WIM on-demand – A modular approach for managing network slices

Stuart Clayman, Francesco Tusa, Alex Galis
Dept. of Electronic Engineering, University College London, London, UK
Email: s.clayman@ucl.ac.uk, francesco.tusa@ucl.ac.uk, a.galis@ucl.ac.uk
Luis M. Contreras
Telefonica, Madrid, Spain
Email: luismiguel.contrerasmurillo@telefonica.com

***Abstract*—This paper explores and makes a case for allocating a Wide-area Infrastructure Manager (WIM) on-demand to support softwarized network slicing, as part of the full NFVI virtualised infrastructure foundation, to ensure that the connectivity attributes prescribed to network slices can be managed with flexibility and adaptability in a full end-to-end slice. We show how creating a WIM on-demand and dynamically allocating a new WIM for each network slice, rather than having one for the whole network, can be beneficial for various slicing scenarios, in a similar way that a Virtual Infrastructure Manager (VIM) on-demand has been utilized. The paper considers some of the components, abstractions, and mechanisms of WIM on-demand.**

***Index Terms*—network slicing, infrastructure slicing, WIM on-demand, VIM on-demand**

I. Introduction

5G networks are currently being designed to meet the requirements of a highly mobile and fully connected society, and to be both application-aware and service-aware [1]. They represent: (i) an evolution, over 4G networks, in terms of capacity, performance, and spectrum access in radio network segments; and (ii) an evolution of flexibility and programmability in both the radio and non-radio 5G network segments from the edge to the core of the infrastructure. As a consequence, there is currently a lot of effort related to the *softwarized* network, the *programmability* of the network, and the huge scale of 5G in terms of possible connected devices, and the infrastructure that will span multiple technological resource domains, even across different administrative borders [2].

The concepts of network softwarisation, programmability, resource heterogeneity, as well as precision networks, are currently being considered by the Network 2030 Focus Group [3], who are studying the capabilities of networks for the year 2030 and beyond, when networks are expected to support novel forward-looking scenarios, such as holographic type communications, extremely fast response in critical situations and high-precision communication demands of emerging market verticals. As a consequence, 5G and Network 2030 will require the current problems in resource management to be solved to allow the new enhancements to be brought in.

The 5G design approach relies on the concepts of softwarisation and programmability as the basis to build real systems with a new configurable data plane [4] which supports slicing. The envisaged configurable *sliced* data plane will have many new functions, and over these slices there will be new precision services, multi-way delivery, with timeliness, scale, and qualitative guarantees. A *Slice* is the composition of suitably configured network functions, network applications, together with the underlying network and data center infrastructure. that are bundled together to meet the requirements of a specific use case, e.g., bandwidth, latency, etc. Slicing is an end-to-end concept covering all network segments including radio networks, wire access, core, transport and edge networks.

Network softwarization aims to reduce operational costs, provide more flexibility, and bring new service paradigms. It will play a key role in providing the mechanisms that allow the dynamic creation of end-to-end slices, and that will be utilised as the main resource management strategy in the future network infrastructures. However, there are still many challenges that need to be addressed to allow the seamless dynamic creation of end-to-end slices on multi-domain infrastructures [5] as envisaged by both 5G and Network 2030.

There is currently a pyramid of software for managing and orchestrating of all the devices and services [6] [7]. Considering the existing layered mechanisms and abstractions available for the management and orchestration of resources (e.g., compute and storage) these systems are quite fragile and are growing in a random and unorganised fashion. As such, we expect new design choices to be carefully made and integrated in the current frameworks that deal with the management and orchestration of end-to-end slices [8].

Even with the drive for softwarization, many network operators are still not able to dynamically allocate the network parts of a slice on-demand. There is still a reliance on paper contracts, people typing in details, running scripts, and sending letters in the post giving the actual end-point information. This raises a big question: *How do you do on-demand, on-the-fly, softwarized and sliced infrastructures, if you have to wait for a piece of paper coming in the post?* The answer is that you need it *all* to be done in software. Any process that requires a human to be involved, and to transfer details by hand to another system is not suitable. For fully softwarized systems, all of the aspects must be done directly with no human intervention. To do these things in software, all the elements need to have soft aspects. There needs to be: (i) the right abstractions to operate over; (ii) the right languages to express operation across them; and (iii) the right run-times to support programmability. These three are large areas of work in themselves.

In response to these three elements, and building on the idea of software-based network management, this paper presents the Wide-area Infrastructure Manager (WIM) on-demand concept. This provides a lightweight, flexibile, and adaptable way to dynamically allocate at run-time, a management and control point for network elements that are part of a tenant requested end-to-end slice. In particular, we explore the use of WIM on-demand for assisting in the management and control of the inter-site network slice connectivity. Similar to cloud tenants being able of instantiate a Virtual Infrastructure Manager (VIM) on-demand for managing and control the cloud infrastructure allocated when providing a Data Center (DC) slice [9], we present the need to control and manage the connectivity paths inter-connecting such compute slices when spread across several locations. The concept of WIM on-demand arises naturally as a complementary element to be offered to the tenants, when required for running their services.

This paper presents the WIM on-demand architectural elements, in section III, designed using the operating system design techniques presented in [10], and a high-level architectural view from [11]. These are required to support such an on-demand model, as well as the set of layered abstractions using slicing elements, showing how they all fit together for service provisioning and integrate with an orchestrator. A design and implementation is also discussed to validate such an approach, in sections IV and V, together with our results, in section VI.

II. Background

Slicing is indicated as a concept that is one of the most important pillars in 5G networks, based on the efforts in standards and academic work. However, there is still no clear unified vision or consensus on it, nor a precise and rigorous definition of what a slice is. This leads to various alternative slicing perspectives. There are multiple contributions related to wireless slicing [12], the scalability and flexibility of slicing [13], as well as slicing surveys [14] and tutorials [15].

The provisioning of services on top of slices, as logically partitioned resources of a multi-domain software-defined infrastructure, can be obtained through different slicing strategies, depending on what layer of the architecture the slicing is introduced and which system elements provide the slicing capabilities. Each approach provides different conceptual ideas of a slice, from a hard partitioning approach up to totally virtualised elements, as well as different management approaches from high-level orchestrators to direct infrastructure management. If slicing is done in the *Orchestration layer* (which is the most commonly used approach) using an inter-domain orchestrator API interaction and / or a peer to peer approach, a slice representation is closer to a set of data structures in the Orchestrator rather than having any actual partitions. Conversely, slicing at *Infrastructure layers* implies some kind of partitioning, and also means that the upper layers, such as Orchestrators and VIMs do not need to know anything about slicing. If a slice part is presented to them, they can carry on working with minimal or no change, because an infrastructure partition looks very much like the whole infrastructure – just smaller. As a consequence, there are trade-offs when selecting one approach or another for slicing. The decision as to which slicing approach to choose will depend on various key aspects of the service requirements under consideration, and can focus on the technical needs of the provider, and the technical abilities and choices of the tenants.

The work presented in this paper has its foundation as slicing performed at the *Infrastructure level*. Our experience [8] is that this approach gives a very modular and flexible way to create slices from different slice parts in different infrastructure domains [16] [17].

There are various projects and initiatives that have explored slicing at the Orchestrator level. These include SONATA [6], 5GEx [7] and 5G-Transformer [18], 5G PAGODA [19], and SLICENET [20]. This approach has the easiest entry position, given that there is already an Orchestrator that does many functions, but it is far more difficult conceptually, as well as to actually implement. For Orchestrator level slicing, the representation of a slice has to be determined across all of the participating management elements, plus all of the main software elements need to be updated and adjusted to have an awareness of slices; and all of the APIs, the modules, and internal function paths, and the data structures need to be adjusted and adapted to factor in slices. We observe that already large pieces of software, get even larger.

There are various organisations that have been addressing slicing and creating standards and definitions, in an on-going basis. The ITU-T Slicing model has been defined in [21], and is used as the basic concept of the Network Softwarization. In this model, Slicing allows logically isolated network partitions, with a slice being a unit of programmable resource such as network, computation, and storage. There is the ETSI NFV Architecture Framework report on "Net Slicing Support" [22], the 3GPP TR23.799 "Study on Architecture for Next Generation System" [23] and 3GPP TR 28.801 "Study on management and orchestration of network slicing for next generation network" [24], which both address Network Slicing, and the ONF Recommendation TR-526 "Applying SDN architecture to Network Slicing". The IETF has "Network Slicing - Revised Problem Statement" [25] and also various documents on Architecture, Management, Models, Use-cases, and Gateway functions. The NGMN Slice capabilities [26] consist of 3 layers: (i) Service Instance Layer, (ii) Network Slice Instance Layer, and (iii) Resource layer. The Service Instance Layer represents the services to be supported, where each one is represented by a Service Instance. Typically, services can be provided by the network operator or by third parties. A Network Slice Instance provides the network with the characteristics required by a Service Instance. There are EU 5GPPP White Papers on 5G Architecture centred on network slicing, from 2016 [27] and 2018 [28]. Additional standards, characteristics, and research activities on slicing can be found in both [15] and [14].

A. *Previous Work*

In our previous work, we made the case for VIM on-demand – a new approach to controlling and managing slice instances, to ensure that their prescribed attributes are propagated end-to-end, including the Data Center resources. This idea was first presented at IETF 100 to the NFVRG [11], and was devised for those situations in which it is important to have enhanced control and interaction over the computational resources of an end-to-end slice. These include telecom scenarios with guarantees, CDN deployments, and services with high levels privacy and isolation requirements.

That work presented a dynamic and modular way to control and interact with cloud resources. A Data Center (DC) Slice Controller allocates a new VIM in an on-demand fashion for each slice segment/part, rather than relying on a single VIM that is reprogrammed to be slice-aware. After a request for a Data Center Slice, the DC Slice Controller checks the requested resource is available, it then reserves the resource, it configures and allocates the new VIM on-the-fly for those resources, and then returns the address of the new VIM to the caller. The newly allocated VIM will control all the virtual resource on the allocated resource. We previously presented and evaluated the design of the DC Slice Controller in [9].

Here, we present a symmetric approach, with the idea of instantiating a management and control point – a WIM – for the network resources of an end-to-end slice. This complements the layered software abstractions available for the creation of end-to-end slices, and makes a step forward toward a unified management of heterogeneous resources required for their automated orchestration. Figure 1 shows how VIM on-demand and WIM on-demand fit into the above devised abstraction layers, and how they can both be allocated dynamically to allow a software-based, finer-grained management and orchestration of the resources of an end-to-end slice.

These ideas are based on the abstractions and concepts outlined in "Lessons from Operating Systems for Layering and Abstractions in 5G Networks" [10], which have further been developed and re-organised to design a layered modular architecture that we believe can guarantee the seamless software-based orchestration of end-to-end slices. The architecture consists of the following functional layers (from the bottom):

Resource: These resources are those that can be part of an end-to-end slice. They mainly include Data Center parts and Network parts held in various domains.

Slice Control: For each resource domain, there is a Slice Control point which is responsible for allocating the individual slice parts and the associated infrastructure manager, so VIM for DC slice and WIM for network slice.

Slice / Partition: These are the individual resource slices (parts) which have been allocated after a request for a slice of infrastructure in that domain.

Slice Management: For each allocated slice part, there is a manager allocated, either VIM on-demand or WIM on-demand, depending on the resource type.

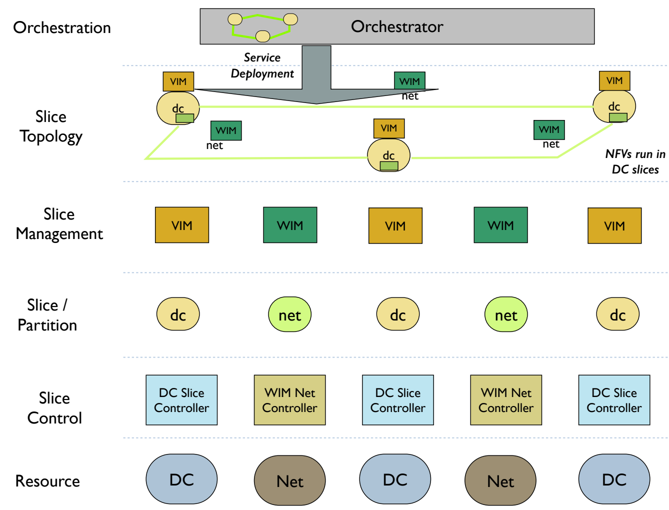

Fig. 1. Layered View

Slice Topology: The full end-to-end topology of a slice can be aggregated from the slice parts, plus the associated infrastructure managers.

Orchestration: The orchestration of services to the full slice is managed by an Orchestrator which will have a view of the whole slice, as well as a direct handle on all the of VIMs and WIMs for the relevant slice parts. In this way, it can directly manage and control the whole slice and the slice parts.

III. WIM ON-DEMAND

A *WIM on-demand* is a dynamically allocated management and control point for softwarized network elements such as those used in the connectivity part of an end-to-end slice. It is considered important to have the option for controlling and managing the connectivity paths inter-connecting DC slice parts in a similar way to the option for tenants to instantiate a VIM on-demand for managing and controlling the DC infrastructure of a DC slice part [9]. The concept of WIM-on-demand arises naturally as a complementary element to be offered to the tenants, and required for running their services.

We observe that the introduction of this new management and control point for the network connectivity of an end-to-end slice, is good for tenants that require a deeper level of control over the network resources of their end-to-end slices. This will grant them the ability to dynamically manage the network elements that have been created for them, and to use those in a programmatic fashion, together with their resource orchestration system, as described in Section I.

The combined usage of VIM on-demand and WIM on-demand on the different slice parts of an end-to-end slice (either DC or Network), is an important step towards the deployment of the layered abstraction mechanisms. These abstractions will allow the uniform orchestration of new network services, which will be deployed by different tenants across multiple resource segments as foreseen by 5G and Network 2030. According to *"A Network Service Provider Perspective on Network Slicing"* [29], different network slices will have different amounts of resource and functionality, and allow different levels of control and management from the tenants side for the slices. As a consequence, there will be different kinds of WIM to match these different aspects. The expected functionality to be offered is explored here, as well as some requirements of the possible scenarios for WIM on-demand.

A. VIM Functionality

First we consider the functionality of a VIM, and we then compare it with that expected from a WIM. The functionality of a VIM is defined in ETSI GS NFV-MAN 001 [30] and the management aspects outlined in ETSI GS NFV-INF 004 [31]. The expected capabilities of a VIM are:

1) Orchestration of the allocation / upgrade / release / reclamation of NFVI resources, and also their optimisation.
2) Managing the association of the virtualised resources to the physical compute, storage, networking resources.
3) Management of VNF Forwarding Graphs (create, query, update, delete), e.g., by creating and maintaining Virtual Links, virtual networks, sub-nets, and ports.
4) Management of security group policies to ensure network/traffic access control.
5) Maintenance of a repository inventory with related information of NFVI hardware resources (compute, storage, networking) and software resources (e.g., hypervisors).
6) Management of the virtualised resource capacity. Usage and performance reporting, density of virtualised resources to physical resources, and fault information.
7) Management of catalogues of virtualised resources.

In this approach to a VIM, there is an underlying assumption that there is one VIM for a whole resource domain, and that VIM will be involved in the overall allocation of resources. As we have presented, this assumption does not hold if we have the VIM on-demand. In our approach, there is a separate Slice Controller that is directly responsible for the allocation of resources and for their elasticity as in item (1), as well as for managing the inventory, as in item (5). Consequently, these functions are not needed by the Infrastructure Manager, when doing on-demand style operation. By analogy, we could expect similar behaviour from a WIM with respect to the connectivity among sites to facilitate the inter-connection of the DC slices via Network slice parts.

B. Functionality expected from a WIM

The WIM will be in charge of the slice connectivity between DCs, while the VIM will manage the slice virtual elements within the DC and their internal connectivity. In order to stitch all of the end-to-end elements, an orchestrator will be able to utilise a software-based approach to program and interact with each VIM and WIM to finalise the instantiation of different end-to-end slices.

The allocation of the network resources for interconnecting DC Slices will be based on the split in functionality among the components shown in Figure 1. More specifically, the expected functionality to be provided by a WAN / Network Slice Controller is for:

1) The allocation / upgrade / release / reclamation of WAN resources (i.e., paths / connectivity) necessary for the inter-connection of compute slice instances from different data centers, including their optimisation.
2) Management of an inventory keeping track of the connectivity resources allocated per tenant, in the form of a catalogue of resources and connectivity.

The concerns of the WIM allocated on-demand for a particular network slice, the expected functionality is:

1) Management of capacity of those interconnection paths and the reporting of their usage, per tenant.
2) Association of the virtualised resources, in this case interconnection paths, to the physical networking resources, which can refer to MPLS LSPs, GRE tunnels, VxLAN connections, optical lambdas, etc.
3) Collection of performance and fault information, facilitating monitoring and alarm management.

When using a WIM, a tenant will be offered a number of methods that allows for a software-based, programmable approach to the execution of network-related management and control operations on a network slice part. These include, e.g., re-routing traffic among slice sites, dynamic establishment of paths, dynamic management of bandwidth, etc. When the network slice is simpler, the WIM will provide more limited methods, such as returning the end-point information of the network slice part back to the tenant. This will allow us to overcome the legacy approach highlighted in the introduction, where that information is supposed to be exchanged via post.

C. Avoiding Conflict in Multiple Allocated Network Slices

There are problems when multiple WIMs have full control on the same network infrastructure. Multiple tenants interacting on the same transport infrastructure resources, all with full control can produce inconsistencies on the configuration and behaviour of the transport network. This can be due to the injection and propagation of contradictory actions from different tenants, triggering reactive measures to actions from other tenants. These situations need to be avoided.

A lack of guarantee on the necessary isolation expected in a slicing environment can have a massive impact. Such isolation can only be totally guaranteed if dedicated resources are allocated to each tenant that needs complete isolation. Because of that, it is necessary to clearly understand the control needs of each tenant, in order to provide a more convenient type of slice and its corresponding level of control for the associated transport network. The two extreme situations could range from a dedicated allocation of resources (network partition) on one hand, to a simple, logical assignment of static inter-site capacity (network overlay) on the other hand.

D. Scenarios of deployment for WIM-on-demand

For better clarity on how the WIM on-demand model works, we consider again the functionality of the VIM, and we observe two main approaches:

VIM-independent slicing: this VIM on-demand model allows tenants to choose the type of VIM, and to have also

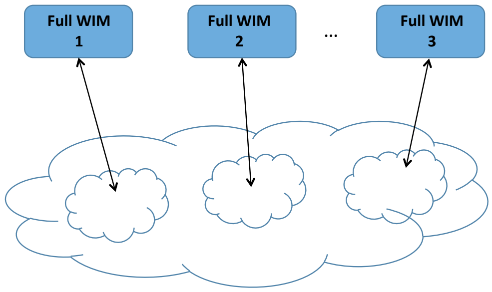

Fig. 2. WIM-independent slicing – Full WIM

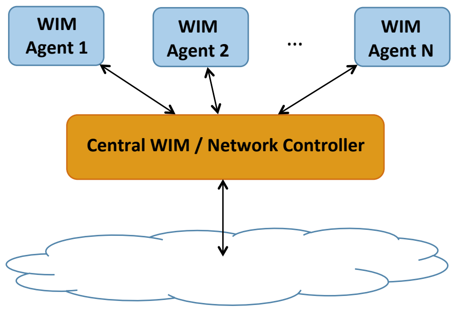

Fig. 3. WIM-dependent slicing – Agent WIM

direct interaction with the remote VIM and control of the allocated resources. In this approach, slicing support at resource providers can be considered lightweight since they do not need to run large slicing-capable VIM instances, and multi-tenancy happens at the resource layers.

VIM-dependent slicing: this is based on multi-tenancy at the VIM level and allows for inter-domain orchestrator to VIM interaction using an allocated VIM agent object. Here, slicing is in the VIM, hence VIM-dependent, but requires a heavyweight VIMs – becoming the lock-in choice of the provider – but it frees the tenant from any VIM responsibility.

There is a symmetric pattern for WIM on-demand. WIM-independent slicing has direct control of the network resource, whereas WIM-dependent slicing has to operate via a main Central WIM / Network Controller. Figure 2 presents the case where the WIM on-demand behaves as a *Full WIM*, with total control on the allocated resources in each domain. In contrast, Figure 3 depicts the scenario where the tenant's WIM behaves as a *WIM Agent*, interacting with a Central WIM / Network Controller which is the actual entity having full control of the underlying transport architecture.

Depending on the kind of slice requested by the tenant, it is possible that all of a WIM's capabilities would not be necessary for all kinds of tenants. Those functions can be related to the type of slice requested (in line with the type of potential slices reported in [29]). Given the type of slice requested and the resource management policies of a network service provider, we expect two main flavours:

- *Full WIM* flavour, where the tenant can have full control of the allocated connectivity resources. This is only feasible if the resources are dedicated to the tenant, in a manner of a *network partition* where the allocated resources are fully under control, in order to avoid any kind of conflict and provide full isolation.
- *WIM Agent* flavour, where the actual control remains on a logically centralized WIM of the network provider, only offering a customized (even limited) view to the tenant's WIM agent. This approach works best when the connectivity resources are not dedicated solely to a specific tenant, but are shared among a number of them. The WIM agent will interact with the central WIM, which actually has the capability of controlling, managing, and operating the connectivity resources.

E. Requirements for both Full WIM and WIM Agent

For WIM on-demand as an operational solution, some requirements have to be taken into account. In the case of the Full WIM, it is necessary to have standardised South-Bound Interfaces (SBI) from the WIM towards the transport network infrastructure, in order to easily integrate the control plane represented by the WIM with the data plane in the transport infrastructure. Examples of that are the device models being defined in IETF [32]. Also, for each transport network provider, it is necessary delegating to the tenant some mechanism or artefact for managing the dedicated infrastructure.

In the case of the WIM Agent, it is necessary to offer standardised North-Bound interfaces (NBI) from the centralised WIM in each transport network domain to the multiple agents running on top of it without ad-hoc integration efforts. As an example of a standard NBI, the ONF Transport API (TAPI) could be considered [33]. Depending on the tenant's needs for network control, isolation mechanisms might also have to be put in place in order to avoid interference between the operations performed by different tenants.

IV. Design for WIM On-demand

To facilitate WIM on-demand, we designed a system that accepts requests for end-to-end slices, allocates a WIM at run-time for the network connectivity, and configures it for the resource requirements, as specified in the request. Using the same working design pattern devised for the DC Slice Controller, the WAN Slice Controller is the component that allocates a WIM on-demand for each network slice. After a tenant's request for a new network slice, the WAN Slice Controller checks the resource availability, reserves the required resources, configures and allocates the WIM for those resources, and finally returns the address of the new WIM to the caller. The WIM plays the important role of returning the relevant attributes of the allocated network resources. This is a vital aspect of a WIM, as it acts as a key component in the software-based management of the network parts of the end-to-end slices.

The WIM is a network slice part control point that supports different operations depending on the underlying network slice that has been activated. There is likely to be a different WIM for different kinds of network slices: some may be very basic, others more complex, depending on what was requested, what

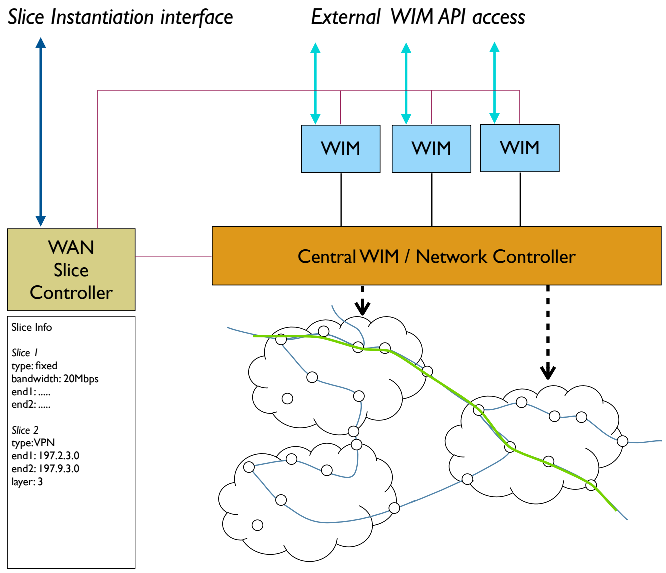

Fig. 4. System view with WAN Slice Controller, WIM and Network Slice

was allocated, and how much control the WIM is allowed. In suitable situations, the network operator might allow some kind of control operations, e.g., for elasticity, to be triggered via the on-demand WIM.

The WAN Slice Controller is the component that resides in each Network Provider and that dynamically creates Network slice parts. In order to create a Network slice, the WAN Slice Controller performs the following operations:

- *Network Management:* the WAN Slice Controller manages all of the transport network resources in the network provider domain that are allocated to participate in slicing and keeps track of which network resources have been allocated to which slice.
- *Slice Creation:* the WAN Slice Controller handles requests for Network slices coming from external sources and determines if it is possible to create a new network slice in the domain, based on resource availability.
- *Resource Provisioning:* if the Network slice creation is possible, the WAN Slice Controller is responsible for providing the set of required transport / virtual links.
- *WIM Deployment:* With dedicated resource, the WAN Slice Controller deploys an on-demand Full WIM to the Network slice, and configures it to use the network resources which have been assigned for the slice. For shared resources, where there is a Central WIM / Network Controller running in that network domain, the WAN Slice controller creates a WIM Agent for the tenant.
- *Network Slice Elasticity:* under the software control of the tenant's Orchestrator, the WAN Slice Controller updates the network resources assigned to the slice on-the-fly as a slice can grow or shrink at runtime if resources are available.
- *Slice Deletion:* the WAN Slice Controller handles requests for the deletion/shutdown of WAN slices. All resources used will be returned to the resource pool, and the WIM allocated for the slice will also be shutdown.

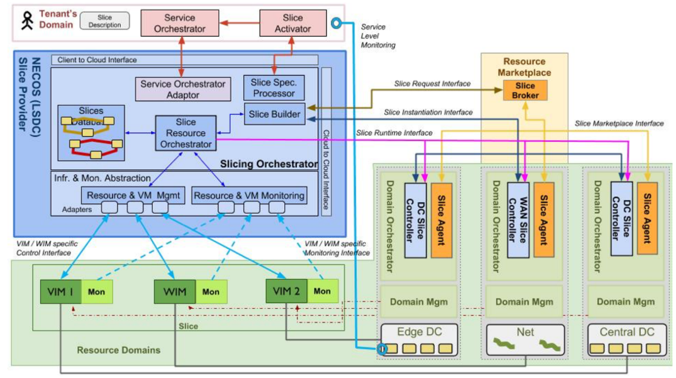

Fig. 5. NECOS Architecture

When we combine the WAN Slice Controller with the WIM on-demand concept, we get a dynamic system that uses a set of layered abstractions to allocate the network connectivity for end-to-end slices, together with the associated management control points, shown in Figure 4.

V. WIM ON-DEMAND IN PRACTICE: THE NECOS MODEL

In this section we discuss how the NECOS project [8] took the architectural ideas and abstractions, including the design of both the WIM on-demand and the VIM on-demand, and built a system focussed on the softwarised, programmable creation of lightweight end-to-end slices. The NECOS architecture, in Figure 5, contains 3 main high-level sub-systems, which are (i) the NECOS Slice Provider (coloured in blue), (ii) the Resource Marketplace (coloured in yellow), and (iii) the Resource Providers (coloured in green). We present the most important aspects concerning the WIM on-demand functionality.

The NECOS Slice Provider is a system that manages the creation of end-to-end Slices from a set of constituent slice parts. In NECOS, a Slice looks the same as the full set of federated resources, with the main attribute being that they look a lot smaller, as they have been sliced. The NECOS Resource Marketplace is the way that the Slice Provider is able to find the constituent slice parts for building up a slice. Rather than having a pre-determined set of providers which have been pre-configured in a federation, NECOS uses a flexible model of a marketplace from which slice parts are provisioned.

Resource Providers are organisations that provide the resources required for the slice parts – namely, Data Center resources with servers and storage, or Network resources. Each resource provider should be capable of offering the slice parts that will compose a full end-to-end slice. Resource Providers also need to provision the relevant Infrastructure Manager on-demand for each slice part; so a VIM for a DC slice part, or a WIM for a network slice part. The managers allow resources to be controlled, managed in software, and utilised by the Slice Resource Orchestrator. These two management components act as a point of interaction for creating and decommissioning both Network and DC slice parts.

For each network domain, a WAN Slice Controller is in charge of instantiating a network slice between two DC slices and either deploying an on-demand full WIM or adding a WIM Agent as an interaction point for the tenant to an existing Central WIM / Network Controller. Similarly, for each data center, a DC Slice Controller is in charge of creating DC slices within the data center, allocating the required compute and storage resources for a given slice part, and returning a handle to a VIM running on it. The VIM can either be a separate instance deployed on-demand, based on the tenant specification, or an existing VIM running in that Resource Domain with a VIM Agent created for the tenant.

There are multiple steps that the Slice Provider uses to find the relevant slice parts from the Resource Marketplace, based on the tenant's specification. Once the slice parts have been selected, the Slice Provider contacts the corresponding WAN Slice Controllers and DC Slice Controllers in each domain. Such communication requires that each remote domain instantiate the slice part by allocating the Resources, and by creating the relevant WIM or VIM. The addresses for the remote VIMs and WIMs are passed back in the request and are stored in the Slice Provider in order to keep track of the deployed slices.

The architectural design concepts of the NECOS system, using both the VIM on-demand and the WIM on-demand approaches, have been evaluated by the implementation of different proof-of-concept prototypes. The WIM on-demand was utilised to exert elasticity operations on the slices as outlined in [34]. In this work, the authors utilise the NECOS components to test their SLOTS algorithm for slice elasticity. The DC Slice Controller and the Network Slice Controller are used to dynamically create multiple slices with different uses, and then to evaluate the resource elasticity control functions tailored to cloud-network slicing-defined systems.

VI. WIM on-demand: System Experience

Another prototype was developed to validate the NECOS multi-domain functionality using the VIM on-demand and WIM on-demand systems. It implements the main NECOS functionalities of the Tenant domain, of the Slice Provider, and of the Resource Provider sub-systems shown in Figure 5. The system demonstrates the ability to create lightweight end-to-end slices via a programmable software-based approach that relies on unified resource management abstractions, as discussed earlier. We show how this is important for the automated allocation of end-to-end slices and for their delivery to the tenants via the Slice as a Service model. Experiments were carried out to validate the feasibility, the flexibility, the adaptability, and the modularity of the proposed slicing ideas on a scenario that included multiple, inter-connected resource-constrained domains.

As this implementation is focussed on lightweight slices, it fits perfectly with those scenarios that include resource domains located at the edge, where the resource availability is likely to be limited. The lightweight characteristic of the slices supported by this system is due to the usage of both the VLSP VIM [35] and a newly devised lightweight WIM

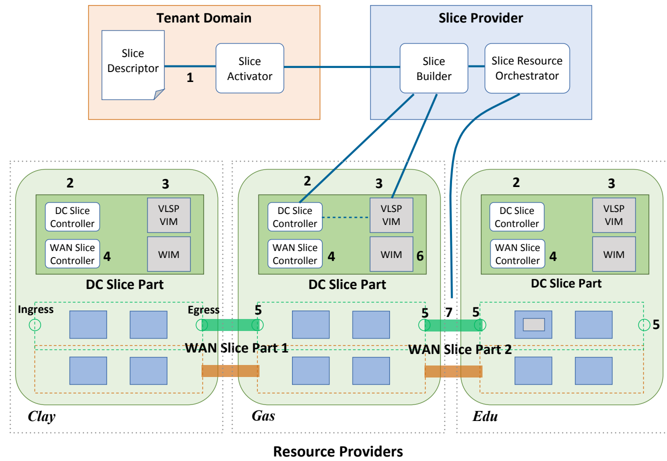

Fig. 6. NECOS Multi-domain Evaluation

prototype that is instantiated on-demand during the setup of the different Network Slice Parts. In this system, each of the NECOS sub-systems was implemented by a separate set of software components, i.e., *Tenant Domain* (top-left), *Slice Provider* (top-right) and *Resource Providers* (three instances at the bottom), as shown in Figure 6.

A. Testbed Description

To facilitate the software deployment and experimentation, these resource domains were created by partitioning a Data Centre located at University College London (UCL) into three different groups (as mini DCs). In Figure 6, each of the mini DCs (called *Clay*, *Gas*, and *Edu*) represents a separate Resource Provider in the testing environment. In order to allocate new network slice parts in the testbed, the existing LAN inter-connection between the mini DCs was partitioned by creating separate overlay networks (i.e., tunnels), each associated to a different network slice part.

B. Software components and their interaction

The implementation details of the system are presented in Figure 6, highlighting the components, the workflow steps, the way they inter-work when a request for a new end-to-end slice is submitted by a tenant to a Slice Provider, and how the required resources are allocated in the Resource Providers.

A tenant that creates a new Slice, will provide the required Slice Descriptor, via a request to the *Slice Activator* module inside the Tenant domain, in step 1, (Fig. 6). In this system, the descriptor is a YAML file that contains the number of DC slice parts, the related type of VIMs, and the number of resources, as well as the way the DC Slice Parts will have to be inter-connected with network slice parts, in order to build the final end-to-end slice. The Slice Activator provides a REST interface that can be utilised by an administrator (or external software system) to upload and submit the desired slice descriptor to the system. The Slice Activator acts like an agent of the tenant, by taking care of passing the Slice requests on to the Slice Provider system, and keeping track of the status of all the requested slices for the tenant.

Once a descriptor is received and validated by the Slice Activator, the Slice request is passed to the *Slice Builder* module of the Slice Provider, who will take care of parsing the slice topology specified in the Slice Descriptor. The Slice Builder is responsible for fetching the required Slice Parts from the various Resource Providers. This system evaluation does not utilise the functionalities of the Resource Marketplace as its main focus is on end-to-end slice instantiation, hence the Slice Builder will be informed (via the Slice Descriptor) about the entry-points of the Resource Providers to be contacted.

The Slice Builder uses the nominated entry-points to interact with the DC Slice Controllers of the specified Resource Providers, represented in step 2 (Fig. 6), in order to submit the requests for the DC Slice Parts that needs to be allocated. Each requested DC Slice is a separate partition of the available physical computational resources that will be managed by an on-demand instance of the VLSP VIM created for each DC Slice. Once the DC Slice is allocated, a handle to each of the instantiated VIM instances will be passed back to the Slice Builder, as represented in step 3 (Fig. 6). (For simplification, the arrows related to these steps have been drawn only for the *Gas* domain, but apply to all of the domains).

A symmetric approach is performed for the network resources, in step 4 (Fig. 6). The Slice Builder submits a separate request for each of the WAN Slice Parts that were specified in the Slice Descriptor. Each request is forwarded to the WAN Slice Controller of the resource domains that host the DC Slice Parts to be inter-connected. The allocated DC Slice Parts are inter-connected to each other via various Network Slice Parts. After the successful allocation of the connectivity resources, an instance of a WIM will be spawned on-demand and a handle to it returned back to the Slice Builder.

In order to connect the *DC Slice Part Gas* to the *DC Slice Part Edu*, the following operations are performed. The Slice Builder sends a request to the *Edu* WAN Slice Controller for the allocation of a network slice with ingress and egress points to be used on that domain for that particular slice, step 5, (Fig. 6). The network resources are configured inside the domain to attach the previously allocated DC Slice Part to the above ingress and egress points. In step 6, (Fig. 6), an instance of a WIM is created on-demand to allow the management of the allocated network resources (in particular, retrieving the ingress / egress address information, and for configuring the connectivity). Using the lightweight components in this testbed, the process of sending the request to the WAN Slice Controller to having a handle returned to an executing WIM, takes between 4 seconds and 6 seconds, depending on the capacity and load of the server.

A similar process is performed on the *Gas* domain to allocate the required network slice ingress / egress points, configure the internal network connectivity, and then start an on-demand instance of a WIM. In order to finalise the creation of the Network Slice Part between *Gas* and *Edu*, some network capacity (in the form of overlay network) needs to be allocated between the egress point of *Gas* and the ingress point of *Edu*: this is triggered in step 7, (Fig. 6), by the Slice Resource Orchestrator using the end-point information collected at run-time from both the WIMs, and by interacting with the WAN Slice Controllers in the *Gas* and *Edu* domains. Traffic can now flow across the different resource domains, indicating the correct setup of the required connectivity in the slice.

Using this system we were able to demonstrate the interworking of both the DC Slice Controllers and the WAN Slice Controllers to perform the automated, software-based creation of an end-to-end slice across 3 domains. The resources were allocated dynamically using a Slice Descriptor and by interacting with on-demand created instances of VIMs and WIMs, which represented the management and control points for each of the slice parts. We were able to get both WIM on-demand and VIM on-demand to work successfully, and demonstrate a modular architecture and created slices that are not only isolated from each other, but also have independent management. This is the most important result achieved by this prototype system, as it demonstrated how the layered abstractions devised for the management of the DC and Network resources allowed an Orchestrator to perform the seamless creation of end-to-end slices via a unified programmable, software-based approach.

VII. Conclusions

Both the WIM on-demand and the VIM on-demand concept are an important contribution as there are some situations in which it is extremely important to have separate Network slices and Data Center slices within a full end-to-end Slice, such as in services with high levels of isolation and privacy requirements, telecom scenarios with delivery guarantees, energy optimisation, and CDN deployments. Through the use of a set of Network and DC Slice Controllers, a new slice can be requested. These slices will be allocated and de-allocated in an on-demand fashion, according to the tenants requests. Each slice comes with its own dedicated WIM, allowing the tenant to have a new level of softwarised control and flexibility.

In order to support service provisioning over a dynamically sliceable infrastructure, it is necessary to have run-time mechanisms to support the slicing of both the Network and Data Center resources. These strategies are part of a bigger picture for control and orchestration in slices, and this approach is highly suitable for use in distributed clouds, seen in the context of 5G networks. The design of our slicing system results in slices being fully isolated from each other. The slice elements are only used by the tenant of the slice, so ensuring isolation.

Network slicing with WIM on-demand allows network operators new technical and economic opportunities, including:

- the ability to deploy different WIMs for different slices, giving flexibility to the tenant, and meaning the provider has more than software solution and charging options,
- the ability to request and instantiate network dynamically at run-time via a software-based approach, as the tenant can use the WIM to get all the necessary information and adjust the slice on-the-fly.

These on-demand functions have not been considered by other slicing approaches [36]. However, the NECOS project has successfully utilised the functionality of WIM on-demand and VIM on-demand to dynamically create end-to-end slices and bring about the Slice as a Service concept, and we have shown them working successfully in Section VI.

As future work, we plan to expand the conceptual ideas presented in this paper. We are currently working on a full architecture for an enhanced approach to the management of end-to-end slices following the layered design strategies already highlighted in this paper. We have addressed the slicing in networks, but we see that there is much missing in this area. Although the network can support the infrastructure slicing operations as a native function, the management parts for dynamic, software controlled run-time operation are missing. A partner operator and network provider is currently investigating how the techniques devised and presented here can be used in their network, and then we plan to investigate how this can be rolled out in other provider networks.

The need for dynamic end-to-end slices, composed of slice parts, all available at run-time under software control, used for service deployment, allows for a layered, composable abstraction for slices that is more controllable and flexible for the tenant and also more dynamic and flexible for the infrastructure provider. It makes it easier for providers to allocate slices to the tenant, while still giving out some level of run-time adaptability and control.

ACKNOWLEDGEMENTS

This work was supported by the EU / Brazil project: NECOS – "Novel Enablers for Cloud Slicing" (777067).

REFERENCES

[1] 5GPPP Architecture Working Group, "View on 5G Architecture," December 2017. [Online]. Available: https://5g-ppp.eu/wp-content/uploads/2018/01/5G-PPP-5G-Architecture-White-Paper-Jan-2018-v2.0.pdf

[2] F. Tusa, S. Clayman, D. Valocchi, and A. Galis, "Multi-Domain Orchestration for the Deployment and Management of Services on a Slice Enabled NFVI," in *IEEE Mobility Support in Slice-based Network Control for Heterogeneous Environments co-hosted with Conference on Network Function Virtualization and Software Defined Networks*, Verona, November 2018.

[3] ITU-T FG-NET-2030, "Network 2030 - A Blueprint of Technology, Applications and Market Drivers Towards the Year 2030 and Beyond," May 2019.

[4] E. Kaljic, A. Maric, P. Begovic, and M. Hadzialic, "A Survey on Data Plane Flexibility and Programmability in Software-Defined Networking," *IEEE Access*, vol. 7, pp. 47 804–47 840, 04 2019.

[5] F. Tusa, S. Clayman, D. Valocchi, and A. Galis, "Hierarchical Service Providers on Sliced Infrastructure," *Internet Technology Letters*, vol. 2, no. 4, p. e105, 2019.

[6] SONATA, "EU H2020 - 5G Service Programing and Orchestration for Virtualized Software Networks," 2015, https://5g-ppp.eu/sonata/.

[7] 5GEx, "EU H2020 - 5G Multi-Domain Exchange (5GEx) project," 2015, https://5g-ppp.eu/5GEx.

[8] NECOS, "EU-Brazil - Novel Enablers for Cloud Slicing," 2017, http://www.h2020-necos.eu.

[9] S. Clayman, F. Tusa, and A. Galis, "Extending Slices into Data Centers: the VIM on-demand model," ser. IEEE 9th International Conference on Network of the Future - NoF, Poznań, Poland., 19-21 November 2018.

[10] S. Clayman and D. Tuncer, "Lessons from operating systems for layering and abstractions in 5g networks," in *NOMS - 2018 IEEE/IFIP Network Operations and Management Symposium*. Taipei, Taiwan: IEEE, 2018.

[11] S. Clayman, "Network Slicing Supported by Dynamic VIM Instantatiation," in *IETF 100, Singapore*, 2017. [Online]. Available: https://datatracker.ietf.org/meeting/100/materials/slides-100-nfvrg-3-network-slicing-support-by-dynamic-vim-instantiation/

[12] S. Zhang, "An overview of network slicing for 5g," *IEEE Wireless Communications*, vol. 26, no. 3, pp. 111–117, June 2019.

[13] P. Rost, C. Mannweiler, D. S. Michalopoulos, C. Sartori, V. Sciancalepore, N. Sastry, O. Holland, S. Tayade, B. Han, D. Bega, D. Aziz, and H. Bakker, "Network slicing to enable scalability and flexibility in 5g mobile networks," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 72–79, May 2017.

[14] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Communications Surveys Tutorials*, vol. 20, no. 3, pp. 2429–2453, thirdquarter 2018.

[15] A. Galis and K. Makhijani, "Infrastructure Slicing Landscape – Tutorial," in *IEEE NetSoft 2018*, Montreal, Canada, 2018. [Online]. Available: http://discovery.ucl.ac.uk/10051374/

[16] L. A. Freitas, V. G. Braga, S. L. Correa, 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 *Workshop on Advances in Slicing for Softwarized Infrastructures (S4SI)*, Montreal, Canada, 2018.

[17] P. D. Maciel, F. L. Verdi, P. Valsamas *et al.*, "A Marketplace-based Approach to Cloud Network Slice Composition Across Multiple Domains," in *2nd Workshop on Advances in Slicing for Softwarized Infrastructures (S4SI), co-hosted at the 5th IEEE NetSoft*, Paris, June 2019.

[18] 5G Transformer, "5G Mobile Transport Platform for Verticals," 2017, http://5g-transformer.eu.

[19] 5G Pagoda, "EU-Japan - A network slice for every service," 2017, https://5g-pagoda.aalto.fi.

[20] Slicenet, "End-to-End Cognitive Network Slicing and Slice Management Framework in Virtualised Multi-Domain, Multi-Tenant 5G Networks," 2017, https://slicenet.eu.

[21] ITU, "ITU-T Y.3011," 2011. [Online]. Available: http://www.itu.int/rec/T-REC-Y.3001-201105-I

[22] ETSI, "Etsi report on net slicing support with etsi nfv architecture framework." [Online]. Available: http://www.etsi.org/deliver/etsi_gr/NFV-EVE/001_099/012/03.01.01_60/gr_NFV-EVE012v030101p.pdf

[23] 3GPP, "Study on architecture for next generation system," Tech. Rep. TR23.799, 2016.

[24] ——, "Study on management and orchestration of network slicing for next generation network (release 15)," Tech. Rep. TR 28.801.

[25] IETF, "Network slicing - revised problem statement, draft-galis-netslices-revised- problem-statement-03," 2018.

[26] NGMN Alliance, "Description of Network Slicing Concept", NGMN 5G P1 Requirements & Architecture, Work Stream End-to-End Architecture," 2016.

[27] EU 5GPPP, "White Paper on 5G Architecture," 2016. [Online]. Available: https://5g-ppp.eu/wp-content/uploads/2014/02/5G-PPP-5G-Architecture-WP-July-2016.pdf

[28] ——, "White paper on 5g architecture," 2018. [Online]. Available: https://5g-ppp.eu/wp-content/uploads/2018/01/5G-PPP-5G-Architecture-White-Paper-Jan-2018-v2.0.pdf

[29] L. Contreras and D. López, "A Network Service Provider Perspective on Network Slicing," *IEEE Softwarization*, January 2018.

[30] ETSI, "ETSI GS NFV-MAN 001, Network Functions Virtualisation (NFV), Management and Orchestration, V1.1.1," December 2014.

[31] ——, "ETSI GS NFV-IFA 004, Network Functions Virtualisation (NFV) Release 2; Acceleration Technologies, Management Aspects Specification, V2.3.1," August 2018.

[32] L. Berger, C. Hopps, A. Lindem *et al.*, "YANG Model for Logical Network Elements," Tech. Rep. RFC 8530, March 2019.

[33] ONF, "ONF Transport API (TAPI) 2.0," August 2017.

[34] A. Medeiros, A. Neto, S. Sampaio, R. Pasquini, and J. Baliosian, "End-to-end elasticity control of cloud-network slices," *Internet Technology Letters*, May 2019. [Online]. Available: https://doi.org/10.1002/itl2.106

[35] S. Clayman, L. Mamatas, and A. Galis, "Experimenting with Control Operations in Software-Defined Infrastructures," in *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, June 2016, pp. 390–396.

[36] A. Galis, F. Tusa, S. Clayman, C. Rosenberg, and J. Serrat, "Slicing 5G Networks: An Architectural Survey," in *The Essential 5G Reference Online*. USA: Wiley, 2019.