory - a task carried out from the particular *DC/WAN Domain Controller*) and the answer received from the controller is passed to the *Slice Broker*.

In section that follows, we highlight the main implementation details of our Marketplace proposal and present our initial experimental results.

V. Experimental Evaluation

A. Description

This section presents a preliminary experimental evaluation of the proposed NECOS Marketplace approach to handling a slice request over different infrastructure providers. Our experimental setting consists of six different test-beds participating in the FED4FIRE federation [24]: (i) w-iLab2, Virtual Wall 1 (VWall1), Virtual Wall 2 (VWall2) and Grid5000 test-beds, which are located in Europe; and (ii) CloudLab test-beds in Utah (ClabUtah) and Wisconsin (CLabWisconsin), i.e., located in the USA. For the purposes of the experiment, each test-bed is considered as a different infrastructure provider, with its own *Slice Agent* responding to slice part resource requests. We collected resource infrastructure data from the above test-beds, i.e., regarding node availability and resource characteristics, such as memory, storage and bandwidth capacities, number of CPU cores, etc. That data form the resource infrastructure information base of each agent, which is the information used by each *Slice Agent* to match particular resource requests incorporating extended EPA characteristics, as they are specified in the information model detailed in Section IV-A.

To evaluate the proposed Marketplace facility, we generated slice requests that consist of a varying number of slice parts, each composed of one or multiple service function resource specifications, referred to as *dc-vdu* (*Data Center Virtual Deployment Unit*). Each *dc-vdu* specification is characterized by particular resource request demands, i.e., number of physical nodes, amount of RAM, disk storage, bandwidth capabilities, etc. Since we assume that each test-bed serves as a single data center (DC) infrastructure provider, it can accommodate only one (DC) slice part. The test-bed resources are organized in clusters of similar in characteristics computing nodes, so we make the assumption that each *dc-vdu* can be accommodated to a single node cluster. Thus, the question that each *Slice Agent* answers is whether it can accommodate or not the request received by the Broker for a single slice part, allocating each *dc-vdu* of the part to possibly different node clusters in its infrastructure.

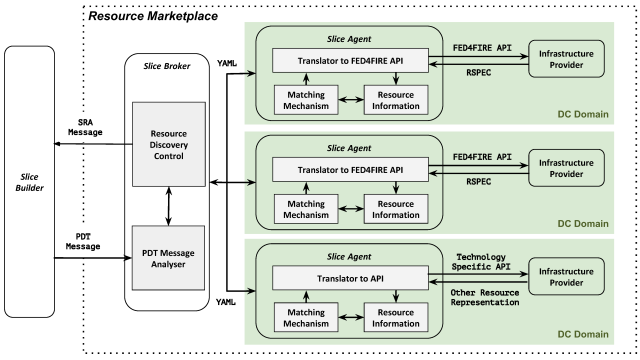

Fig. 5. An implementation view of the Marketplace.

B. Marketplace Implementation Details

Fig. 5 shows the implementation details of the proposed Marketplace facility that includes the main architectural components of the *Slice Broker* and the *Slice Agents*, as well as the messages exchanged during the slice resource discovery process. We now briefly outline the required components we implemented along with example messages exchanged, i.e., to realize the functionality described in Section IV-B. A more complete relevant message exchange example can be found in the NECOS deliverable D4.1 [21].

The *Slice Broker* upon the reception of the PDT message, decomposes the latter to a number of requests, one for each slice part. As mentioned, such a request must contain all the necessary information, i.e., resource characteristics' demands for each part, which are not all defined in the same section of the PDT message. This occurs since different components refine the original tenants request for a slice: for instance, the *Slice Activator* is responsible for specifying characteristics such as EPA attributes, where the *Slice Specification Processor* decides on the slice graph, i.e., the number of slice parts and *vdus* residing in each part. The role of the *PDT Message Analyser* component is to aggregate all that information, forming a complete request message, in terms of necessary demand information. Follows an example slice part request for a particular *dc-vdu* (i.e., a `web_server_VM`):

```
"dc-slice-part": { "name": "dc-slice1",
  "vdus":{[
     {"dc-vdu": {
       "id":"web_server_VM",
       "description":"web-servers
            for elastic CDN deployment",
       "host-count-in-dc": { "equal": 5 },
            ... }
       },...]}
```

The requested EPA attributes are defined in the service function section of the PDT message, as shown below:

```
"service-function": {
   "service-element-type": "vdu",
     "vdu":{"id": "web_server_VM",
            "epa-attributes":
               {"host-epa":
                 {...,
                  "storage-gb": 2,
                  "memory-mb": 4096,...}
                  } ...
```

The role of the *PDT Message Analyser* is to collect all the necessary information in the request message for the resource discovery, i.e., filtering out the unneeded data. For instance, the following presents an aggregated request message for a particular slice part:

```
"dc-slice-part":
   {"name": "dc-slice1",
           ...
    "vdus":{[
     {"dc-vdu": {
       "id":"web_server_VM",
       "host-count-in-dc": 5,
       "storage-gb": 2,
       "memory-mb": 4096 , ...},
     {"dc-vdu": ...}
        ]}
```

The textual constraints (such as `equal:`) in the PDT message get translated into a suitable form for the *Slice Agents* to directly process them. Our implementation relies on the advanced pattern matching and symbolic manipulation characteristics of Prolog to perform this task. The output of this process is directed to the *Resource Discovery Control* component to initiate the discovery process.

The *Resource Discovery Control* component is responsible to query all agents in the marketplace for each slice part and collect all the responses. In the current implementation, this component forms all alternative slices (i.e., the slice part allocations to the different providers) using the backtracking mechanism of Prolog.

To retrieve the real-time resources' status from the test-beds, the *Slice Agent* is equipped with a Python *Translator* component. This component communicates directly with each test-bed (i.e., DC domain) using the FED4FIRE API. The test-beds respond with the status of their available resources in an RSpec format [25]. The component is also responsible for translating the response message in a more coherent format, in compliance to the NECOS information model. It then forwards the message to the resource *Matching Mechanism* component. In order to discover which cluster can accommodate each slice part, the *Matching Mechanism* component translates the request originating from the *Slice Broker* to a set of resource availability constraints the part should satisfy. For instance, the previous message is translated to the following:

```
dc_part('dc-slice1',location:undefined,[
          'dc-vdu'('web_server_VM',
                   ['available-nodes'>= 5 ,
                    'min-storage-gb'>= 2,
                    'memory-gb'>=4 ]),
        'dc-vdu'(...),
           ]).
```

This allows a simple matching process that discovers all *dc-vdus* a provider can host in a node cluster. The output of this initial matching action forms a (simple) resource allocation problem (i.e., *dc-vdu's* to node cluster's) that the *Slice Agent* solves to answer positively to the request. Although we currently employ a simple backtracking process to avoid overtaxing a cluster with more *dc-vdu's* it can manage due to node availability constraints, a constraint logic programming approach could solve the problem much more efficiently, i.e., minimizing certain optimization criteria stated by the infrastructure provider. Upon success, the *Slice Agent* fills the PDT request message with its offering and responds it back to the *Slice Broker*.

C. Experimental Results

We organize our experimental evaluation in three series of trials to investigate the solution space of the Marketplace based on various requested slices, i.e., the first two with quantitative and the third with qualitative results. In the first series, we adjust the slice request by increasing the number of physical nodes within a *dc-vdu*, the latter annotated with a single resource specification, i.e., the memory size (RAM). In the second series of experiments, we generate slice requests of similar demands in node availability, but with three resource specifications for each *dc-vdu* (i.e., RAM, disk storage and bandwidth capabilities). Finally, the last experiment involves a cloud core / edge slice request example and demonstrates its geographical distribution.

In the first and second series of results, we validate the overall functionality of the marketplace and obtain an idea on the size of the solution space. Essentially, we vary the demands in slice requests as described above and compare the number of alternative slice solutions for scenarios with a different number of slice parts (*SP=[2,4]*), and *dc-vdus* per slice part (*VDUs/SP=[1,3]*).

In the first series results that are depicted in Fig. 6, the number of alternative solutions decreases as the requested nodes per *dc-vdu* increases. Such a behaviour is expected, as the number of providers that can accommodate the slice part decreases. Note that the number of parts plays a significant role to the solutions, since it allows for more combinations of providers (i.e., alternatives) to be generated.

Similar results hold for the second series of experiments, as depicted in Fig. 7. An interesting point to note here is that in slices with a high number of parts and an increased set of resource requirements, the decrease in the number of solutions is sharp. This is attributed to the fact that very few infrastructure providers can accommodate resource-demanding slice parts.

It is important to emphasize again that, from the first two series results shown in Figs. 6 and 7, we can notice the

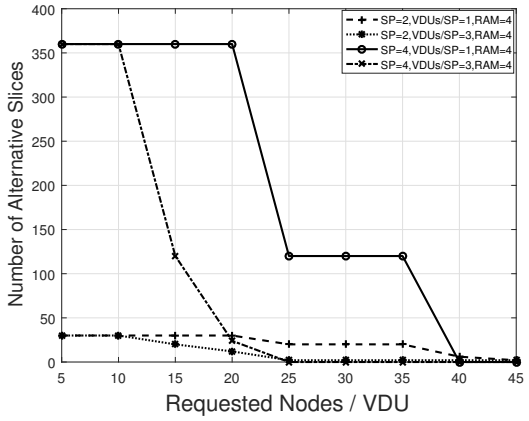

Fig. 6. Number of alternative slices with one resource requirement.

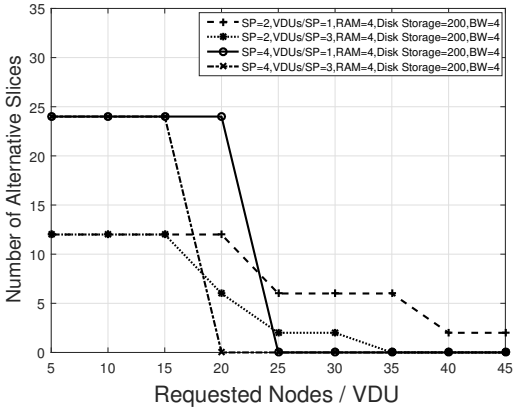

Fig. 7. Number of alternative slices with three resource requirements.

curves in both graphs behave the same, where the number of alternative solutions decreases as the demand for requested nodes increases. Even with the increase in slice parts, which provides more alternative combinations, the number of possible slices drops significantly from 20 requested nodes per *dc-vdu* onwards. Finally, as also expected, the more requirements for resources (three instead of one), the less is the number of alternative slices.

The final experiment aims to demonstrate the allocation of a slice to a set of core / edge cloud providers. The slice consists of four slice parts, each hosting two *dc-vdus*. The slice parts have diverse resource demands, reflecting their role in the slice, i.e., higher demands for the core cloud slice resources and lower demands for the edge cloud resources. The following example presents the constraints for a core cloud slice:

```
dc_part(dc-slice-1,location:'undefined',
 ['dc-vdu'(vdu11,[
   'available-nodes'>=2,'memory-gb'>64,
   'min-storage-gb'> 250,'nics-bw'> 2]),
  'dc-vdu'(vdu12,[
   'available-nodes'>=1,'memory-gb'>=128,
   'min-storage-gb'>=500,'nics-bw'>=5])
 ])
```

TABLE I
SLICE PART ALLOCATION TO DIFFERENT INFRASTRUCTURE PROVIDERS.

Slice Part 1	Slice Part 2	Slice Part 3	Slice Part 4
Grid5000	CLabUtah	CLabWisconsin	VWall2
Grid5000	CLabUtah	VWall1	VWall2
Grid5000	CLabUtah	w-iLab2	VWall2
Grid5000	CLabWisconsin	CLabUtah	VWall2
Grid5000	CLabWisconsin	VWall1	VWall2
Grid5000	CLabWisconsin	w-iLab2	VWall2
Grid5000	VWall1	CLabUtah	VWall2
Grid5000	VWall1	CLabWisconsin	VWall2
Grid5000	VWall1	w-iLab2	VWall2

whereas the example indicated below shows the constraints for a slice part corresponding to an edge cloud, where resource are less demanding:

```
dc_part(dc-slice-2,location:'undefined',
 ['dc-vdu'(vdu21,[
   'available-nodes'>=1,'memory-gb'>=8,
   'min-storage-gb'>=80,'nics-bw'>=1]),
  'dc-vdu'(vdu22,[
   'available-nodes'>=1,'memory-gb'>=8,
    'min-storage-gb'>=80,'nics-bw'>=1])
 ]),
```

Similar constraints exist in the other two parts. In the current experimental setup, the marketplace generated nine alternative allocations for the slice, as depicted in Table I. Note that since the request did not define any geographic constraints on the slice parts, the marketplace generated all possible solutions, distributing slices between Europe (Grid5000, w-iLab2, VWall1 & VWall2) and USA (CLabUtah & CLabWisconsin). This demonstrates that such a loosely coupled resource discovery model can manage a diverse set of geographically distributed providers.

VI. Conclusions and Future Work

In this paper, we highlight the NECOS Marketplace approach handling slice requests over multiple infrastructure providers. We describe in detail the main Marketplace components and their associated workflows, implementing dynamic slice resource discovery. We also carried out real experiments utilizing measurements on the resource availability of different open-access test-beds. Our experiments: (i) validate the proposed facility and highlight the magnitude of the solution space for the complete slice offerings; and (ii) provide an example of slice resource discovery utilizing geographically-distant resources over both core and edge clouds.

The Marketplace approach faces interesting challenging issues, including:

- The investigation of the scalability capabilities of the proposed slice resource discovery facility, e.g., accommodating large numbers of parallel requests and more infrastructure providers.
- Extend the proposed marketplace facility to accommodate alternative business models (e.g., open-access marketplaces and marketplaces for closely-cooperating infrastructure providers).
- Investigate algorithms for the selection of appropriate slice parts based on various cost models.
- Experiment with slice resource discovery workflows for slice elasticity and the involved performance trade-offs.

REFERENCES

[1] F. S. D. Silva, M. O. O. Lemos, A. Medeiros, A. V. Neto, R. Pasquini, D. Moura, C. Rothenberg, L. Mamatas, S. L. Correa, K. V. Cardoso, C. Marcondes, A. ABelem, M. Nascimento, A. Galis, L. Contreras, J. Serrat, and P. Papadimitriou, "Necos project: Towards lightweight slicing of cloud federated infrastructures," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, June 2018, pp. 406–414.

[2] S. Clayman, F. Tusa, and A. Galis, "Extending Slices into Data Centers: the VIM on-demand model," in *2018 9th International Conference on the Network of the Future (NOF)*, Nov 2018, pp. 31–38.

[3] L. A. Freitas, V. G. Braga, S. L. Corrła, L. Mamatas, C. E. Rothenberg, S. Clayman, and K. V. Cardoso, "Slicing and allocation of transformable resources for the deployment of multiple virtualized infrastructure managers (vims)," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, June 2018, pp. 424–432.

[4] L. Bondan, M. F. Franco, L. Marcuzzo, G. Venancio, R. L. Santos, R. J. Pfitscher, E. J. Scheid, B. Stiller, F. D. Turck, E. P. Duarte, A. E. Schaeffer-Filho, C. R. P. dos Santos, and L. Z. Granville, "FENDE: Marketplace-Based Distribution, Execution, and Life Cycle Management of VNFs," *IEEE Communications Magazine (COMMAG)*, vol. 57, no. 1, pp. 13–19, jan 2019.

[5] G. Xilouris, E. Trouva, F. Lobillo, J. M. Soares, J. Carapinha, M. J. McGrath, G. Gardikis, P. Paglierani, E. Pallis, L. Zuccaro, Y. Rebahi, and A. Kourtis, "T-NOVA: A marketplace for virtualized network functions," in *European Conference on Networks and Communications, EuCNC 2014, Bologna, Italy, June 23-26, 2014*, 2014, pp. 1–5. [Online]. Available: https://doi.org/10.1109/EuCNC.2014.6882687

[6] I. Afolabi, A. Ksentini, M. Bagaa, T. Taleb, M. Corici, and A. Nakao, "Towards 5G network slicing over multiple domains," *IEICE Transactions on Communications, Special section on Network Virtualization, Network Softwarisation, and Fusion Platform of Computing and Networking. Vol 100B, N11, November 2017*, 11 2017. [Online]. Available: http://www.eurecom.fr/publication/5375

[7] K. Katsalis, N. Nikaein, E. Schiller, A. Ksentini, and T. Braun, "Network slices toward 5g communications: Slicing the lte network," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 146–154, 2017.

[8] Z. Yousaf, "D5.1 Definition of connectivity and QoE/QoS management mechanisms intermediate report," *5GNOMA Project Deliverable*, 09 2016. [Online]. Available: http://www.it.uc3m.es/wnl/5gnorma/deliverables/

[9] M. Breitbach, "D7.2: Communication and dissemination final report," *5GNORMA Project Deliverable*, 12 2017. [Online]. Available: http://www.it.uc3m.es/wnl/5gnorma/deliverables/

[10] Open network automation platform. [Online]. Available: https://www.onap.org/

[11] Developer wiki: Open network automation platform. [Online]. Available: https://wiki.onap.org/

[12] A. Boubendir, F. Guillemin, C. Le Toquin, M.-L. Alberi-Morel, F. Faucheux, S. Kerboeuf, J.-L. Lafragette, and B. Orlandi, "Federation of cross-domain edge resources: a brokering architecture for network slicing," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*. IEEE, 2018, pp. 415–423.

[13] P. Twamley, M. Müller, P.-B. Bök, G. K. Xilouris, C. Sakkas, M. A. Kourtis, M. Peuster, S. Schneider, P. Stavrianos, and D. Kyriazis, "5gtango: An approach for testing nfv deployments," in *2018 European Conference on Networks and Communications (EuCNC)*. IEEE, 2018, pp. 1–218.

[14] E. Kapassa, M. Touloupou, A. Mavrogiorgou, and D. Kyriazis, "5g & slas: Automated proposition and management of agreements towards qos enforcement," in *2018 21st Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*. IEEE, 2018, pp. 1–5.

[15] M. Zhao, F. Le Gall, P. Cousin, R. Vilalta, R. Muñoz, S. Castro, M. Peuster, S. Schneider, M. Siapera, E. Kapassa *et al.*, "Verification and validation framework for 5g network services and apps," in *2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. IEEE, 2017, pp. 321–326.

[16] E. Kapassa, M. Touloupou, and D. Kyriazis, "Slas in 5g: A complete framework facilitating vnf-and ns-tailored slas management," in *2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA)*. IEEE, 2018, pp. 469–474.

[17] S. Dräxler, H. Karl, M. Peuster, H. R. Kouchaksaraei, M. Bredel, J. Lessmann, T. Soenen, W. Tavernier, S. Mendel-Brin, and G. Xilouris, "Sonata: Service programming and orchestration for virtualized software networks," in *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2017, pp. 973–978.

[18] J. Bonnet, "D4.3. Service Platform First Operational Release and Documentation," *SONATA Project Deliverable*, 07 2017. [Online]. Available: http://www.sonata-nfv.eu/deliverables/

[19] M. A. Kourtis, M. J. McGrath, G. Gardikis, G. Xilouris, V. Riccobene, P. Papadimitriou, E. Trouva, F. Liberati, M. Trubian, J. Batall, H. Koumaras, D. Dietrich, A. Ramos, J. F. Riera, J. Bonnet, A. Pietrabissa, A. Ceselli, and A. Petrini, "T-nova: An open-source mano stack for nfv infrastructures," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 586–602, Sept 2017.

[20] S. Clayman, "D3.1: NECOS System Architecture and Platform Specification. V1," *NECOS Project Deliverable*, 10 2018. [Online]. Available: http://www.h2020-necos.eu/documents/deliverables/

[21] P. Papadimitriou, "D4.1: Provisional API and Information Model Specification. V1," *NECOS Project Deliverable*, 10 2018. [Online]. Available: http://www.h2020-necos.eu/documents/deliverables/

[22] L. Qiang, A. Galis, L. Geng, K. Makhijani, P. Martinez-Julia, H. Flinck, and X. de Foy, "Technology Independent Information Model for Network Slicing," Internet Engineering Task Force, Internet-Draft draft-qiang-coms-netslicing-information-model-02, Jan. 2018, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-qiang-coms-netslicing-information-model-02

[23] ETSI Industry Specification Group (ISG) NFV, "GS NFV-IFA 013 - V2.3.1: Network Functions Virtualisation (NFV) Release 2; Management and Orchestration; Os-Ma-Nfvo reference point - Interface and Information Model Specification," ETSI, Tech. Rep., 2017. [Online]. Available: http://www.etsi.org/standards-search

[24] T. Wauters, B. Vermeulen, W. Vandenberghe *et al.*, "Federation of Internet experimentation facilities: architecture and implementation," in *European Conf. on Networks and Communications (EuCNC)*, Jun. 2014, pp. 1–5.

[25] M. Berman, J. S. Chase, L. Landweber *et al.*, "Geni: A federated testbed for innovative network experiments," *Computer Networks, Elsevier*, vol. 61, pp. 5–23, Mar. 2014.