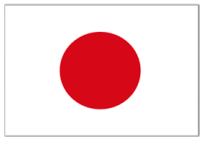

D4.1 – Scalability-driven management system

Aalto, Eurecom, Ericsson, Orange, UT, WU, KDDI, NESIC, HITACHI

Document Number	D4.1
Status	Issue 1.0
Work Package	WP 4
Deliverable Type	Report
Date of Delivery	19/February/2018 (M18)
Responsible	HITACHI
Contributors	Aalto, Eurecom, Ericsson, Orange, UT, WU, KDDI, NESIC, HITACHI
Dissemination level	PU

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 723172, from the Swiss State Secretariat for Education, Research and Innovation, and from the Japanese Ministry of Internal Affairs and Communications.

Change History

Version	Date	Status	Author (Company)	Description
1.0	19/Feb/2018		See AUTHORS	First approved Version

AUTHORS

Full Name	Affiliation
Sławomir Kukliński	Orange
Tomasz Osiński	Orange
Lechosław Tomaszewski	Orange
Taleb Tarik	Aalto University
Miloud Baga	Aalto University
Ibrahim Afolabi	Aalto University
Ilias Benkacem	Aalto University
Nicklas Beijar	Ericsson
Adlen Ksentini	Eurecom Institute
Pantelis Frangoudis	Eurecom Institute
Daisuke Okabe	Hitachi
Kota Kawahara	Hitachi
Hidenori Inouchi	Hitachi
Itsuro Morita	KDDI Research
Yoshinori Kitatsuji	KDDI Research
Zaw Htike	KDDI Research
Phyo May Thet	KDDI Research
Cédric Crettaz	Mandat International
Kazuto Satou	NESIC
Masato Yamazaki	NESIC
Hiroshi Takezawa	NESIC
Akihiro Nakao	The University of Tokyo
Shu Yamamoto	The University of Tokyo
Du Ping	The University of Tokyo
Yoshiaki Kiriha	The University of Tokyo
Toshitaka Tsuda	Waseda University
Takuro Sato	Waseda University

Executive Summary

In this deliverable, we have provided the basis for the work of all WPs of the 5G!Pagoda project, which aims to align European and Japanese views on 5G slice-based mobile social infrastructure; especially focusing on the features for dynamic creation and scalable management of end-to-end slices as a means to deploy private, customized mobile social infrastructure for different services.

Our contributions in this deliverable are three-fold:

First, we introduced the architecture of scalable orchestration including slice federation and migration functions in a single administrative domain, and the design of scalable management of Network Functions Virtualization (NFV) domain slicing systems.

Second, although eMBB, URLLC, and MTC are publicly known as the three major requirements for the general 5G system, so far there are very few discussions about concrete Key Goal Indicators (KGIs) to fulfill each use-case requirement. We identified six keys and values to characterize eight different use-cases. These keys are (1) the number of slices, (2) location, (3) frequency, (4) lifecycle time, (5) heterogeneity, and (6) hierarchy level. Then we decided to choose three Key Performance Indicators (KPIs) to evaluate the scalability of a given network slicing infrastructure implementation, namely, (1) the number of instances (of slices) (2) status change frequency of instances (of slices) and (3) latency in computation and data processing. To the best of our knowledge, this is the first attempt among all other network slicing R & D projects.

Finally, we introduced a new infrastructure design concept called 'Resource Pool', which is the key to achieve the three KPIs described above. In order to provide and develop new and flexible services with tenants, it is necessary to analyze the lifecycle of network resources and to provide a mechanism for dy namically allocating available network resources to the tenants. Recently, managing the lifecycle of slices has caught much attention as well as constructing slices, especially the agility and flexibility in resource allocation becomes a necessity. We propose two novel technologies. First, we propose multi-layer slicing structure where upper layer slices enable added (temporal and functional) values on top of lower slices. For example, the top layer slice represents service level and the bottom layer is virtualization of physical infrastructure resources. Second, as an application of the proposed multi-layer slice model, we present a middle layer between the top and the bottom layer slices called 'Resource Pool' that implements life-cycle management of resources and especially adds temporal value, that is, acceleration of resource allocation.

In other words, 'Resource Pool' is a collection of *digitalized* virtual assets, which can be converted from all the physical assets. Network slice is one of the virtual assets. Using 'Resource Pool' as a middle-layer function of Resource Orchestrator (RO), we may quickly cope with changes and/or failures, e.g. resource demand up and down in the upper service layer, a sudden physical failure in the lower network layer and so on. Since 'Resource Pool' is intended to provide necessary and sufficient resources for the corresponding upper and lower tasks on the basis of dynamic and smart forecasting. As a result, 'Resource Pool' based orchestrator can make the most use of limited resources and thus provide a highly agile and resilient 5G service platform for service and network operators and their end customers.

As a conclusion, with our three novel contributions described above, the deliverable will guide the 5G!Pagoda project as follows:

- The design of the fundamental orchestration, federation, and migration architecture in each technology domain described in the deliverable enables the rapid development and deployment of different services on the 5G mobile network infrastructure.

- The design of scalable NFV domain management architecture described in the deliverable enables the synchronization of the further developments among the 5G mobile network infrastructures.

Table of Contents

List of Tables

List of Figures

1. Introduction

1.1. Motivation and Scope

One of the challenges associated with Network Slicing is the orchestration and management of a high number of deployed network slices. As described in D2.1, several 5G-use cases need the deployment of a massive number of network slices, requiring scalable orchestration and management mechanisms.

This document details 5G!Pagoda contributions to support a massive number of network slices, via scalable orchestration and management functions. Relying on the 5G!Pagoda reference architecture, defined in the D2.3, and integrating Edge Platform, different functional blocks (OSS/BSS, NFVO, and VNFM) will be detailed aiming at managing a massive number of networks slices. Mainly, by describing basic policy and the function to realize “Scalable Orchestration and Scalable Management” of 5G!Pagoda. Furthermore, the document presents some example of scalable appliances and VNF; focusing on CDNaaS and mobile network VNFs.

1.2. Structure of the document

Following this introductory section, the remaining part of the document is structured as follows:

- Chapter 2 provides key terminologies to be used in the document. It includes the descriptions of abbreviations and technical terms.
- Chapter 3 describes the D2.1 use-case requirements from the viewpoint of scalability management.
- Chapter 4 describes the overview of the scalable management and orchestration architecture.
- Chapter 5 describes the concept of the scalable orchestration, orchestration model and sequence from the implementation point of view and the novel technology called “Resource Pool” and “Slice Stitching”.
- Chapter 6 describes the technology domain-specific considerations for scalable management architecture and implementation.
- Chapter 7 describes the activities and surveys by the other standard parties for the slice management and orchestration.

2. Terminology

2.1. Abbreviations

Throughout this document, the following acronyms, listed in Table 1, are used.

Table 1 – List of Acronyms

Abbreviations	Original terms
5G PPP	5G Infrastructure Public Private Partnership
AMF	Access and Management Function
API	Application Programming Interface
BSSO	Business Service Slice Orchestrator
CDN	Contents Delivery Network
CSAR	Cloud Service Archive
DSSO	Domain Specific Slice Orchestrator
E2E	End to End
EPC	Evolved Packet Core
EM	Element Management
eMBB	enhanced Mobile Broadband
EMS	Element Management System
ETL	Extraction Transformation Loading
ETSI	European Telecommunications Standards Institute
FCAPS	Fault, Configuration, Accounting, Performance, and Security
ICN	Information Centric Networking
IoT	Internet of Things
JSON	JavaScript Object Notation
MANO	Management and Network Orchestration (ETSI NFV)
MEC	Mobile Edge Computing
MDSO	Multi-Domain Slice Orchestrator
MME	Mobility Management Entity
mMTC	massive Machine Type Communications
NFV	Network Function Virtualization
NFVI	NFV Infrastructure (ETSI NFV)
NFVO	NFV Orchestrator (ETSI NFV)
NSD	Network Service Descriptor (ETSI NFV)
NSO	Network Service Orchestrator
PNF	Physical Network Function
OAI	Open Air Interface

QoE	Quality of Experience
REST	REpresentational State Transfer
RO	Resource Orchestrator
RU	Radio Unit
SDN	Software Defined Network
SLA	Service Level Agreement
TOSCA	Topology and Orchestration Specification for Cloud Applications
UE	User Equipment
URLLC	Ultra-Reliable and Low Latency Communications
VIM	Virtual Infrastructure Manager (ETSI NFV)
VNF	Virtual Network Function (ETSI NFV)
VNFM	VNF Manager
WAN	Wide Area Network
WIM	Wide Area Network Infrastructure Manager

2.2. Terminology

Table 2 provides a list of terminology used in this document along with its definition.

Table 2 – Terms defined in this document.

Terminology	Definition
The Fifth Generation of Mobile Communications System (5G system)	The proposed next major phase of mobile telecommunications standards beyond the current 4G/IMT-Advanced standards. Rather than faster peak Internet connection speeds, 5G system planning aims at a higher capacity than the current fourth generation of the mobile communication system, allowing a higher number of mobile broadband users per area unit and allowing consumption of higher or unlimited data quantities in gigabyte per month and user. Note that 5G system does not strictly represent the advancements in the radio area, mainly concentrating on the connectivity and data services.
Slice	An isolated collection of programmable resources to implement network functions and application services through software programs to accommodate individual network functions application services within each slice without interfering with the other functions and services on the other slices.
Network Function Virtualization	A network architecture concept that uses the technologies of IT virtualization to virtualize entire classes of network node functions into building blocks that may connect, or chain together, to create communication services.
Virtualized Network Function	Software implementations of network function that can be deployed on a Network Function Virtualization Infrastructure.
IMT-2020	A provisional name of the equivalent standard on 5G system defined in ITU-R WP 5D.
Working party 5D – IMT systems	A standard community responsible for the overall radio system aspects of International Mobile Telecommunications (IMT) systems, comprising the IMT-2000, IMT-Advanced and IMT for 2020 and beyond.
5G!Pagoda	A research project federating Japanese and European 5G system test-beds to explore relevant standards and align views on 5G system mobile network infrastructure supporting dynamic creation and management of network slices for different mobile services
The Third Generation Partnership Project (3GPP)	Collaboration between groups of telecommunications associations, known as the Organizational Partners. The initial scope of 3GPP was to make a globally applicable 3rd generation (3G) mobile phone system specification based on evolved Global System for Mobile Communications (GSM) specifications within the scope of the International Mobile Telecommunications-2000 project of the International Telecommunication Union (ITU). The scope was later enlarged to include the development and maintenance of GSM and

	related ‘2G’ and ‘2.5G’ standards including GPRS and EDGE, UMTS and related ‘3G’ standards including HSPA, LTE and related ‘4G’ standards, an evolved IP Multimedia Subsystem (IMS) developed in an access independent manner and next generation and related ‘the fifth generation’ standards.
Internet of Things (IoT)	The internetworking of physical devices, vehicles (also referred to as ‘connected devices’ and ‘smart devices’), buildings and other items – embedded with electronics, software, sensors, actuators and network connectivity that enable these objects to collect and exchange data. In 2013 the Global Standards Initiative on the Internet of Things (IoT-GSI) defined the IoT as ‘the infrastructure of the information society’.
Content delivery network	A globally distributed network of proxy servers deployed in multiple data centers. The goal of a CDN is to serve content to end-users with high availability and high performance. CDNs serve a large fraction of the Internet content today, including web objects (text, graphics, and scripts), downloadable objects (media files, software, and documents), applications (e-commerce, portals), live streaming media, on-demand streaming media and social networks.
Quality of Experience	A measure of a customer's experiences with a service (web browsing, phone call, TV broadcast, call to a Call Centre). QoE focuses on the entire service experience and is a more holistic evaluation than the more narrowly focused user experience (focused on a software interface) and customer-support experience (support focused).

3. Scalability Requirement

3.1. Use-cases and requirements

Deliverable 2.1 “Use case scenarios and technical system requirements definition” (D2.1) provides eight use cases and three reference use cases. D2.1 has already presented the technical requirements. In this section, we discuss in which scenario and when scaling is required and quantify the scalability requirements for each use case.

3.1.1. Smart Drive-Assistant Services

The major goals of smart drive-assistant service system are to provide safety and stress-free driving experience. For safety purpose, various control information needs to be exchanged between neighboring vehicles and roadside units (RSU). This control information needs to be exchanged with extra low latency. The mobile edge server may be required for analyzing the data and real-time decision making. On the other hand, the driver and passengers on board should be able to access multimedia data for in-vehicle infotainment.

Therefore, regardless of the capability of network slice and user density, at least two types of network slices are required: one is low latency network slice for control message dissemination and/or exchange, the other one is high bandwidth network slice for multimedia and infotainment. In terms of user density, not only the number of automobiles and number of passengers on board but also the number of sensors embedded in a car should be taken into account. Currently, each vehicle has an average of 60-100 sensors on board. As cars are rapidly getting “smarter” the total number of sensors is projected to reach as many as 200 sensors per car.

When the traffic jam or some accidents happen, hundreds of vehicles will be stuck in an area for certain period. In such scenarios, not only the total number of users (including sensors) but also the amount of data generated by sensors and the total traffic generated by passengers will be increased dramatically. The system may need to be scaled out by creating new network slices or scaled up by increasing more resources (compute, network, etc.,) to existing network slice. Nonetheless, in such scenario, the scaling out or up should be done in a few seconds.

The key performance indicators to scale the Smart Drive-Assistant Services are shown in Table 3.

Table 3 – Requirement of the Smart Drive-Assistant Services

#	Requirement Parameter	Range	Description
1	Number of slices	3-100	A number of slice according to the service providers, such as the number of automobile companies.
2	Location	A Nationwide	The smallest size of slices is the size of the country. Network service provider may provide their services covering multiple countries.
3	Frequency	A day	The modification of slice at the highest frequency.
4	Lifecycle Time	Years	The creation and deletion of slices are extremely low frequent, such as many years, on average.

5	Heterogeneous	1	Every service provider How much different kind of slices?
6	Hierarchy Level	1	Service providers only request the resources for management and controlling their system

3.1.2. Industrial Factory Management

Important challenges for smart industrial factory management would be to ensure the continuity of productions and to maintain “globally-equivalent high qualities of the manufactured products”. Globally centralized controlling of the product manufacturing lines all over the world is one of the cost-efficient approaches to achieve this. In this case, 5G systems will provide wireless communication facilities for collecting the various operational status and production information from sensor devices and surveillance cameras and control manipulators.

When the demands for the product increase dramatically or new products are about to be introduced, a new product-line(s) would be implemented. For the former case, the current network slice providing 5G communication facilities may need to be scaled-up and, for the latter one, a new network slice may need to be created, i.e., scale out. In this scenario, the network operator (who is providing communication services) or the orchestrator can get the information, such as the type of products, and requirements, such as latency, bandwidth, in advance and can perform scaling out/up accordingly. In such case, scaling can be done in a few hours or a day, yet it needs to ensure the continuity of current productions and its requirements.

The key performance indicators to scale the Industrial Factory Management are shown in Table 4.

Table 4 – Requirement of the Industrial Factory Management

#	Requirement Parameter	Range	Description
1	Number of slices	1,000-10,000	A number of slice according to the service providers, such as the number of Industry factories
2	Location	Multiple countries	The smallest size of slices is the size of a city. The network operator may provide their services covering multiple countries.
3	Frequency	Several minutes	The modification of slice at the highest frequency.
4	Lifecycle Time	Several hours	The creation and deletion of slices are extremely low frequent, such as several years, on average.
5	Heterogeneity	From 10 to 1,000	Depending on the number of manufactured products.
6	Hierarchy Level	1	Service providers only request the resources for managing and controlling their system

3.1.3. Ensuring QoS on Demand

The key parameters for this use case are bandwidth and user density. As stated in D 2.1, in case of a large-scale sports event, such as the 2020 Tokyo Olympics, more than eighty thousand people are expected to attend the Olympic opening ceremony. Some audiences may share HD live videos (i.e., Facebook live) with friends and relatives that are far away. Some may take HD/360 photos and post them on Facebook or

Instagram. Such applications require high data rate. In this circumstance, the instant surging of bandwidth requirements is very likely to occur. The events normally last just a few hours and the operators do not need to provide ultra-high network capacity for 24/7. The required network capacity needs to be provided on demand. Operators can get the events information in advance and scale out/up in order to meet the requirements during the events and, they may scale down the network capacity after the events. Therefore, scaling, not only scaling out/up but also scaling down, occurs more frequently in this use case, and scaling should be done in a few seconds. Additionally, constant monitoring would be required for optimal scaling.

The key performance indicators to scale the Ensuring QoS on Demand are shown in Table 5.

Table 5 – Requirement of the Ensuring QoS on Demand

#	Requirement Parameter	Range	Description
1	Number of slices	100-1,000	A number of slice according to the service providers.
2	Location	Event sites	The smallest size of slices is the size of stadiums.
3	Frequency	Several minutes	The modification of slice is in the highest frequency.
4	Lifecycle Time	Several hours	The creation and deletion of slices are extremely high frequent, such as every multiple year, on average.
5	Heterogeneity	1 or 2	Depending on the QoS types provided by the service providers.
6	Hierarchy Level	1	Service providers only request the resources for management and controlling their system

3.1.4. Smart/Virtual Office

In the smart/virtual office scenarios, dramatic changes of network capacity or users' density are unlikely to occur. However, when interactive business meetings need to be held, the orchestrator may need to create/initiate a new network slice with specific requirements. The orchestrator may create one dedicated network slice with specific requirements for one meeting for security or reliability reasons. In such case, the orchestrator needs to create (or scale out) multiple network slices sequentially or simultaneously. Either case, the network slices should be created or, in other words, scaling out should be done in a few seconds. The communication environment for a meeting spread multiple sites, and they belong to an organization, such as a company.

The key performance indicators to scale the Smart/Virtual Office are shown in Table 6.

Table 6 – Requirement of the Smart/Virtual Office

#	Requirement Parameter	Range	Description
1	Number of slices	100,000-1,000,000	A number of slice according to the number of companies.
2	Location	Meeting place in an office and/or office in a tenant building	The smallest size of slices is a company.
3	Frequency	A few seconds	The modification of slice is triggered by creation a satellite meeting and office environment.
4	Lifecycle Time	30 minutes	The creation and deletion of slices are extremely frequent.
5	Heterogeneity	2-10	Depending on the number of isolated network segments constructed by a company.
6	Hierarchy Level	1	Service providers only request the resources for management and controlling meeting and office network environments.

3.1.5. Contents Delivery Network as a Service

As described in D2.1, for CDN providers, there are two types of users:

- CDN customers (or content provider) renting CDN servers to host their content, and
- CDN end-users who download content from CDN.

The idea of CDN as a service (CDNaaS) platform is the offer of a tool that allows different CDN customers (content provider) to create their CDN slices, on top of different cloud networks, without writing a single line of code or deploying any server.

In large-scale sport events such as the 2020 Tokyo Olympics, tourists from different countries will be visiting Japan to cheer up their athletes. During competitions of their athletes, they may be taking HD videos and photos to share with their friends and relatives. During a game that involves a Finnish team, Finnish tourists may take HD short videos and start streaming them. When the numbers of video sessions exceed a predetermined value, a CDN slice may be created over a cloud in Japan and a cloud in Finland connected through a tunnel with the right QoS. Videos from Finnish tourists are aggregated in a cache at the cloud in Japan and then transferred in bulk to a cache in the cloud in Finland. In this way, friends and relatives of the Finnish tourists, most likely being in Finland, will be having access to these videos from nearby caches. The resources of the CDN slice can be released once the game is finished and traffic from Finnish tourists becomes below a certain threshold.

This use case reflects the same scenario as described in section 3.1.3, “Ensuring QoS on Demand”, yet the scaling out/up a CND slice would be more challenging. When the content provider needs to scale out its CDN slice, the CDNaaS platform needs to perform scaling in different domains. For example, it rents new CDN servers from different cloud providers and deploys in Finland and Japan. It also needs to collaborate with network operators who providing communication services and ensure providing the efficient bandwidth for the communication between Finland and Japan, and also to the CDN end-users in both regions. Nonetheless, scaling (out, up or down) should also be done in a few seconds.

The key performance indicators to scale the Contents Delivery Network as a Service are shown in Table 7.

Table 7 – Requirement of the Contents Delivery Network as a Service

#	Requirement Parameter	Range	Description
1	Number of slices	5-100	A number of slice according to the number of CDN service provider.
2	Location	A city	The smallest size of slices is a company.
3	Frequency	A second	The modification of slice is triggered by CDN service provider requests.
4	Lifecycle Time	A day	The creation and deletion of slices are high to meet event organization where CDN service provider provide communication environments.
5	Heterogeneity	2-10	Depending on the number of isolated network segments constructed by CDN service providers.
6	Hierarchy Level	2	Service providers and CDNaaS provider organize two layers.

3.1.6. Advancement of Medical Services

The advanced communication features to be provided by the 5G systems are expected to raise the advanced medical treatment opportunity such as enhancing the surgery support system resulting in allowing patients to have remote medical surgeries As described in D 2.1, The remote surgery will need support system revising every step in the surgery operation with video image through the surgeon's operation where the extremely high-resolution video camera and surgery support system are connected via the wireless communications. Additionally, it demands the low-latency communications. Moreover, remote surgeries should be performed in anywhere, in the rural areas or even in the moving ambulances, and anytime.

One promising solution for this scenario would be creating MEC (mobile edged computing) slices on demand. When the remote surgery has to be performed, the orchestrator analysis and decides the location of the MEC server, allocates or scale out the resources (computing, storage, network) at the edged cloud and create a network slice. Optimal placement of MEC server will ensure low latency communication.

The key performance indicators to scale the Advancement of Medical Services are shown in Table 8.

Table 8 – Requirement of the Advanced of Medical Services

#	Requirement Parameter	Range	Description
1	Number of slices	10,000-100,000	A number of slice according to the number of Hospital and ambulances.
2	Location	A room of hospital and/or an ambulance	
3	Frequency	A few seconds	A time for a preparation of surgeons to begin their surgery operation
4	Lifecycle Time	A few hours	The creation and deletion of slices are extremely high for the duration of surgery operation.
5	Heterogeneity	2	The video stream, and management and controlling surgery operation.
6	Hierarchy Level	1	

3.1.7. Massive IoT

The massive IoT use case encapsulates for a very large number of applications and domains, including eHealth, Smart Cities, remote control and enhanced map information. To provide various IoT services, multiple network slices can be deployed in parallel.

When we consider the scalability of the IoT network slice, the important parameters would be service type, number of devices, device longevity and service lifespan. Even a single application or service such as smart city consists of millions of devices which will require huge amounts of resources (computing, storage, and network) for operation. In the smart city case, an immediate surge of the total number of devices or traffics are unlike to occur. Since the devices are designed to operate 10 to 15 years in minimum, new devices will gradually be added during these years. Consequently, the resources (computing, storage, and network) will also need to be scaled up/out gradually. This shows that when a new IoT service is introduced and a new network slice is created, it has to ensure robustness for the future growth of devices and traffic volume for next 10 or 15 years. The key performance indicators to scale the Massive IoT are shown in Table 9.

Table 9 – Requirement of the Massive IoT

#	Requirement Parameter	Range	Description
1	Number of slices	100-10,000	A number of slice according to the number of cities.
2	Location	A city	
3	Frequency	A day or a month	Gradual increase/decrease of devices
4	Lifecycle Time	A few years	The creation and deletion of slices are extremely low.
5	Heterogeneity	1-3	If the device network requests multiple QoS types, the multiple types of slices are required.
6	Hierarchy Level	1	

3.1.8. Disaster Handling

In case of disaster (e.g., earthquake, and tsunami) some of the existing communication solutions can be unusable or provide significantly degraded services. It is quite crucial to resume the communication services as soon as possible. From scalability management perspectives, there would be two options: create a new network slice such as emergency network slice as described in D 2.1 or scale up the current network slices and resume the communication service. In the Either case, the time to recovery (TTR) should be a few minutes. Moreover, in such scenario, the network slices or services should be operator independent which means any users should be able to access any network operators' services.

The key performance indicators to scale the Disaster Handling are shown in Table 10.

Table 10 – Requirement of the Disaster Handling

#	Requirement Parameter	Range	Description
1	Number of slices	1	Number of the slice
2	Location	Multiple cities and/or Nationwide	
3	Frequency	A few minutes	According to clarifying damage of disaster, the resource for the slice is modified.
4	Lifecycle Time	Several months	depending on the damage of disaster
5	Heterogeneity	1	Any communication for people safeties are provided in a single slice, and best effort types of communication are the first.
6	Hierarchy Level	1	

3.2. Summary of the requirement and issue

From the clarifications of the requirements in terms of the scalability to manage and controlling slices in the previous section, the required key performance indicators are shown in Table 11.

Table 11 – Required key performance indicators

#	Requirement Parameter	Minimum of Range	Maximum of Range
1	Number of slices	1	1,000,000
2	Location	A room in a hospital, an office, and an ambulance	Nationwide
3	Frequency	A few seconds	A month
4	Lifecycle Time	30 minutes	Years
5	Heterogeneity	1	1,000
6	Hierarchy Level	1	10

To meet these performances specified in the network operators, service providers and application service providers need to ensure the scalable management and controlling of slices. Hereafter these network operators, service providers, and application service providers are referred to as slice operators.

The slice operators should maintain a sufficiently large number of slices (up to 1,000,000). Some of the slices have the same features in communication, and the others are individually different. In terms of the frequency of operating, slices vary from few seconds to a month. The severe slice operation is a high frequent modification of slices, and it forces the acceptable duration to converge a single operation in less than multiple times shorter duration than the frequency interval, such as less than a second for convergence against the few-seconds frequency of slice operations.

Regarding the quick convergence, the operational area (slice area) is another key to be solved. In the case, that slice is constructed or modified in a small area (meeting place, and an ambulance), the actual signaling can be a single network segment (subnet). However, in the case, that slice is expanded over the country (nationwide), the signaling for slice operation should be performed in a parallel manner. Additionally, the slice operators organize layers (e.g., a slice operator provides their resources to the other slice operator and the latter slice operator provides their resource to another slice operator), each layer of slice operator also perform slice operation in a parallel manner in the case of the high frequent slice operation.

4. System Overview

4.1. Introduction

One of the key targets of the upcoming 5G systems is to build a novel network architecture that shall support not only classical mobile broadband applications and services, but also vertical industries (e.g. automotive systems, smart grid and public safety) and IoT-based services. Besides devices operated by a human (i.e. smart-phones and tablets), 5G systems will also include sensors, actuators, and vehicles.

All these requirements have been driven by the envisioned 5G system use-cases. Indeed, several SDOs and ongoing 5G research projects have defined different 5G system use-cases with different targets.

Before detailing the technical details on 5G!Pagoda orchestration and management system, it is important to understand what are the differences between orchestration and management. According to IMT-2020 [1], the definition of orchestration and the management are described as follows.

- Orchestration
  The processes aiming at the automated arrangement, coordination, instantiation and use of network functions and resources for both physical and virtual infrastructures by optimization criteria.
- Management
  The processes aiming at fulfillment, assurance, and billing of services, network functions, and resources in both physical and virtual infrastructure including compute, storage, and network resources.

The adopted definition in 5G!Pagoda follows the definition of IMT-2020 and from the viewpoint of resources it should be simply described as follows: the “Orchestration” will allocate the necessary resources based on some automation algorithm and the “Management” will secure the allocated resources.

4.2. Overview of the system architecture

The envisioned common architecture of the 5G!Pagoda orchestration system is illustrated in Figure 1. Deliverable 2.1 “Use case scenarios and technical system requirements definition” (D2.1) has already provided a basic system architecture and the system architecture in Figure 1 follows this basic architecture and gives more technical details from the viewpoint of the scalable DSSO orchestration and management.

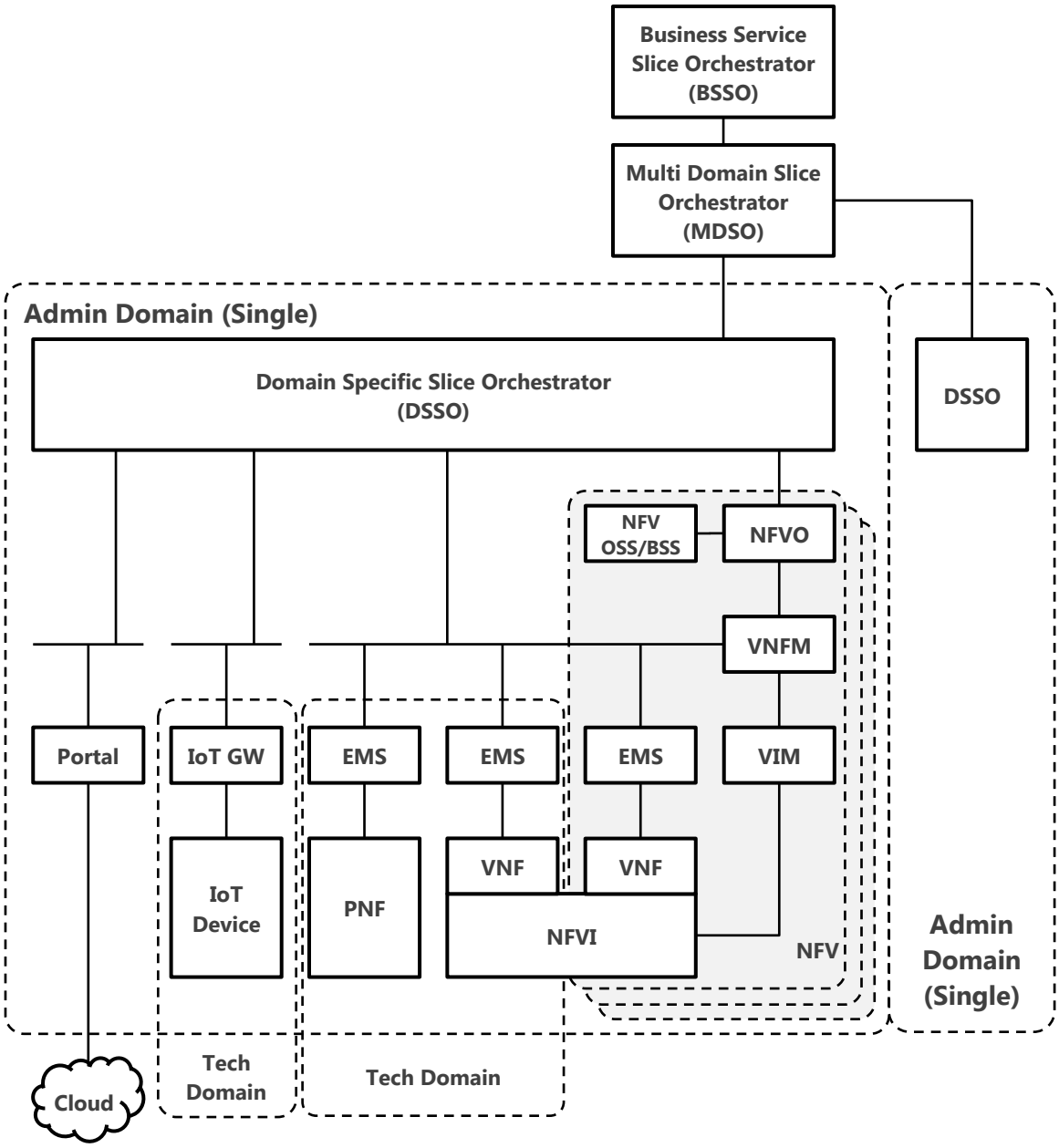

Figure 1 – System Architecture

Figure 2 shows the scope of management and orchestration described in section 4.1.

The orchestration is described in Chapter 5 and management is described in Chapter 6.

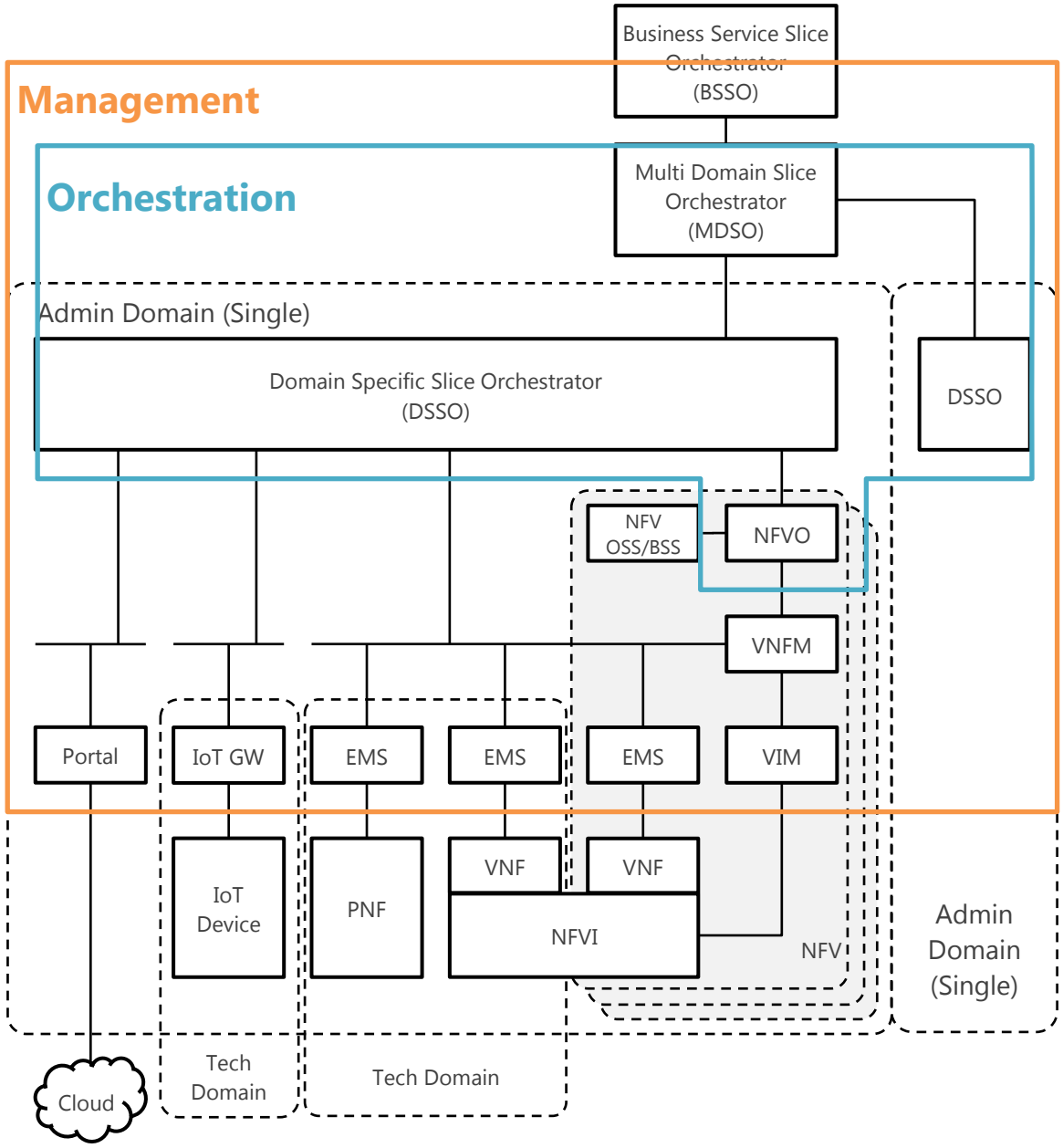

Figure 2 – Management and Orchestration Architecture

4.2.1. Domain-Specific Slice Orchestrator (DSSO)

The domain-specific slice orchestrator (DSSO) is in charge of the end-to-end orchestration. DSSO is like the NFVO defined by ETSI MANO but with additional functions required to interact with the multi-domain slice orchestrator (MDSO), for example, north-bound APIs for the multi-domain slice orchestrator to consume.

DSSO has the following functions.

- Slice Orchestration and Management
  Slice is a space where end users can utilize the services by slice service provider. Slice management functions are as follows:

- slice creation
- slice deletion
- slice monitoring.

- Resource Management
  Resource management functions manage resources to implement slice service. Resource management functions are as follows:

- resource registration to a resource pool
- resource allocation based on the request from the service management

- Service Management
  Service management functions manage services to the end users. Service management functions are as follows:

- service creation
- service deletion
- resource request to resource management

It should be noted that DSSO can exchange information between multiple NFVOs in the same administrative domain, allowing to provide separate slice corresponding to each NFVO. Likewise, MDSO (Multi-Domain Slice Orchestrator) can exchange information between multiple DSSOs. Typical information is the life-cycle management of slice, which is only allowed to run on a specific administrative domain. Note that this sort of functionality is already offered by the northbound API of the NFVO in the form of the processing of NSDs. A DSSO and NFVO may be integrated into the same entity.

4.2.2. NFV

4.2.2.1 NFVO

The target technology domain which the DSSO will manage is not only limited to NFV technology domain but also PNF and/or IoT devices technology domain as described in Figure 1.

The Network Function Virtualization Orchestrator has the functionality defined by ETSI NFV with its two roles:

- Resource Orchestrator (RO) enabling the brokering of the NFVI resources between the multiple, parallel slices. The NFVO represents an aggregation point for the administrative domain for resources management. The NFVO communicates with multiple VIMs and WIMs and is able to allocate the resources appropriately across them.
- Network Service Orchestrator (NSO) providing indications on how the system should scale and where the network functions should be placed following the Network Service Descriptor (NSD) information.

The NFVO provides scaling functionality to replicate an NFV or resize an NFV when the load is increased. Additionally, the NFVO is extended to support additional commands which result in the dynamic changing of the Network Slice Blueprint information. By this, the active service can be dynamically modified during runtime with additional actions compared to the static Network Slice Blueprint based decisions. For example, with this new functionality, new VNFs can be added during runtime to a running system (e.g. a more performant firewall in case of a network attack).

4.2.2.2 VNFM

The role of VNFM is defined by ETSI MANO specification. It should:

- allocate appropriate resources to the VNFs or to delegate this operation to the NFVO,
- receive events on the completion of the specific operations and information on the dynamic configurations,
- configure the Element Manager (EM) of VNFs with the dynamic configurations and control the execution of lifecycle events.

4.2.2.3 VIM/WIM

The Virtual Infrastructure Manager (VIM) is defined in the ETSI NFV architecture. Additional functionality of the VIM includes the capability to control the data plane functionality such as in the form of an SDN controller or an ICN or CDN information and content control in order to be able to provide a separation of the data plane when the data traffic is directly routed through the network (i.e. deep data plane programmability). The inclusion of the SDN Controller is common in many 5G PPP projects. The Wide Area Network Infrastructure Manager (WIM) has the role to define the virtual networks between different parts of a slice on top of common transport networks (i.e. the inter-data center environment sharing rules).

4.2.2.4 NVFI

5G!Pagoda uses a similar NFVI as defined in the high-level ETSI NFV architecture. However, a specific implementation of the virtual network is considered, covering deep data plane programmability and inter-data center WANs.

4.2.2.5 VNF/PNF

The VNF is a component of virtual network function and the PNF is a component of physical network function. These two will be utilized mostly as the functional resource, which DSSO manages. A VNF can consist of multiple virtual machines linked together.

The DSSO will deploy VNFs via NFVO and VNFMs and control VNFs as well as PNF via EMS.

4.2.2.6 EMS

The EMS is an Element Management System and it will manage and control VNFs and PNFs. The 5G!Pagoda EMS is the same as the EMS of ETSI NFV or EMS of legacy systems.

4.2.3. Multi-Domain Slice Orchestrator (MDSO)

The main role of MDSO is to provide a slice on top of multiple administrative domains. It contains the following functionalities:

- Receives requirements from BSSO on the requirements for the specific slice. The requirements may be received in a static description form such as TOSCA or an NSD file.
- Selects the applicable domains for the slice and locates the DSSO of each domain.
- Establishes secure connections to multiple DSSOs.
- Acquires, if permissible, knowledge on the available resources in the specific administrative domains in terms of available infrastructure and available services (e.g. stored virtual machine images).

- Negotiates with the DSSOs the resources and their locations to be allocated for a slice customer.
- Makes decisions based on the requirements received on the split of the slice functionality across the multiple administrative domains.
- Commands the installation of the slice over the multiple administrative domains.
- When the installation is successful, exchanges connectivity parameters between the different DSSOs to be able to stitch together the slice.
- Announces the tenant through the business slice orchestrator on the successful installation of the slice as well as on the connectivity and management points.
- Informs the tenant, through the business slice orchestrator and/or the slice-specific OSS, of any SLA breaches or any other types of major failures of the deployed slice.
- Disposes a slice from the multiple domains.

The MDSO implements Slice Placement Function that is responsible for allocating and interconnecting slice-specific virtual network functions (VNFs) according to resource constraints and service requirements, e.g. related to latency. The task of the multi-domain Slice Placement Function is the selection of domains covered by the slice. In particular, the set of domains must be connected to ensure a path from each domain to each of the other domains. For this purpose, it may be necessary to select from a set of alternative domains the domains to be on the path between two end domains, e.g. between the two domains of a UEs participating in a call. For the above purpose, the multi-domain placement function is required to have information on available resources and cost of intermediate domains. Based on this information, the multi-domain slice selector can consult a domain selection policy function to, according to given policies, determine the domain to include into the slice. Domains can also be proactively covered by a slice, even if there are no UEs or transport needs connected to that domain. In particular, for placement of services and VNFs, domains such as generic virtualization infrastructure providers, can be included. This may be e.g. based on the price of hosting infrastructure, a central location or the availability of resources (e.g. bandwidth). The decision is made by the domain selection policy function. VNF Placement function may use different placement algorithms designed and evaluated in WP3 (task 3.3) and WP4 (task 4.2) of 5G!Pagoda project.

4.2.4. Business Service Slice Orchestrator (BSSO)

The BSSO has the role of a portal to advertise the possible services, to trigger their deployment and in case of success, to transmit to the slice administrator the specific entry points to the new slice management. It is also used by slice tenant to reconfigure their slices after slice deployment. The BSSO provides the interface toward the slice operator (a tenant or vertical), including the API to query for the availability of resources and blueprints, pricing information and status of deployed resources as well as the API for uploading new blueprints, deploying new slices and destroying slices. The BSSO is connected to one or several multi-domain slice orchestrators. The BSSO also interfaces the OSS/BSS of the domains in which it offers services.

The BSSO is the interface to the slice tenant. It stores a registry of slice descriptions/blueprints of the tenant. The tenant can manage slice description by uploading, modifying and deleting slice descriptions. It can set the pricing information and the policy and specific slice specific requirement. The BSSO may also provide a user-friendly interface for creating and editing slice descriptions, e.g. graphically rather than specifying raw TOSCA descriptions. It may provide components, e.g. a basic EPC network, as a starting point,

which can be modified by the user. It may further provide a library of preconfigured NFVs that can be included in the slice. The BSSO is connected to one or several MDSOs based on contracts. When the creation of a slice is triggered, it selects the MDSO most suitable for the specific slice based on business requirements. For slices split between multiple domains, the BSSO provides the slice description to the MDSO. The MDSO receives the slice specific requirements from BSSO as a slice description in a standardized form such as a TOSCA descriptor packaged as a CSAR. The MDSO locates the relevant DSSOs, performs authentication and establishes secured connections to them. The MDSO acquires from the DSSOs knowledge on the available resources in the different administrative domains in terms of available infrastructure and available services (e.g. stored virtual machine images). The MDSO then negotiates with the DSSOs the resources and their locations to be allocated for a slice customer. This may also involve cost negotiation. The MDSO deciding the split of the NFVs of the slice between the domains based on the requirements. This involved splitting the slice description into separate slice descriptions for each destination domain. The MDSO also plans the stitching and possibly modifies the slice description to include interfaces necessary for the split. The MDSO installs the slice over the multiple administrative domains by providing the domain-specific slice description, which is sent in a standardized format such as TOSCA packaged in a CSAR. The MDSO waits for completion of the slice setup. When the installation is performed, it exchanges connectivity parameters between the different DSSOs to be able to stitch together the slice. Once the slice is correctly set up, the DSSO announces the tenant through the BSSO and provides the BSSO about created connectivity and management points. If necessary, the MDSO informs the tenant via the BSSO of any SLA breaches or any other types of major failures of the deployed slice.

4.2.5. Others

The 5G!Pagoda DSSO will be able to create the slices by combining and utilizing the multi-vendor resources. It means that the 5G!Pagoda DSSO architecture will allow an integrated use of a variety of resources, i.e. cloud and/or (IoT) devices component as well as network component represented by VNF and PNF.

5. Scalable Orchestration

This chapter addresses scalable orchestration solutions to meet the requirements described in Chapter 3, which is based on the architecture described in Chapter 4.

5.1. Scalable Orchestration Policy

This section addresses some basic ideas to achieve scalability in the 5G!Pagoda orchestration system. Three KPIs to evaluate the performance of the scalable orchestration system are introduced.

These KPIs are applicable to the corresponding instance, which is necessary to be scalable and thus are universal for all the use cases described in Chapter 3.

Three KPIs are described as follows:

- Number of Instances

  This KPI indicates a total number of instance objects, which the DSSO should control. If there are some attributes, which are linked to the managed object, the number of those attributes should be counted as well. This KPI mainly corresponds to “#1 Number of slices”, “#5 Heterogeneity” and “#6 Hierarchy Level” of Table 11 in section 3.2. If there are many Heterogeneity KPI and Hierarchy Level KPI, then the number of instances that the orchestrator needs to control will increase.

- Status Change Frequency of Instances

  This KPI indicates a total number of operations for the managed instance objects (ex. Slice operations such as creation, modification, deletion, etc.) in a fixed period executed by the DSSO. This KPI corresponds to “#3 Frequency” and “#4 Lifecycle Time” of Table 11 in section 3.2. This means that short lifecycle time will lead to more frequent status change.

- Latency of Compute Processing

  This KPI indicates a computing process delay and is closely related to the computing resource scalability.
  This KPI corresponds to “#3 Frequency” and “#6 Hierarchy Level” of Table 11 in section 3.2. Although “Latency of Compute Processing” heavily depends on “#3 Frequency” , the former two KPIs, “Number of Instances” and “Status Change Frequency of Instances” are most critical. Therefore, it can be claimed that the instances with large numbers and frequencies will need to be controlled as quickly as possible to minimize the latency of computing process.

If the number of the controlled objects is the only item to increase, it will cause a major problem. However, significant orchestration performance degradation will occur if a state of a large number of managed objects changes too frequently, or if it becomes necessary to change the state of a large portion of the controlled objects in an extremely short amount of time. The above scenarios show that additional requirements such as Frequency and/or Latency combined with Number will cause a big impact on the orchestration performance.

Thus, it should be mandated to reduce the number of objects to be monitored within the same time frame. Two practical methods are known for the scalable orchestration, and they are described below:

- Information Aggregation

  It is possible to reduce the number of the controlled objects by aggregating the information for the controlled objects at the same time. Note that this orchestration system can control more objects in the higher orchestration layer compared to the lower orchestration layer by introducing a hierarchical orchestration technique. For example, the orchestration utilizing the resource pool which is described in Chapter 3 is one of those methods which can reduce the information volume by controlling a relevant resource in a unified format and thus cope with a quick change of state.

- Scope Limitation

  It is possible to limit the scope in which the orchestration system should control at the same time frame, so that the number of the controlled objects does not explode in the entire system. Note that this orchestration system can virtually reduce the number of the controlled objects by classifying those controlled objects into multiple areas.

These two methods mentioned above are usually utilized in a combined manner in the real orchestration operation system. Chapter 3 has already addressed the requirements for the scalable orchestration in each 5G use case based on the proposed metrics, and it is expected to utilize the method described in this Chapter.

5.2. Resource Pool

The ‘Resource Pool’ is an information model of logical entities to represent virtual resources corresponding to the physical resources offered by a variety of appliances such as cloud, IoT devices, PNF and VNF.

The relationship between ‘Resource Pool’, the DSSO and infrastructures composed of various kinds of physical and virtual appliances are illustrated in Figure 3. The ‘Resource Pool’ is controlled by the DSSO.

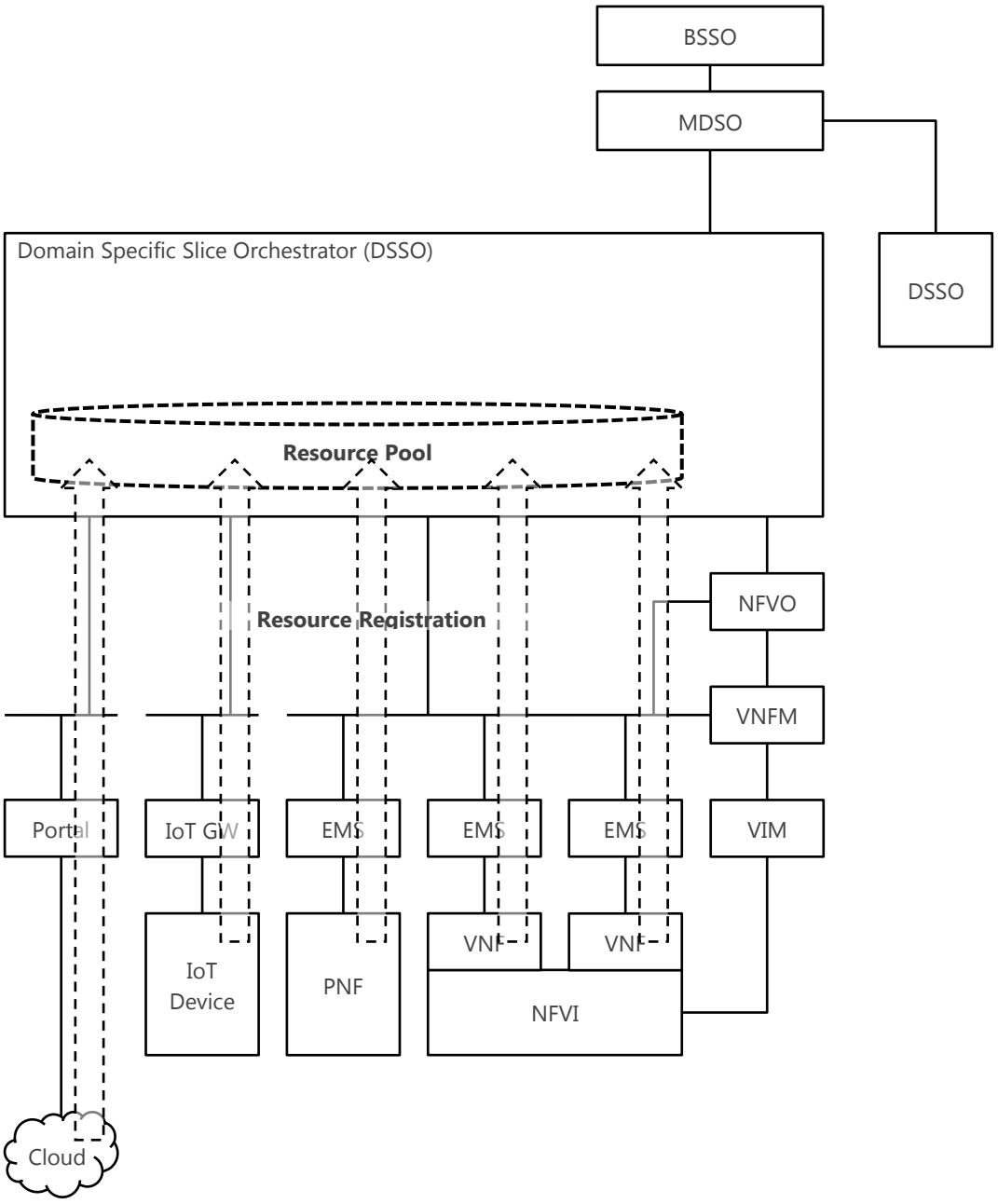

Figure 3 – Resource Pool

The main benefit brought by the 'Resource Pool' is focused on the function called 'Response Time Accelerator'that is intended to accelerate service velocity from the viewpoint of the DSSO client which is typically an OSS/BSS.

The 'Response Time' function has two components. The first component is a provisioning time needed to configure the underlying infrastructure by the 'Resource Pool', and the second one is a smart demand forecasting provided by the OSS/BSS.

Hereafter, the demand forecasting is assumed to be correct. Supposing the situation where a service provider wants to start a new network service for his customers, it will conventionally take at least one week for his network operator to prepare and test the dedicated circuits alone.

However, if the demand forecasting by the OSS/BSS is correct, then virtual resources which were necessary and sufficient for that service could be pre-provisioned almost instantly, for example, within less than 10 seconds. Thanks to the technology like a smart demand forecasting, the network operator will be able to provide a new agile service delivery platform not only to service providers but also to end customers as well.

The basic concept of the 'Resource Pool' is similar to the server and network virtualization technology used in the cloud, but the main difference is an attempt to apply to the E2E slice in a single administrative domain. The introduction of a smart demand forecasting by the OSS/BSS will bring a great benefit.

From the digital transformation perspective, the role of the 'Resource Pool' can be summarized as follows:

- To present all of the available virtual resources to the service providers and network operator before starting the actual services (Digitalization [26] [27])

  The resources which resource pool can manage will depend on the functions that each appliance has in nature. The resources are diverse, for instance, CPU frequency, number of cores, memory size and storage space are typical computing resources. Bandwidth, latency, and address space are typical network resources. Furthermore, router, gateway, firewall, and even the radio frequency of IoT devices can be represented as virtual resources. All the mentioned virtual resources can be represented as digitalized virtual assets.

- To develop and operate (start/stop) agile services using allowed virtual resources provided by the service providers and network operator (DevOps Support)

  The resources which 'Resource Pool'can manage will become instantly available after they have been represented as virtual resources at the first registration time during the pre-provisioning stage. Therefore, once the service using the pre-provisioned virtual resources are created, a system configuration for that service will be instantly completed. The DSSO is programmed to allocate more resources to the growing services and deallocate surplus resources for the shrinking services under the direction of the OSS/BSS and thus can support DevOps functionality.

5.3. Slice

One of the characteristics of the 5G!Pagoda orchestration system is that the slice is classified either as a specific slice or an end-to-end slice. Note that both slices are loosely coupled as illustrated in Figure 4.

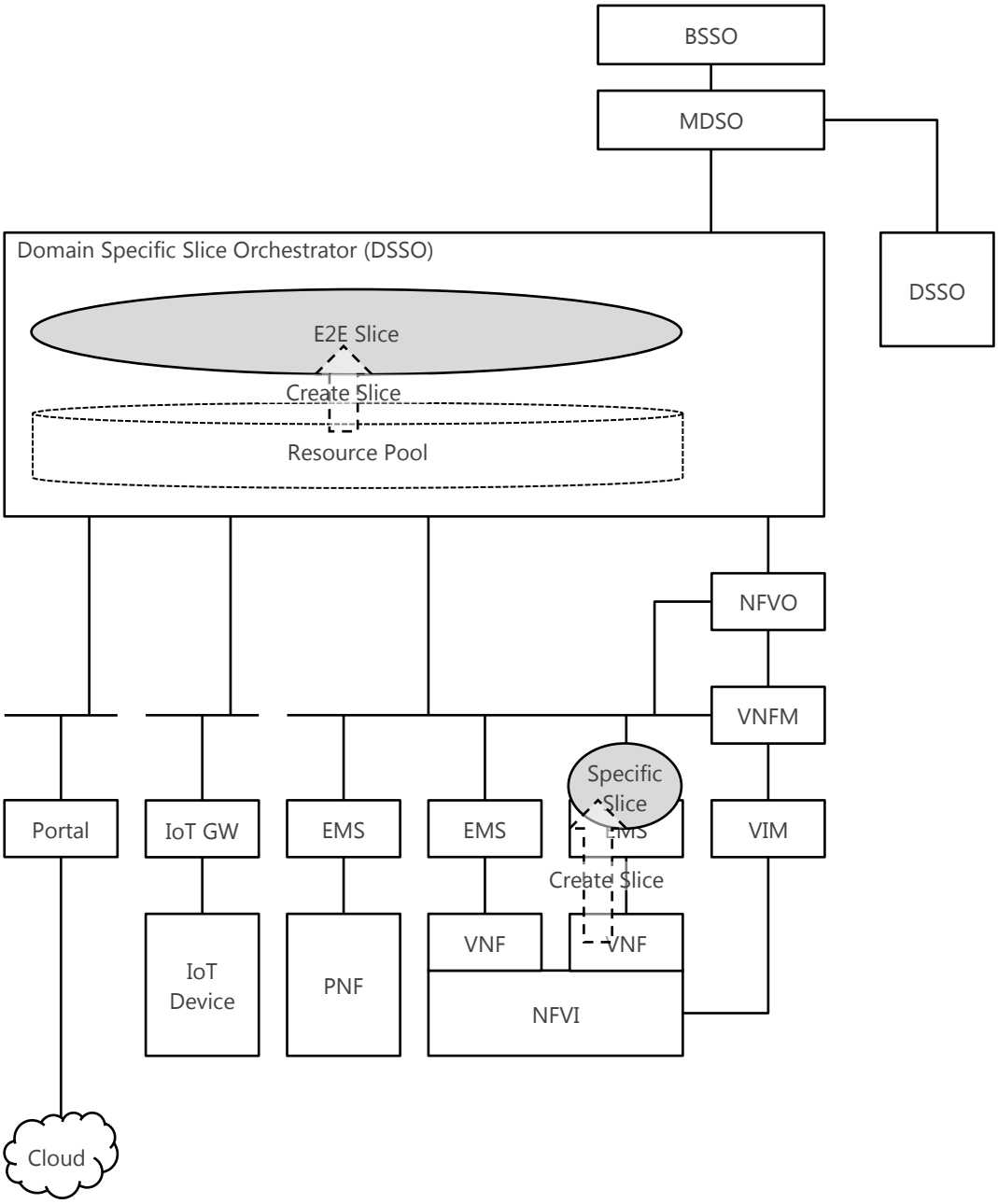

Figure 4 – Slice

5.3.1. E2E Slice

The E2E(end-to-end) slice which the DSSO is intended to manage is the slice where services are built using all the resources ranging from one endpoint of a certain technology domain, e.g. Cloud to another endpoint of the opposing technology domain, e.g. IoT Device within a single administrative domain. On the other hand, the E2E slice, which the MDSO is, intended to manage ranges from an endpoint of a certain administrative domain to another endpoint of the opposite administrative domain.

Since the resources managed by the DSSO are diverse, it will be necessary to create an E2E slice by combining networking, cloud, and other technologies. The scope of the E2E slice in 5G!Pagoda is not limited to that of the 'vEPC slice' realized by NFV management and orchestration system. Thus, the E2E slice

orchestration system architecture will need to consider more advanced technologies expected in the near future, as well as existing technologies, e.g. NFV.

Most of the technology domains, e.g. carrier cloud, optical, and packet transport system can be built by employing NFV MANO technology, but a single administrative domain, which aggregates multiple technology domains, will be built by employing a high quality and performance E2E slice by combining NFV and an emerging IoT technology.

To provide a benefit of the E2E slice to end users, it will be necessary to consider the following requirements for the E2E slice creation and management.

- To manage a carrier-grade communication session, e.g. FCAPS
- To realize high-speed provisioning and business agility by scaling out and in
- To handle a huge number of slices with scalabilities
- To realize slice functions with flexibilities
- To realize slice federation between different administration domains

5.3.2. Specific Slice

Specific slice is a slice that is created within a specific technology domain and thus is a manager of that technology domain, e.g. EMS can only manage a network within the scope of that slice. The function provided by a specific slice is different from one slice to another. However, it should satisfy some of the E2E requirements because a specific slice is a part of the E2E slice.

The common and dedicated slice described in D2.3 is a kind of the specific slice which is closed within a certain technology domain and inherits ‘Common Slice’ and ‘Dedicated Slice’ architecture.

Specific slice usage depends on the functions realized by that technology domain, and typical usage sequences are as follows:

- A device, which has a function to manage a specific slice, starts up autonomously without any instructions from the DSSO.
- The device mentioned above creates ‘Common Slice’ for initial setting.
- The device creates ‘Dedicated Slice’ via ‘Common Slice’.

5.3.3. Slice Stitching

There exist various requirements to realize 'Slice Stitching' in the real world. Management of the address space, which is managed individually at each slice, exchange of routing information, protection, and data exchange between slices, are addressed.

Physical connection patterns to realize 'Slice Stitching' are illustrated in Figure 5. Since the term 'federation' is generally used in the SDN research community, hereinafter it is referred to as 'Slice Federation'.

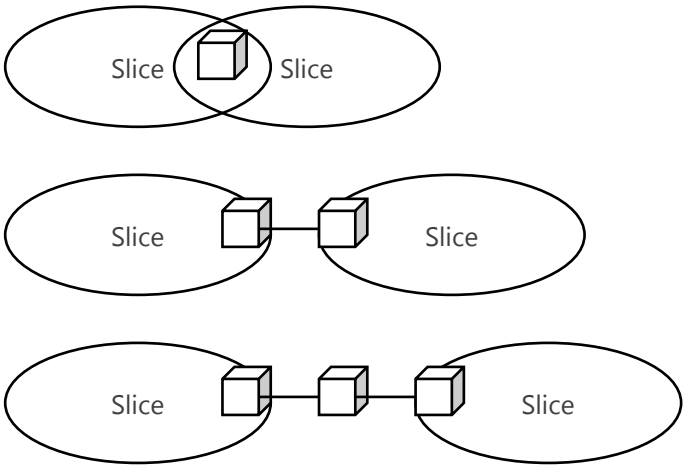

Figure 5 – Slice Stitching Patterns

Slice federation procedures over multiple administrative domains are illustrated in Figure 6**Error! Reference source not found.**.

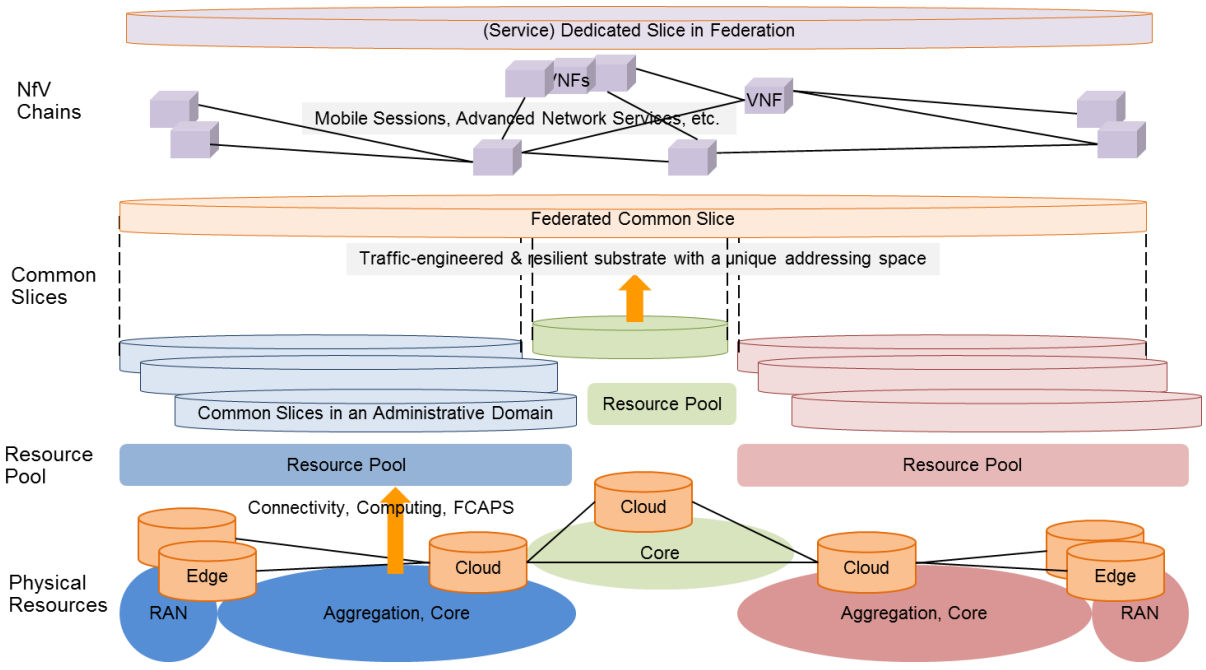

Figure 6 – Slice Federation Procedures

Slice federation allows a single virtual system across multiple administrative domains. The MDSO has 'Slice Federation' function, but the DSSO does not because 'Slice Federation' is only necessary to combine multiple slices over multiple administrative domains, and 'Specific Slice' management over multiple technology domains is consistently maintained owing to a single management policy under the DSSO administration.

The goal of slice federation is as follows:

- Unique addressing space and isolation
- Advanced E2E network services utilizing various VNFs and SFC (Service Function Chaining)
- Distributed computing resource utilization

Integrated virtualized resources will be created as long as the functional requirements listed in Table 12 are fulfilled, and as a result, scalable and dynamic utilization of virtual resources will be achieved.

Table 12 – Slice Federation Service Functions

#	Slice Type	Nodes	Functions	Description
1	Common Slice	Exchange GW	・Protocol Conversion ・Address Space Conversion	Will not be scalable
		VRF	・Slice Boundary Routing ・Routing Table Synchronization	
		L2 over L3 Tunnel	・Global IP Address Space Utilization	
2	Dedicated Slice	Service GW	・Federated slices Access VNFs as a User	Will not be scalable
		Resource Ceding	・Common Slice Utilization in the Visiting Domain	Use visitor's resource as if it were its own resource.
		Component Service	・VNFs utilization in the Visiting Domain as a Service Component	
		Roaming	・User Authentication for VNFs in the Visiting Domain	Ex) HLR/HSS Federation Use visitor's resource without any contract

Furthermore, the following functions are necessary to achieve more sophisticated slice federation.

- Protocol agnostic
- Service lifecycle management
- Operational visibility
- Automatic operation and autonomous operation
- Intuitive, intent-based networking
- Programmable, self-service

The procedures of each slice federation function class are summarized in Table 13.

Table 13 Procedures of Slice Federation Functions Class

#	Slice Federation Function Class	Common Slice Function Procedures (Substrate)	Dedicated Slice Function Procedures (Network Services)
1	Session Information and Control Exchange		·VNF Specification Exchange ·VNF Control Info Exchange ·Service Mapping Info Exchange
2	Service Catalogue	·Resources Info Collection ·SLA(TE, Resiliency) Info Collection	·Service Graph Info Collection ·SLA(TE, Resiliency) Info Collection ·VNF Info Collection
3	Functions Exchange	·VNF Instantiation ·VNF Management ·VNF Configuration ·VNF Monitoring	·VNF Instantiation ·VNF Management ·VNF Configuration ·VNF Monitoring
4	Resources Exchange	·Resource Pool Info Exchange ·Abstraction Infor Exchange ·I/F to PHY Resources Info Exchange	·Resource Pool Info Exchange ·Abstraction Info Exchang4e
5	Transport	·Topology Discovery Info Collection ·Reachability Info Collection ·Forwarding Policy Info Collection ·SLA(TE, Resiliency) Info Collection	

5.3.4. Slice Migration

Adding new features to the network often requires a tedious integration to support the new and old features. Network slicing allows the operator to setup a test network in a separate slice to test and verify the integration. This test slice can be based on a snapshot of the live slice and contain additional components for testing. The connectivity to the external networks can be limited. Test tools and validation mechanisms can be run on this slice without risking interruptions in the live slice. Moving real UEs to the test slice allows small-scale test and performance comparison between the old and new setup. Once the tests are completed, the test slice can be switched into a live slice, replacing the former live slice and UEs gradually moved to the new slice. This can be done by updating the slice selection mechanism. [16]

If the slice contains the PGW the user need to re-attach (which terminates current traffic) to the new slice, as current standards do not allow relocation of the PGW with active sessions present.

5.4. Information Model

This section describes the information model that will realize the scalable orchestration policy mentioned in the previous section.

It is possible to clearly map the relationship between each component of the concept level architecture into the implementation by introducing the proposed information model.

5.4.1. DSSO Information Model

Firstly, the DSSO information model corresponding to the system configuration in Figure 1 is illustrated in Figure 7.

In this information model, it is assumed that the expression of components necessary for the scalable orchestration has fewest elements as possible.

There are three main components, i.e. Resource, Service and Slice. The auxiliary information element is necessary to link these three main components.

This consideration of the modeling should avoid the complexity of the implementation by reducing the total number of constructed elements and lead to manage more use cases other than described in Chapter 3.

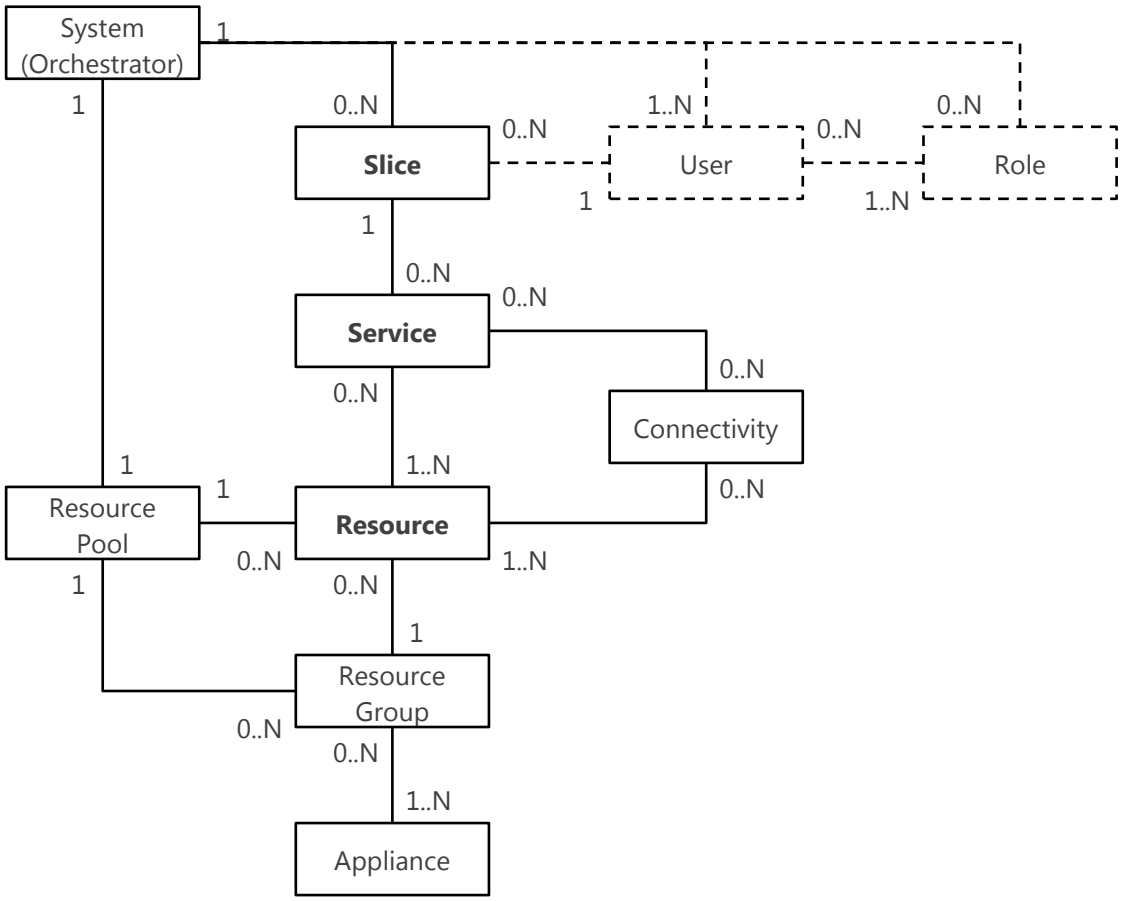

Figure 7 – Orchestrator Information Model

Each component is described in Table 14.

Table 14 – Component description of Information Model

#	Component Name	Description
1	System	Orchestrator
2	Slice	Group which manages various services
3	Service	Functions which are provided by the combination of various resources
4	Resource	Components which realize service
5	Resource Pool	Logical entity which pools resources
6	Connectivity	Connection information between resources
7	Resource Group	A group of resources First, when an appliance includes some resources, all the resources are assigned to the resource pool as a single resource group. Second, when the resource group is assigned to a service, those resources are divided into a separate resource.
8	Appliance	A physical and virtual object which can have arbitrary functions. For network service, that appliance is, i.e. VNF and PNF.
9	User	An account information which is managed by the orchestrator
10	Role	A role assigned to the user.

5.5. Sequence

Three different types of registration sequence are illustrated in Figure 8, Figure 9 and Figure 10, respectively.

In these example sequences, the DSSO has the orchestrate functions that manage the Service, Slice and Resource Pool according to the information model in Figure 7.

5.5.1. VNF type resource registration

Figure 8 shows a basic sequence to register the resource from the VNF type appliance.

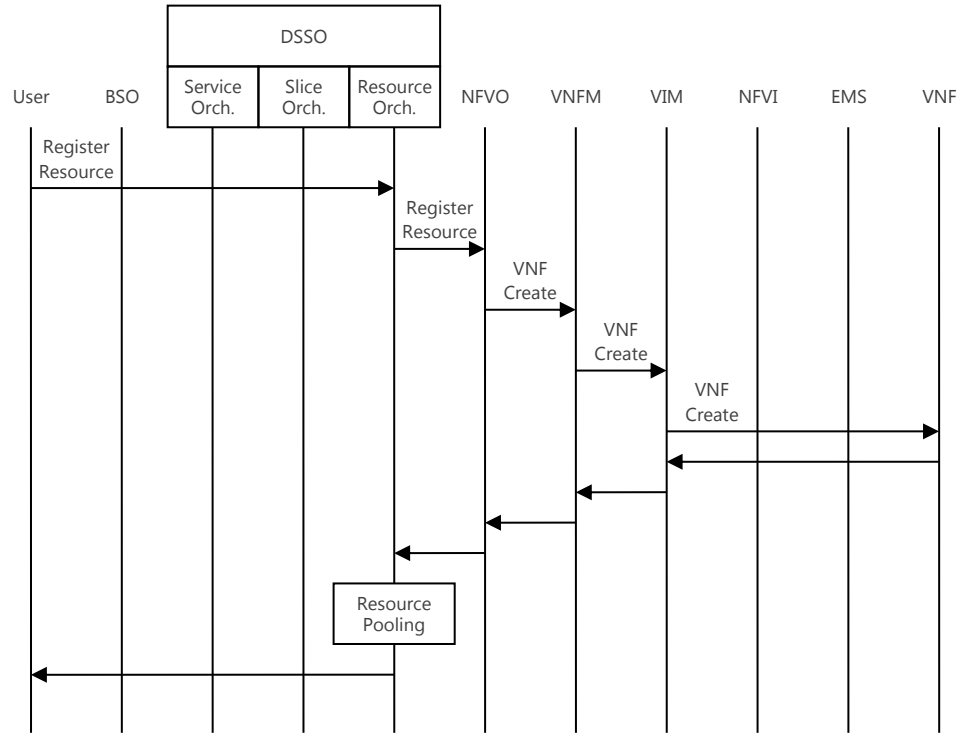

Figure 8 – Sequence of resource registration from VNF

This sequence shows a basic sequence to register a VNF type of appliance as a resource that the orchestrator manages.

In this example, the leftmost user who is a VNF resource orchestrator triggers the resource registration operation, and the resource orchestrator in the orchestrator operates as a proxy between the user and NFVO. The NFVO deploys the VM and VNF on the NFVI via the VNFM and the VIM based on the request from the resource orchestrator.

Some use-cases indicate the situations where the orchestrator needs to register not only the VNFs but also the service that the NFVO will provide as a resource. In that case, the resources can be registered simultaneously by making use of the NSDI which is a service descriptor provided by the NFVI.

5.5.2. PNF (or other Physical) Type resource registration

Figure 9 shows a basic sequence to register the resource from PNF type and any other physical devices.

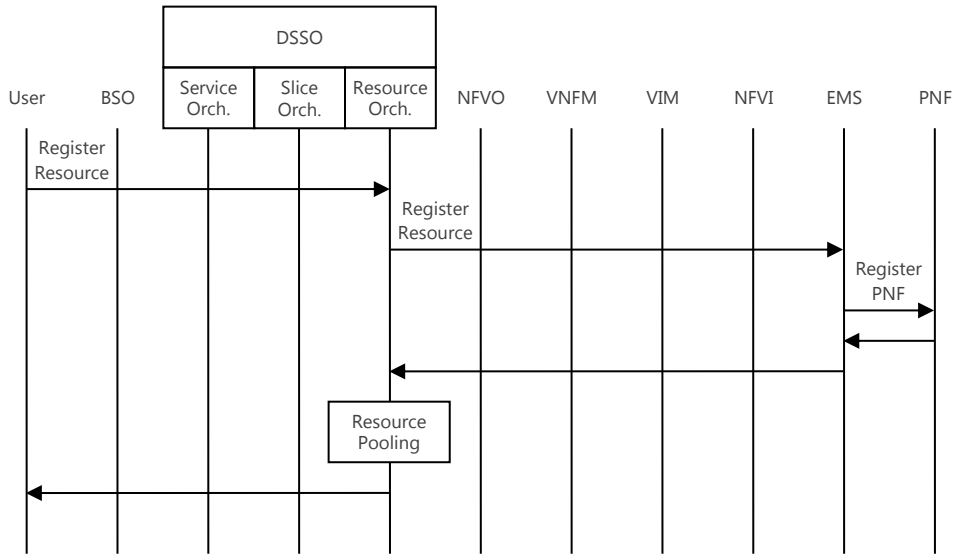

Figure 9 – Sequence of resource registration from PNF

This sequence shows a basic sequence to register a PNF type of appliance as the resource that DSSO orchestrate. Unlike VNF, it is not necessary to deploy PNF, but sequences such as registration from the EMS are required for some PNF.

In this example, the leftmost user who is a PNF resource orchestrator triggers the resource registration operation, and the resource orchestrator operates as a proxy between the user and EMS. In some cases where there is no EMS, it is necessary to access the PNF directly from the orchestrator. In either case, the resources can be registered simultaneously by using a key descriptor like NSDI depending on each use case.

5.5.3. Service Creation

Figure 10 shows a basic sequence to create a service by the user request.

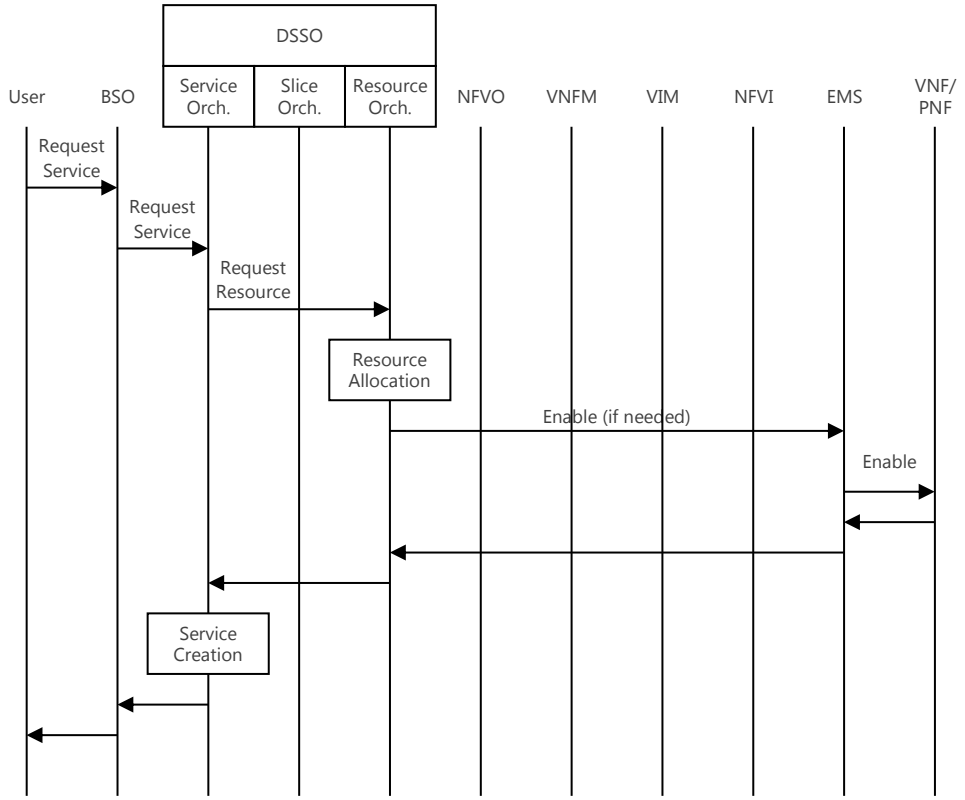

Figure 10 – Sequence of service creation

This sequence shows a basic sequence to register a service using the previously registered resources.

In this example, the leftmost user is a service orchestrator triggers the service registration operation、and the requested service is constructed to use the allocated resource controlled by the orchestrator. When registering the service, the orchestrator configures the appliance according to the necessity. In this case, the orchestrator configures the Enable setting of the appliance via the EMS. The configuration details are depending on the target appliance.

6. Implementation examples of scalable slice management

In this chapter an example implementation of the 5G!Pagoda management and orchestration concept is presented in Figure 11 which illustrates the positioning of each section in this chapter from the viewpoint of the system overview described in Chapter 4.

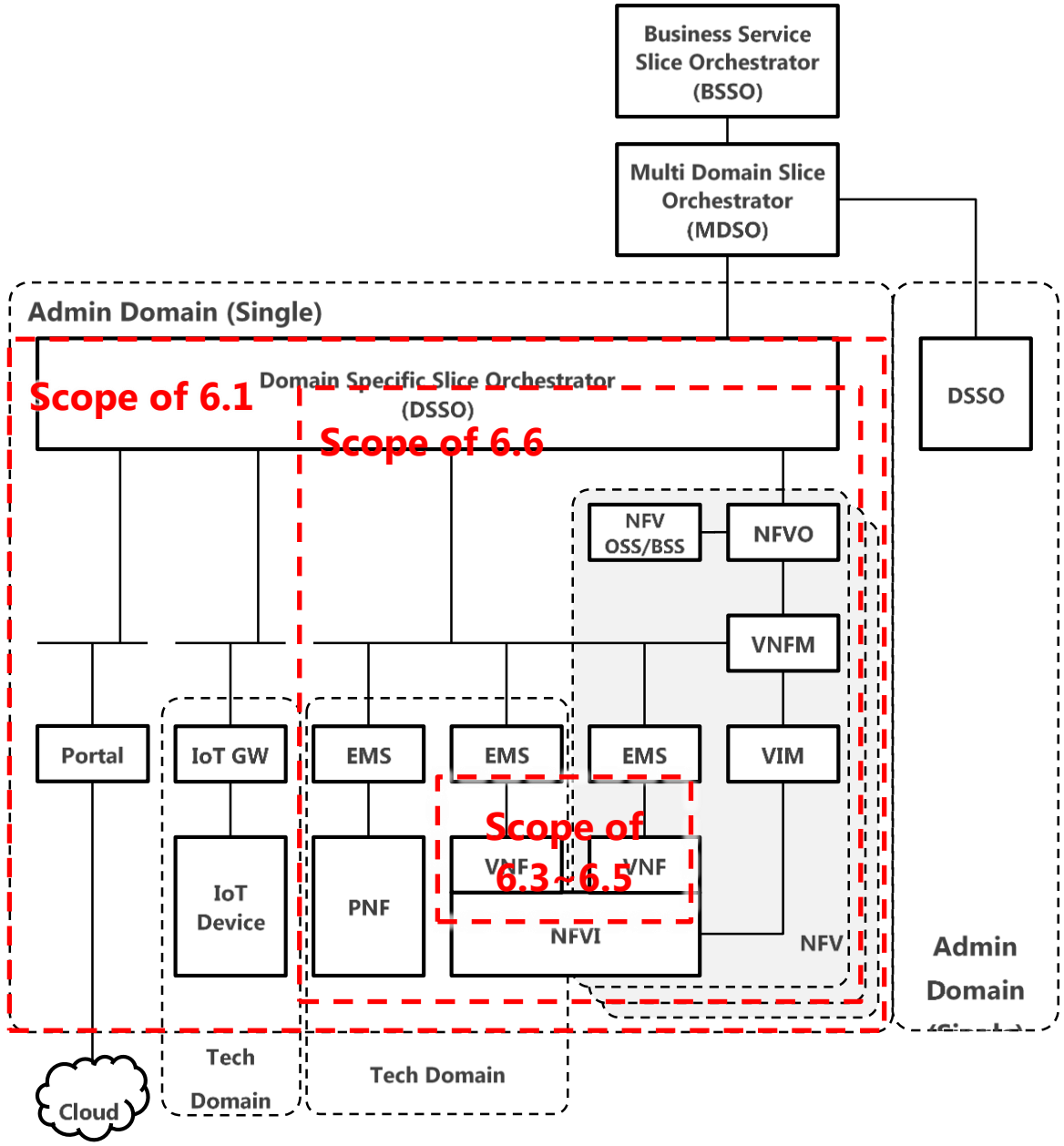

Figure 11 – Positioning in the common architecture

6.1. Implementation of scalable management in a single and multiple NFVO domains

In this section, a single and multiple NFVO domains scalable management and orchestration implementation that is compliant with the overall concept presented in Figure 1 is described. The single NFVO case is used for the description of the concept in details and later on extensions in order to support multiple NFVOs of single DSSO are presented. For clarity in the figures of these sections only virtual network functions are presented, however, also legacy hardware subsystems or nodes (PNFs) can be easily included. The MDSO and BSSO don't concern in this analysis because the mentioned functional block does not raise the scalability issues.

In the state-of-the-art section of this deliverable, there have been described different approaches to management and orchestration of network slices. The analysis has shown that so far there is no complete solution that addresses all issues of network slice management and orchestration. The existing approaches can be characterized as it follows:

- All of them use ETSI MANO [10] part to orchestrate the VNF based part of the network. In some cases, MANO modifications have been proposed.
- In most concepts, the role of OSS/BSS, as well as Element Managers of the MANO framework, is not clearly defined.
- In some concepts, there are present MAPE components of the autonomic network management typically added as a part of OSS/BSS – i.e. external to a managed network or network slice.
- In order to provide multi-domain orchestration, in some concepts a hierarchical approach (in line with ETSI IFA013 [12] recommendation is used.
- The management interface to slice operator is not explicitly defined in the analyzed cases.

It has been decided to apply the following approach:

- The architecture should be split into Common and Dedicated slices as defined in deliverable D2.3. Moreover, each slice has its dedicated management system that is brought to life together with the slice.
- The slice management can be used by slice operator (for example a vertical) for managing the network slice and its services during the slice runtime. It has been assumed that these operations should give a possibility of flexible configuration of the slice and provide the information about slice KPIs to the slice operator. This slice management interface should be simple and lightweight. All the operations related to performance troubleshooting of slices operations should be automated. Such automation can be performed by implementing the autonomic network management paradigm (generally in line with ETSI GANA [6] approach)) in the slice.
- In order to increase management scalability and shorten the management system response time, the VNFs that compose a slice should have the embedded management functionality i.e. a kind of local ANM at the node level. This functionality can be achieved by VNF per se or by EM of such VNF. Moreover, the ANM paradigm should be used for the entire slice.

- The management should be implanted as an external function to a slice function (as defined in ETSI MANO framework), but it should have its own management platform that is a part of the slice, therefore being a composition of VNFs.
- The ETSI MANO framework is a part of the overall management and plays a slave role to the OSS/BSS part. The NFVO shall expose all interfaces and operations to the OSS/BSS part, as it is defined in ETSI IFA013 [12] :

- NSD interface (for NSD/PNF: onboard, enable, disable, update, delete, query, subscribe, notify);
- NS Lifecycle interface (create ID, instantiate, scale, update, query, terminate, delete ID, heal, get operation status);
- NS Lifecycle Change Notification interface (subscribe, notify);
- NS Performance Management (PM) interface (create PM job, delete PM jobs, subscribe, notify, query PM job, create threshold, delete thresholds, query threshold);
- NS Fault Management interface (subscribe, notify, get alarm list);
- VNF Package management interface (onboard, enable, disable, delete, query, fetch, subscribe to new notifications. notify of on-boarding/changes of VNF packages).

- In order to implement efficient and scalable management of slices, the framework can be modified but such changes should be minimized. Such modifications should be a subject of standardization. In the presented concept, it is possible to use MANO ‘as it is’, however some modifications of MANO could provide important benefits of scalability.

6.1.1. Single NFVO domain management architecture

The overall concept of scalable management and orchestration in case of single NFVO domain is presented in Figure 12; unless otherwise noted, the approach is applied to both, the Dedicated Slices and to the Common ones.

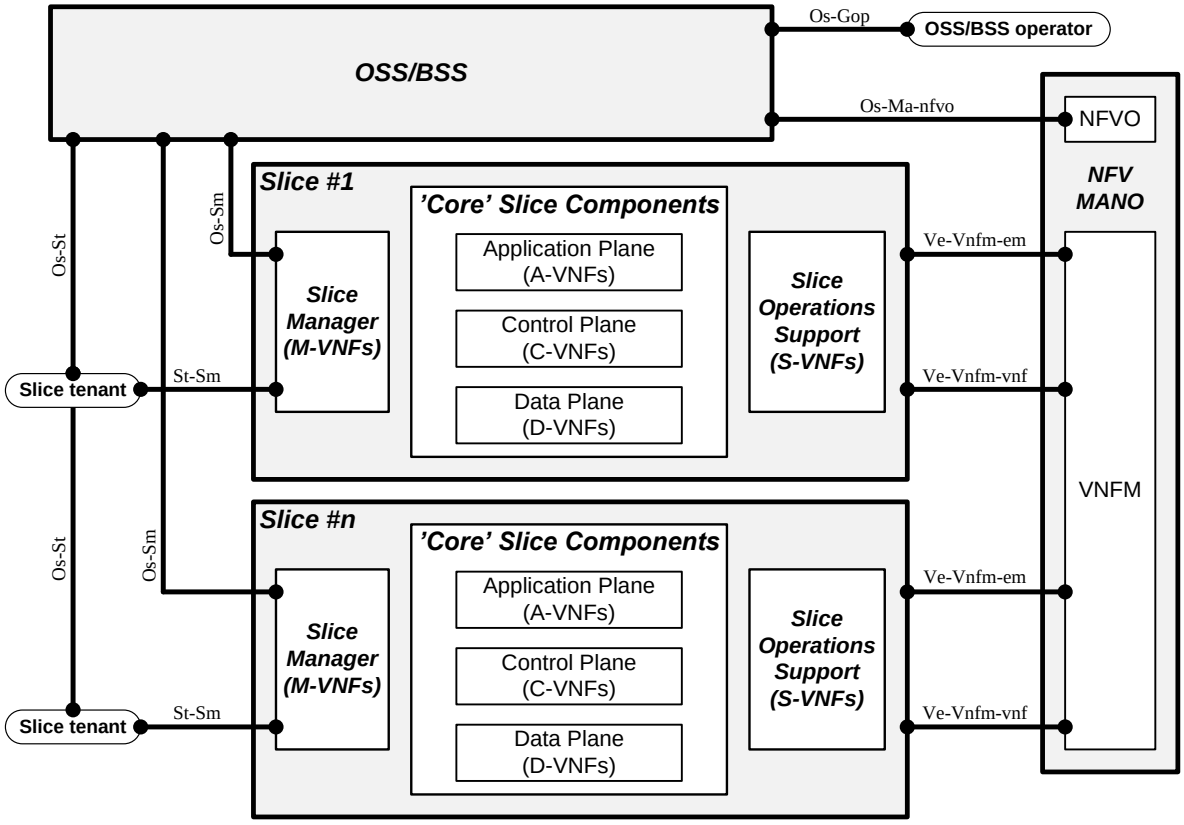

Figure 12 – The overall 5G!Pagoda management and orchestration architecture – a single NFVO domain case

The presented figure is compliant with 5G!Pagoda architecture described in D2.3, but it provides more details about management of slices and their orchestration. The figure follows the ETSI MANO approach with some minor changes that do not require a change of ETSI MANO.

The Slice Tenant can request a creation of slice by NFVO-domain specific OSS/BSS via the Os-St reference point. In response to this request, the NFVO-domain specific OSS/BSS sends appropriate requests to NFVO that is responsible for proper resource allocation, VNF placement, and chaining as well as VNF initial configuration (done by VNFM). All the operations are performed by ETSI MANO compliant orchestrator. The Slice Blueprint used for slice creation consists not only of 'the core' network instance functions (grouped into Application, Control, and Data planes), but also Slice Operation Support (SOS) functional entities and Slice Management entities. After creation of a slice, the Slice Tenant can use the interface at St-Sm reference point for slice management.

More detailed slice management architecture is presented in Figure 13. In order to describe the 5G!Pagoda approach to slice management, more details about EEM, Slice Management (SM) and NFVO-domain specific OSS/BSS is provided in subsequent sections.

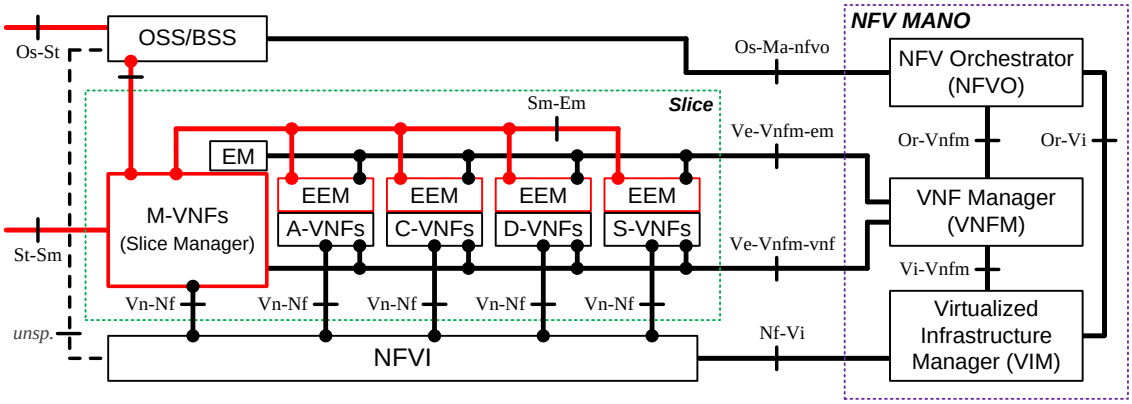

Figure 13 – Slice management concept is shown in a single slice example. The red color shows 5G!Pagoda specific management related interfaces and functions.

6.1.1.1 Embedded Element Manager

The Embedded Element Manager (EEM) plays a slave role in relation to M-VNF(s). Its scope of activity is limited to a single VNF to which it is attached. It is proposed to embed the autonomic behavior of VNF that is attached to the EEM. The internal functionalities of EEM are presented in Figure 14.

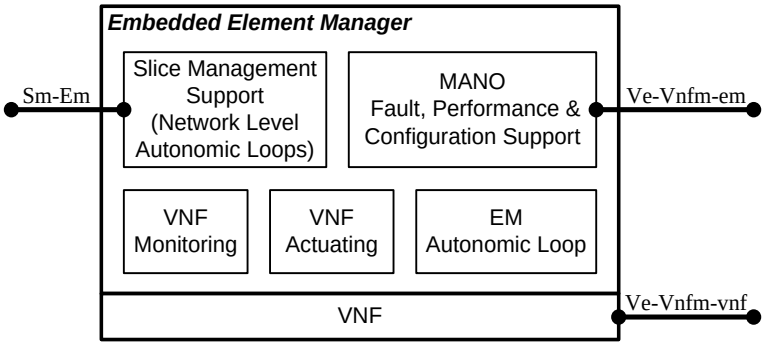

Figure 14 – Internal components of the Embedded Element Manager (EEM)

The functional components of EEM have the following roles:

- MANO Fault, Performance and Configuration Support component is responsible for initial VNF configuration and abstracted reporting of Faults and Performance to VNM (using Ve-Vnfm-vnf reference point) to VNM according to ETSI IFA013 [12] .
- VNF Actuating component is used to change VNF configuration by VNFM, EM Autonomic Loop and Slices Management Support (listed here in increasing priority order).
- VNF Monitoring component is used for monitoring of VNF and provide input: to VNFM for MANO related Fault and Performance reporting, to EM Autonomic Loop and to Slice Management Support. The later reporting can be used for Slice/Network Level Autonomic Loop.
- EM Autonomic Loop is a VNF level autonomic engine that uses local VNF Monitoring and VNF Actuating in order to perform a change of VNF configuration. This component can be used for example for plug-and-play inserting/placement of VNF. The Sm-Em reference point can be also used for information exchange between EEMs.
- Slice Management Support functional component performs multiple roles. It provides an interface to external Autonomic Control Loop(s) based on MAPE monitoring and actuating

functionalities, reports VNF performance and faults related to SM. It can be also used for configuration of VNF by the use of Sm-Em reference point.

It is worth to note that all these operations are VNF and SM specific, therefore no generic implementation of them can be done. The presented approach improves significantly management scalability by taking some – VNF-specific – decisions be EEM and by preprocessing monitoring data within the EEM.

6.1.1.2 Slice Manager

The Slice Manager (SM) internal architecture is presented in Figure 14.

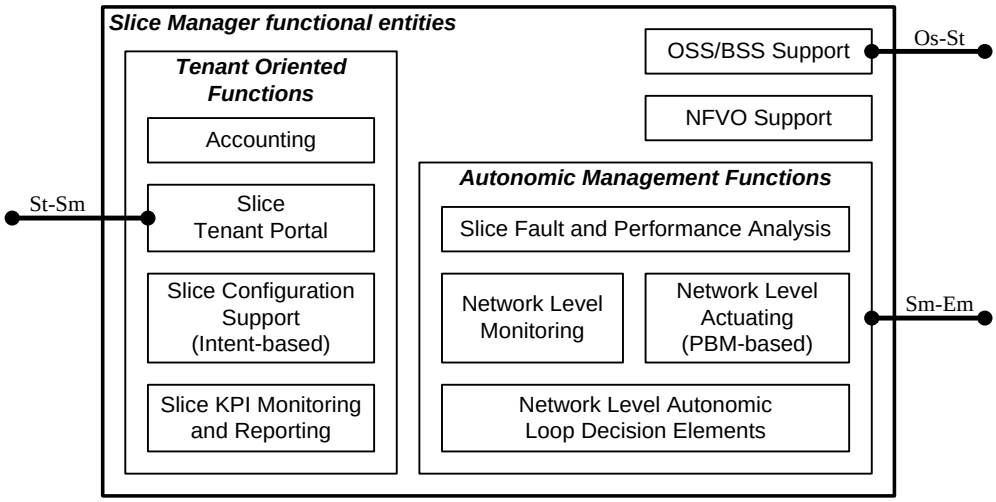

Figure 14 – Slice Manager functional entities

The Slice Manager (SM) is implemented as a part of the network slice, i.e. as a set of VNFs (M-VNFs) responsible for slice management and external management related interactions. Similarly, to EEMs the functionalities of SM are not slice agnostic. There are many implementation possibilities of SM, starting from the classical approach. The Slice Manager is used by slice tenant (operator) in order to monitor slice behavior, change its configuration and provide billing support. It is assumed that this type of management is lightweight and comfortable and most of the management tasks, including most of FCAPS functions, are automated. Embedding the SM, which plays a role of 'slice-wide OSS/BSS', provides inherent scalability and separation of slice management from other slices.

The Slice Manager consists of functional entities that are responsible for autonomic slice management, tenant-oriented operations, and interaction with NFVO-domain specific OSS/BSS and NFVO.

The Tenant Oriented Functions include:

- Accounting component is responsible for the accounting of customers as well as provides slice level accounting with a set of appropriate databases.
- Slice Tenant Portal gives the slice tenants a gateway point for interactions with its slice management functionalities.
- Slice Configuration Support (Intent-based) component is used for changing slice core functions configuration, impacting their behavior, etc.
- Slice KPI Monitoring and Reporting entity is responsible for providing the slice tenant an insight into the slice performance for internal purposes and SLA tracking.

The Autonomic Management Functions consist of the functional entities that implement the MAPE paradigm (real-time feedback loop based management). They include:

- Network Level Monitoring is used for collection and processing (including filtering) of slice network related information. This information is used to implement the autonomic behavior of management operations serves as an input for performance and for fault analysis (proactive as well as reactive one). It can be also used for proactive fault indication to VNFM.
- Network Level Actuating (PBM-based) component is used for changing the network functional entities configurations based on the Network Level Autonomic Loop decision. It is suggested to use Policy-Based Management mechanism to enforce changes.
- Network Level Autonomic Loop Decision Elements are a set of entities that after processing the monitored data and tenant input elaborate decisions regarding network level reconfigurations in order to maximize network instance performance, handle faults, etc. There can be many optimization goals and the implementation of this component is left to the implementation. The function can use EEMs for distributed implementation of some of its functions (cf. monitoring). Such distributed implementation improves network management scalability as well as shortens management decisions execution time.
- Slice Fault and Performance Analysis component is responsible for continuous performance analysis with proactive identification of faults. The functionality includes root-cause analysis. The output of this analysis is used for autonomic decisions, performance, and fault reporting to the tenant. It can be also used for sending appropriate performance and fault-related VNFM actions. Using (via Orchestration Support functional entity of SOS).

The NFVO-domain specific OSS/BSS Support functional component is responsible for interactions between Slice Manager and NFVO-domain specific OSS/BSS. Its functionalities will be described in NFVO-domain specific OSS/BSS dependent part.

The NFVO Support functional entity is an optional entity. Its role lies in indirect interaction with NFVO (via NFVO-domain specific OSS/BSS) in order to provide cross-optimization of SM and NFV related operations.

6.1.2. Single administrative domain OSS

In the preceding subsections, there has been described the components of the scalable management architecture that are a part of aa single NFVO domain. The administrative domain, however, may have several NFVOs and in such case aa single, OSS can be used. The internal architecture of the administrative domain-specific OSS/BSS is presented in Figure 15.

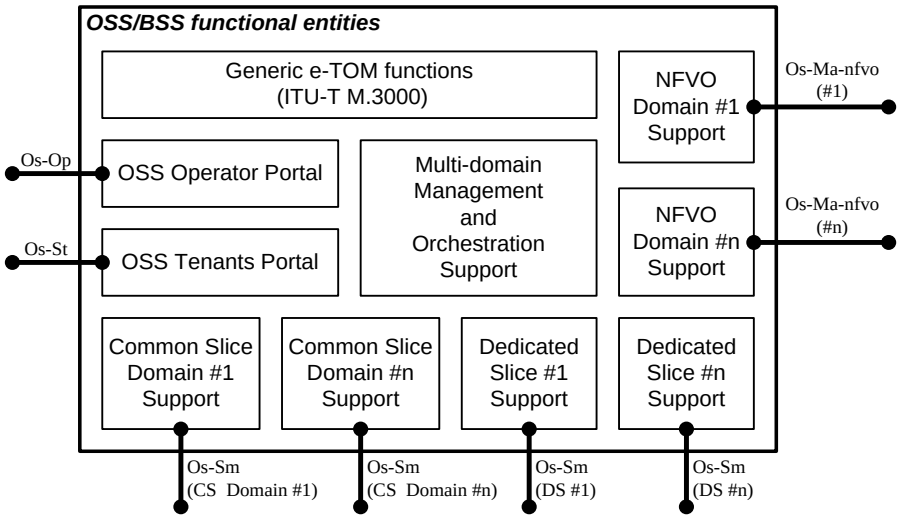

Figure 15 – Internal NFVO-domain specific OSS/BSS architecture

The OSS/BSS has generic management functions (marked as generic e-TOM functions) and slicing specific functional components that include:

- The administrative domain-specific OSS/BSS Operator Portal that is used by the system operator for management purposes.
- The administrative domain OSS Tenants Portal that is used by all tenants for operations related to Dedicated or Common Slice lifecycle management as well as for accessing information regarding slice catalog, historical information about slices and their accounting data.
- Generic eTOM functions – a set of generic network/service management functions as defined by ITU-T in M.3000. This component has some slice related functional blocks (CRM, etc.)
- NFVO Support functional component is the NVO counterpart on the OSS side and it plays a master role in communication with NFVO. This part handles all MANO specific operations, provides Network Services repositories, etc. There is a single NFVO support entity per administrative or technological domain. The interaction between NFVO and NFVO-domain specific OSS/BSS uses the ETSI MANO Os-Ma-nvfo reference point and its interfaces without any modifications.
- Multiple Dedicated Slice Support entities are used for interactions between slice embedded Slice Managers and NFVO-domain specific BSS/OSS. The interaction is provided via interfaces at Os-Sm reference point. Each Dedicated Slice has its own entity. The interactions are used for passing some accounting related data, enabling triggering of NFVO decisions by SM (if allowed) and passing NFVO related information to SM in order to perform of cross-optimization of NFVO and SM decisions.
- Multiple Common Slice Support functional entities perform a similar role to Dedicated Slice Support functional entities. They have an additional function that is related to the handling of Dedicated Slices that are attached to each Common Slice.
- The Multi-Domain Management and Orchestration Support (MDMOS) functional entity performs a key role in slice management and orchestration:

- It implements a dialogue between the slice tenant (via administrative domain-specific Tenants Portal) that leads to a creation or termination of a slice.
- According to tenant requirements it orchestrates and initializes a slice as a single-domain (with the involvement of a single NFVO) or multiple-domains slice via involvement of two or more NVFOs and providing appropriate slice stitching mechanisms.
- In case of multi-domain based slices, the MDMOS provides integration of their SMs. In order to provide the end-to-end management, it can give to one of SM a master role or it can take the master role.

The involvement of MDMOS in case of multi-domain orchestration is presented in Figure 16. In the figure, internal functional components of MDMSO are also presented, namely Multi-Domain Slice Configurator and Multi-Domain Orchestrator.

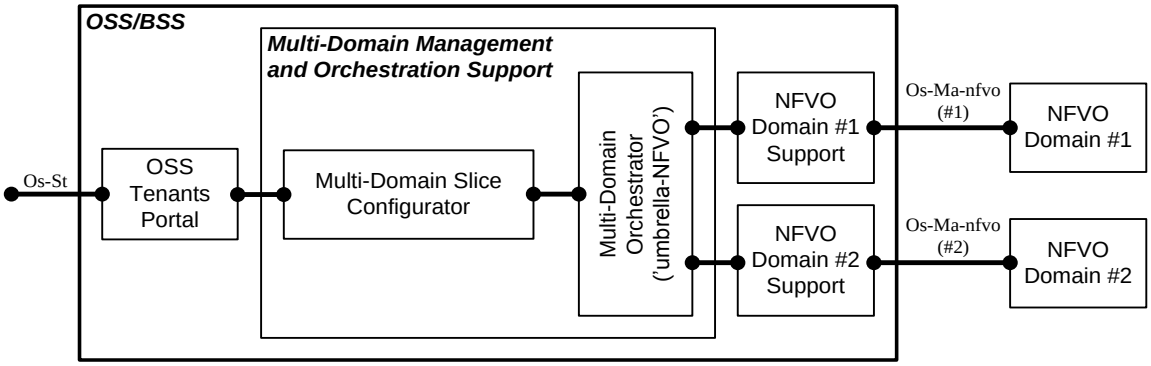

Figure 16 – MDMOS role in multi-domain orchestration (only selected components of NFVO-domain specific OSS/BSS are shown in the picture)

The Multi-Domain Slice Configurator (MDSC) is involved in a dialog with a tenant, which requests slice creation. This dialog leads to the selection of appropriate the Network Service template (blueprint) and its flavor [10]. It also analyses, if the requested service covers single or multiple domains. In the latter case, it adds a component that is responsible for multi-domain slice management (part of SM) and adds or removes some components responsible for inter-domain operations support (part of SOS). In case of management, this operation leads to a selection of master SM which is used for interaction with the tenant via Slice tenant portal. In order to achieve its goal, the MDSC has rights to modify the SOS and SM templates of involved domain slice blueprints.

The Multi-Domain Slice Orchestrator (MDSO) takes also slice blueprints from MDSC and coordinates their deployment in a single administrative domain. During slice runtime, it keeps monitoring of the end-to-end slice and coordinates the end-to-end slice reconfiguration. It performs a role of 'umbrella-NFVO' as defined by ETSI [13].

6.2. Implementation example of scalable monitoring of slices

This section shows a way, how scalable monitoring as a part of management and orchestration can be implemented. In fact, it shows how the architecture design contributes to the monitoring scalability.

The 5G!Pagoda management concept follows the general ITU split of the management architecture (see M.3000 [2]) into business, service, network and element layers. As slicing itself is referred to establishing of separated networks on a shared infrastructure belonging to numerous owners, it does not disrupt the cited

ITU model. It makes it, however, harder – multiple networks (i.e. slices) have to be managed at the same time.

One of the most important problems of network management is proper and economical monitoring of network and services. Typically, network/service monitoring raises the most important problems linked with the scalability of network management. In general, monitoring may have multiple goals, like network/service fault detection, KPI reporting, SLA monitoring, security attacks detection etc. In fact, the monitoring may be considered as a service that can be exploited for different purposes.

As it has been already pointed out, a scalable management approach lies in the distribution of management functions, e.g. implementing part of them as a part of each slice and using the autonomic network management paradigm. These features impact the way in which the monitoring data are transferred and processed. In the subsequent subsections, it will be described in more details how the architecture defined in the section 6.1.1 contributes to the overall scalability of monitoring in 5G!Pagoda. The described monitoring approach is focused on the management part of the architecture and does not deal with the scalability of monitoring for NFV orchestration.

6.2.1. Single NFVO-domain-specific monitoring architecture

The slice monitoring architecture follows the management and orchestration concept presented in the section 6.1.1. The key feature of the architecture is the embedded management intelligence of each node that can be exploited for node self-management, slice self-management capabilities and finally coordination of management and orchestration functions in the administrative domain-specific OSS/BSS. This split defines the way how the monitoring data are processed and stored.

In this section a monitoring of the multi-NFVO domain environment is concerned but, the initial focus is on a single NFVO domain – the multi-domain monitoring information exchange is always very limited; therefore, it does not raise the scalability issues related to management. In fact, the multi-domain monitoring concerns only NFVO-domain specific OSS/BSS functional components of the management architecture.

The overall concept of slice monitoring is presented in Figure 17. Please note that this picture shows the concept in a simplified way and some functional components of the architecture are intentionally removed.

The monitoring operations have multiple goals. Monitoring should be programmable, enabling the measurement reports to be composed according to the needs of the monitoring information consumer. The most intensive monitoring happens at the network element level. The EEM has embedded control loop as a part of the node and the EEM is also a consumer of the monitoring information. This fastest monitoring is therefore performed locally. It is also used for producing information that is consumed by VNFM. A slower monitoring information flow is produced by EEM for the Slice Manager (SM). It has to be noted that SM can be implemented in a distributed way (many VNFs can be used for SM implementation) therefore, the collection of monitoring data can be hierarchically distributed. SM uses the information for multiple purposes including network level autonomic operations. It interacts with NFVO-domain specific OSS/BSS, sending there mostly information related to slice KPI, events, accounting and monitoring data necessary for proper multi-slice management and orchestration. Part of this information is by NFVO domain specific OSS/BSS ‘internal’ purposes (that includes embedded control loop operations) and part is used by slice tenant and the NFVO-domain specific OSS/BSS operator. The Slice Manager uses the interface at St-Sm reference point to provide slice related information to slice tenant and the interface at Os-Sm reference point to NFVO-domain specific OSS/BSS. The NFVO-domain specific OSS/BSS has also information about consumption and status of slice resources (obtained from NFVO).

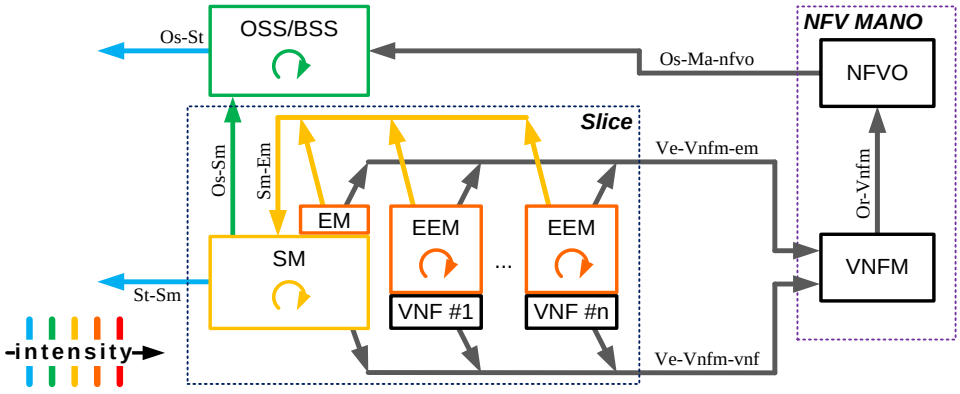

Figure 17 – Simplified schema of monitoring flows

In subsequent subsections, more details about monitoring functionalities of EEM, Slice Manager and NFVO-domain specific OSS is provided. The monitoring information is always specific to VNF, SM and NFVO-domain specific OSS/BSS type and implementation, therefore the provided description can be treated as guidance only on how the overall monitoring in a multi-slice environment can be implemented.

6.2.1.1 Embedded Element Manager monitoring functions

From the monitoring point of view, the EEM plays a role of a dedicated monitoring agent tightly coupled with VNF (one agent per each core VNF). The internal functionalities of EEM related to monitoring are presented in Figure 18.

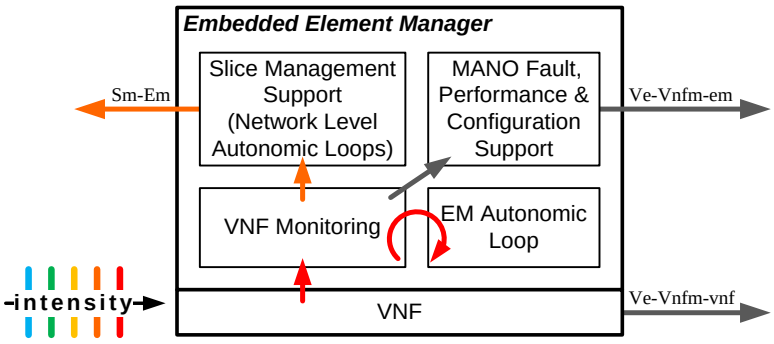

Figure 18 – Monitoring-related part of EEM

The functional components of EEM have the following roles in monitoring:

- VNF Monitoring is the core component of monitoring functionality within EEM and is used for all interactions with VNF related to fault and performance monitoring (reception of notifications, VNF polling, and VNF performance jobs management) and threshold events management. Its output feeds the EM Autonomic Loop functional block and M-VNF (via Slice Management Support functional block and interfaces at Sm-Em reference point). It may also interact with VNFM via MANO Fault, Performance & Configuration Support functional component.
- EM Autonomic Loop function is the largest consumer of data produced by VNF Monitoring component. It uses these data for local VNF control loop (via VNF Actuating functionality, not shown in the figure).

- Slice Management Support functional component acts as a proxy for interactions between VNF Monitoring functional component and its master SM (M-VNF), reporting VNF performance and faults to SM. It can be also used for controlling the monitoring part of EEM.
- MANO Fault, Performance and Configuration Support component is responsible for interaction of EEM monitoring agent with VNFM (using interfaces at Ve-Vnfm-em reference point). It supports fault management, performance management and Indication interfaces over Ve-Vnfm-(vnf/em) reference points according to NFV-IFA 008 [14] :

- The Fault Management interface exposed by VNFM allows VNFs or their E(E)Ms to receive notifications about fault-related events that affect the virtualized resources of specific VNFs.
- The Performance Management interface exposed by VNFM allows VNFs or their E(E)Ms to receive performance information about virtualized resources of these specific VNFs. Additionally, E(E)Ms are able to manage performance related jobs and thresholds.
- Indication interface exposed by VNF/E(E)M allows VNFM to get a feedback from VNFs or their E(E)Ms about the experience of the performance of virtual resources provided to these specific VNFs.

6.2.1.2 Slice Manager monitoring functions

In terms of monitoring, the SM (M-VNF) plays a key role. It is used by the slice tenant (operator, vertical) in order to monitor performance and behavior of a slice and to provide billing support (KPIs, SLA measurements). The internal composition of SM functional blocks related to monitoring is presented in Figure 19.

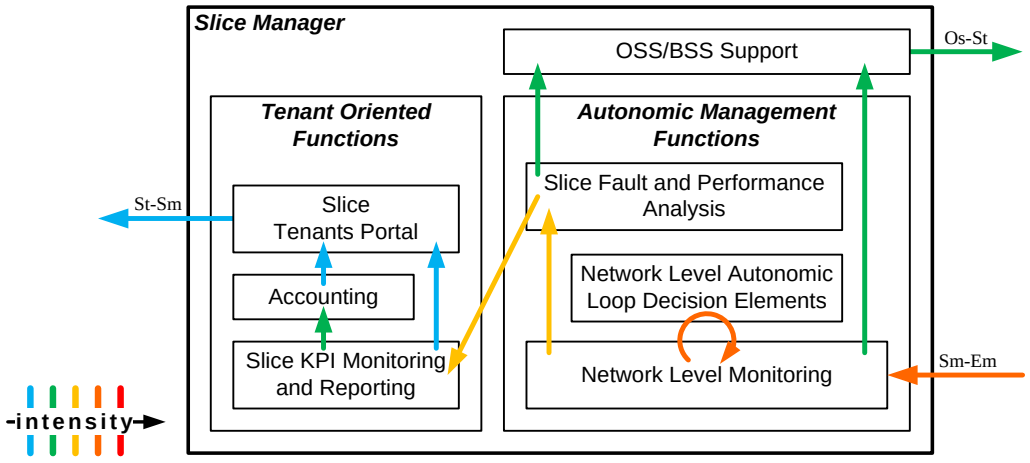

Figure 19 – Slice Manager functional components related to monitoring

The Autonomic Management Functions part related to monitoring consists of the following functional components:

- Network Level Monitoring is used for collection and processing (including filtering) of information related to the sliced network. This information feeds the autonomic behavior of network level management operations, is used for the analysis of slice performance and for faults detection. The information is obtained from (E)EMs.
- Slice Fault and Performance Analysis component is responsible for continuous performance analysis with proactive identification of faults or load related trends. The functionality includes root-cause analysis. The output of this analysis is used for performance and faults reporting to tenant or orchestrator operator and may be also used for autonomic decisions

at the network level. It can be also used for initiating proactive actions related to performance or fault issues of VNF.

- Network Level Autonomic Loop Decision Elements component is a consumer of monitoring data as well as fault and performance information in order to make the autonomous control at the network level. It uses basically the raw, 'high speed' data provided by the Network Level Monitoring component and optionally the preprocessed monitoring information (e.g. correlated events, predicted trends) from Slice Fault and Performance Analysis function.

The Tenant Oriented Functions related to monitoring include:

- Accounting component responsible for billing of tenants based on slice behavior and fulfillment of performance KPIs according to SLA.
- Slice Tenant Portal giving a tenant an entry point for interactions with its slice management functionalities associated with network monitoring. It is assumed that this management is lightweight, comfortable and automated; therefore, no detailed access to raw data of monitoring is necessary.
- Slice KPI Monitoring and Reporting entity is responsible for providing the slice tenant the insight into the slice performance for internal purposes and SLA tracking.

The slice faults and performance related data (including slice KPIs and accounting data) should be also accessible for the orchestrator operator at the NFVO-domain specific OSS/BSS entity. In this case, they should be fetched via the interfaces at Os-Sm reference point and the NFVO-domain specific OSS/BSS Support functional component of SM (not shown in the figure for the overall picture simplicity).

6.2.1.3 The administrative domain-specific OSS/BSS monitoring functions

Hitherto the considerations of monitoring architecture were focused on a case of a single slice and single NFVO domain. For multi-slice and multi-domain case, the NFVO domain specific OSS/BSS entity acts as an 'umbrella OSS/BSS', exposing the global view to the tenant via Os-St reference point. In Figure 20 the flows of monitoring data within the administrative domain-specific OSS/BSS are presented.

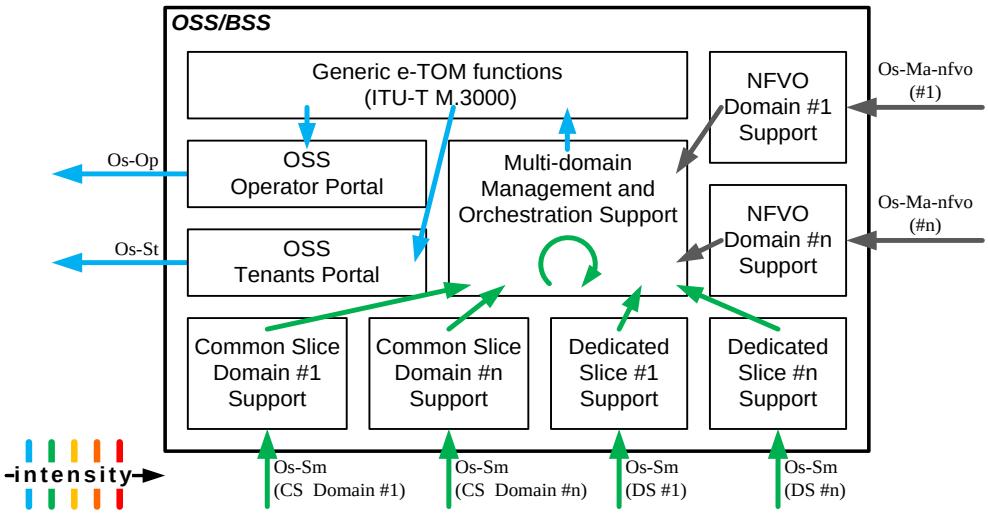

Figure 20 – Monitoring flows in NFVO-domain specific OSS/BSS

The Common Slice Domain Support functional entity collects and stores locally information related to the management of its Common Slice. This information, as well as similar information related to the Dedicated Slices (stored in the Dedicated Slice Support functional entity), is used by the Multi-Domain Management

and Orchestration for end-to-end optimization of multi-domain slices. The Multi-Domain Management and Orchestration Support component is responsible for monitoring of the end-to-end slice during its lifetime and also for coordination of the slice reconfiguration using multiple NFVOs. The NFVO Domain Support entities keep the monitoring information that is obtained from NFVOs as well as commands that were sent to NFVO. The Multi-Domain Management and Orchestration Support functional component has embedded autonomic management mechanisms that are mostly used for autonomic handling of multi-domain management related issues. It obtains monitoring information from NFVOs and from Service Managers. That enables this entity to make cross-optimization of management and orchestration. The KPIs/reports and accounting information about the end-to-end slices is also obtained from this component and will be finally exposed to orchestrator operator and slice tenant via the interfaces at Os-Op and the Os-St reference points respectively.

6.3. Scalable Redundancy Management

6.3.1. Requirements for Scalable Redundancy Management

In the case of system failure, there are two types of redundant schemes, which are inherently utilized in the communication system:

- restarting the sessions for the communication service, and
- succeeding the session status to the recovered functional components or a system

Although these two schemes have individual procedures in the system recovery, they must memorize the sessions. Individual schemes are described in the following subsections.

It should be noted that the recovery procedure will basically utilize surplus resources where the recovered functional components and/or system are built.

6.3.2. Restarting Sessions

6.3.2.1 Issues in Restarting Sessions

When restarting sessions, all the session users should be notified of the restart event and restarting sessions. The primal (basic) procedure in this scheme takes the following steps after the service system recognizes the failures.

(1) The service system estimates the failed component of the service system and the required resources for the recovery.

(2) The service system sets up the functional components or service system corresponding to the failed ones with the estimated resources.

(3) The user terminals or applications recognize that their sessions have failed, and restarts their sessions. This procedure can be originated by the user terminals or applications by themselves, and in this case, this step starts while steps (1) and (2) are running, and it continues to fail until step (2) competes. Or the service system may be able to notify the user terminals and applications if the system failure does not affect the notification procedure, and user terminals and applications will restart their sessions.

From this procedure, the requirements of the orchestrator and service system can be:

- The services system preliminarily manages the resources and individual functional components for step 1,
- The service system needs to receive the resources for setting up the functional component again for step 1,
- The services system maintains the system configuration to restore the failed component for step 2,
- The services system notifies user terminals and applications to restart their sessions after the system recovers,
- The service system should have a robust recovery procedure, although the user terminals and applications continue to restart session while the recovery procedure in the service system runs, as mentioned in step 3, and
- The user terminals and applications should also have a robust procedure to restart a session because the system in recovery may not respond to any operation.

The first three requirements (underlined texts) are likely to be supported by the orchestrator maintaining the environment for the service system. Regarding the first requirement, the orchestrator keeps maintaining a map of the functional components and resources. This is used for the second requirement (setting up the functional component again instead of failed ones). Regarding the third requirement, maintaining configuration may be included in the first requirement, and the configuration may be combined to be stored. The actual configuration should be the network parameters (subnet, IP address, and routing/forwarding configuration) and assignments of CPU, memory, and storage.

In case of a highly-frequent slice-operation (e.g., within an interval of few seconds) as described in Section 3.1, the demand for the service will change frequently and rapidly while the recovery procedure progresses. Therefore, restarting the sessions and establishing additional sessions will run in a mixed manner. (Although there should be session terminations, it is pended and restarting session is made prior because the affected sessions are restarted.)

The key approach for the mixture of rapid demand change and restarting sessions for recovery is the utilization of the redundant functional components. There are same transactions, that is, the session establishments in both cases (rapid demand change and restarting sessions). In the traditional approach, sufficiently large amount of redundant capacity meets this requirement.

However, when there is an insufficient amount of resources for the redundant functional components, it becomes difficult to cope with both rapid demand changes and restarting sessions simultaneously. The following section discusses how to cope with this situation.

6.3.2.2 Solutions in Restarting Sessions

In the case where the system (redundant) capacity is limited, there is an approach that function component of the service system has a huge number of instances with a small amount of resources. In this situation, system failure could limitedly affect the sessions (portions of sessions), and thus restarting session completes quickly.

Figure 21 shows an example of session distribution over many instances of functional components. In a traditional approach, a small number of instances accommodates a large number of sessions (see the upper part of the figure), but the lower part of the figure distributes sessions over many instances of functional components. Hereafter this is referred to as ‘session distribution redundancy’. To satisfy an efficient

accommodation, multiple service systems share the same hardware. In this figure, four service systems are accommodated in the same hardware infrastructure.

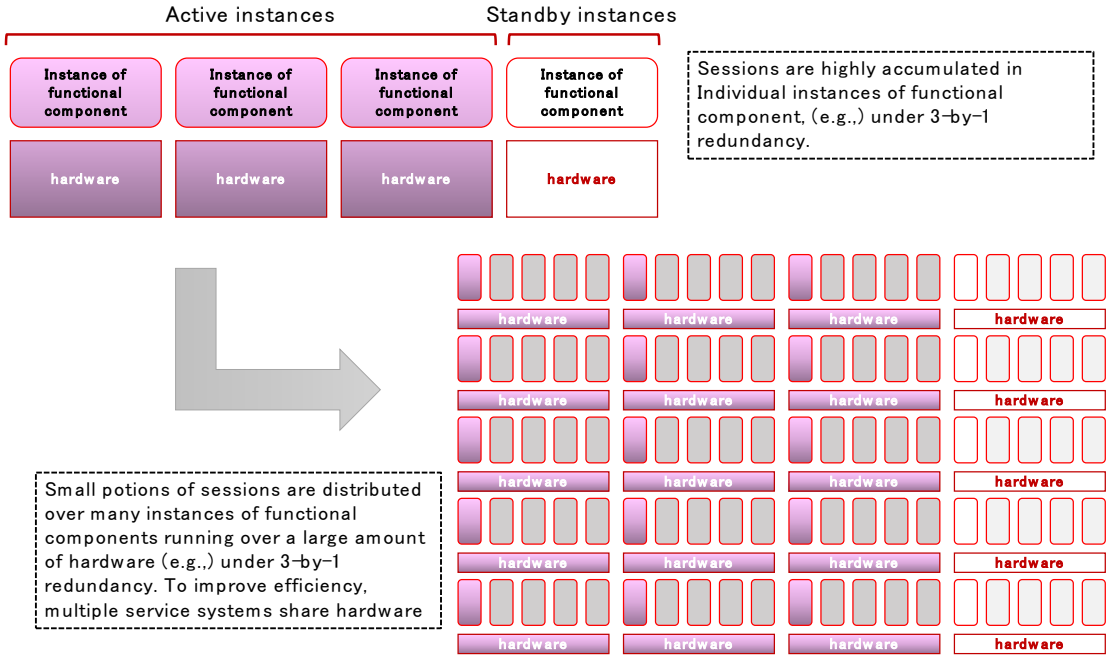

Figure 21 – Distribution of sessions over a large amount of hardware

The session distribution redundancy can take the conventional technique regarding the redundancy mechanism. The number of instances for the individual functional components should be computed with the dynamicity and interval of slice operation, and duration of restarting sessions

Although there should be multiple service systems to make this work efficiently, the 5G mobile communication system would be implemented on the basis of accommodating multiple service systems to its infrastructure.

6.3.3. Succeeding Session Status

When succeeding session status, user terminals and application do not perform any specific procedures for recovery. The primal (basic) procedure in this scheme takes the following steps.

(1) The system itself recognizes the system failure (a portion of the system or functional components).

(2) The system estimates user terminals and applications affected by this failure.

(3) The system recovers the failed functional components and portion of system

(4) The system starts recovery procedure with these recovered entities.

Regarding step (3), in many cases, the user terminal and applications cannot wait for the non-responsiveness during the failed period. Therefore, step (3) should be completed quickly before the user terminals and applications recognize that recovery.

From this procedure, the requirements of service system can be:

- The service system preliminarily manages the resources and individual functional components for step 1,

- The services system has a map of assigned user terminals and applications to the functional components or portion of service system for step 2, and
- The recovery of the service system should quickly recover before the user terminals and applications recognize the recovery,

All the requirements need to be supported by the orchestrator in the succeeding session status. The first and second requirements are the same as restarting sessions.

Regarding the third requirement, the system should prepare the redundant functional components so as to cope with the component failure. Therefore, the service system needs an active-standby mechanism to succeed the session status from the failed (active) functional component. The actual mechanism should include copying any maintenance (update) session, which is in between the active, and standby session while the normal operation of the service system (before the failure occurs).

In terms of the highly-frequent slice operations, the failure of functional component affects nothing on the slice operation. To make a quick switch-over between active and standby instance, the functional components of service system maintain consistency of individual session states, and this can be achieved by the conventional technique. To gain and keep the above-mentioned redundancy level, multiple standby instances are employed in the commodity service systems.

6.4. Management of Edge Computing

6.4.1. Introduction

Edge computing places VNFs at the network edge, which can reduce latency between the UE and the VNF and allow more efficient bandwidth utilization in the core network. Moreover, edge computing may provide applications with location awareness and information from the radio network. Several use cases can benefit from edge computing through offloading of processing from devices, or from moving cloud services closer to the user from the current location in the centralized cloud. Typical services include IoT, vehicle communication, industrial control, and media distribution.

While the focus of NVF is on virtualizing network functions of the mobile network, edge computing primarily enables applications to be placed at the network edge. Many of the applications may be provided by a third party. In this section, NFV refers to both network functions and application services. The platform can be dedicated to edge computing or shared with other network functions or applications. For cost reasons, both applications and NFVs need to be run on the same platform according to ETSI Whitepaper [15] . This creates further challenges in terms of orchestration, management, and security. Edge computing will use (as much as possible) the NFV management and orchestration entities and interfaces.

Edge computing nodes can be deployed at the eNodeB, at the RNC, at a multi-RAT cell aggregation site (e.g. in enterprises, shopping malls, stadiums, hospitals), or at aggregation points at the edge of the core network. As nodes can be deployed at several layers of different distance to the UE this forms a hierarchical computing platform.

6.4.2. Orchestration of edge computing

Edge computing has a profound impact on orchestration and scaling. It introduces a need to dynamically deploy and remove instances of NFVs depending on UE locations. In centralized computing, it is typically sufficient to run a single copy of a VNF, which can be scaled up or scaled out depending on the load. Scaling

can be done to arbitrary locations in the data center, and the choice of location is left to the VIM. On the contrary, a VNF that is allocated to the edge requires more precise placement. An edge VNF instance serves a particular set of UEs, i.e. those directly under its location. Each UE connected to the slice should be assigned to an instance of the edge VNF. For instance, when base stations operate as edge nodes, each base station with a UE in the given slice should have a running VNF instance. If the processing capabilities are not at the base station level but higher up in the hierarchy, a single node can serve the UEs of a set of base stations, and should run the instance of the edge VNF if there are UEs served by any of the covered base stations. Consequently, scaling of edge VNFs refers to scaling according to the number of UEs as well as to the locations of the UEs to be served by the VNF. The slice is thus extended toward the edge in the locations of the UEs.

Edge computing allows splitting a centralized NFV instance into smaller NFV instances at each edge node. Not all edge nodes may have this NFV instance; rather the coverage may be reduced to only nodes with UEs requiring this NFV. Moreover, a centralized instance of the NFV may complement the distributed ones, e.g. in order to cover edge nodes without virtualization capabilities, edge nodes without available capacity or edge nodes with a low number of UEs requiring the service of the NFV.

Typically, several services can be linked to each other in a form of a service function chain, where the data plane traverses several VNFs in a sequence. Some of these services may be provided at the network edge. Orchestration should ensure that the sequence traverses the edge-core axis consistently. Once the traffic has been forwarded to the core, the traffic should not be passed back to the edge. Not only would this cause excess bandwidth use, but also remove all benefits of edge computing, including low latency and bandwidth reduction through local computation. In hierarchical edge computing scenarios, service function chains take the form of a tree, where the first level is specific for individual base stations and combining into common VNFs at a higher level.

The slice blueprint (e.g. in TOSCA format) defines the VNFs and their relationships. It gives the instructions about the location and scaling on a high level. It does not specify the precise locations of the VNF. Rather we propose to attach an edge priority attribute to the blueprint. The edge priority indicates how important it is for the VNF to be deployed at the edge and it may propose the VNF to be located at the edge. The final decision depends on the edge orchestrator implementation, the available resources and the need to aggregate UEs to be served by a feasible number of VNF instances. The VNF may not be placed at all edge nodes, e.g. for capacity reasons, and a complementing instance may be placed centrally to compensate for the edge nodes without the VNF.

When a UE joins a slice, whose blueprint defines the use of an edge VNF, the scaling function must check if there already is a running edge VNF instance in the edge in the location of the new UE. In that case, the user is assigned to the existing instance. On the other hand, if no such instance exists, the orchestrator must determine the location to place a VNF instance serving the new UE. An edge placement algorithm has been presented in Deliverable D3.1.

If the last UE served by the edge VNF leaves the network, the edge VNF can be removed. The removal may not be performed immediately, especially if there is a high probability that a new UE will need an edge VNF at the same node within a reasonable time. Therefore, the removal can be initiated after a period when no UEs has been utilizing the VNF. Another approach is to do the removal in the form of “garbage cleaning” that may be performed periodically as well as triggered on-demand by the need to reclaim resources at the edge nodes. The timers for removal need to be adjusted according to the overhead of instantiation, e.g. long in the case of VMs but short in the case of containers.

Mobility affects similar changes in the VNF placement as UE addition and removal. When the UE moves between areas served by different VNFs, a new VNF instance may have to be placed in the new area if one

is not yet existing. Similarly, when the last UE leaves an area, then the edge VNF of that area may become redundant and scheduled for removal.

6.4.3. Edge orchestration architecture

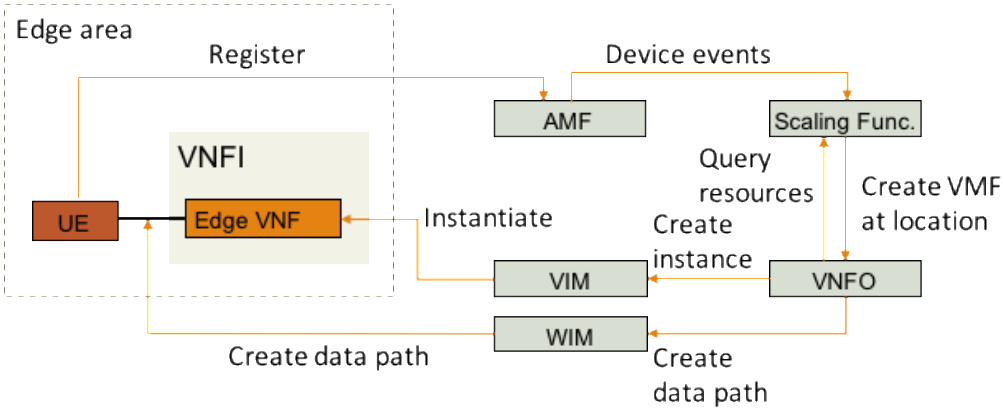

Figure 22 – Components involved in edge orchestration

The architecture for edge computing is illustrated in Figure 22. The placement of edge VNFs is managed by a scaling function that is part of the intra-slice management. Because edge VNF instances are located based on UE location, edge computing requires the scaling function to interact with a node providing user location information such as the Mobility Management Entity (MME) in 4G or the Access and Management Function (AMF) in 5G. The scaling function uses the services of the orchestrator through the orchestrator's intra-slice API to create new instances of a VNF at the desired location. Moreover, a mechanism is needed to assign UEs to a particular VNF instance (or set of instances). This implies setting up the data plane to route the user traffic to the edge VNFs. This can be accomplished via an SDN controller or WIM.

Here, we focus on the changes that edge computing brings to the orchestration in section 6.1. The architecture involves the following components:

- User Equipment (UE): The UE can be part of one or several network slices. Its location is registered to the AMF.
- Core Access and Management Function (AMF): The AMF provides registration, reachability, mobility management and connection management in the 5G network and corresponds to the MME of LTE.
- VNF Orchestrator (VNFO): The VNFO provides via an API service to the slice for creating and removing instances of a particular VNF at a particular location. It also provides services to the slice for obtaining performance and load information as well as the free resources available. For the scalability reason, the VNFO can be distributed.
- NFV Infrastructure (NFVI): Edge computing is based on the same kind of NFVI as the core network. However, because of the nature of edge VNFs (serving a lower number of users) and the smaller available capacity at edge nodes, uni-kernels and containers provide a plausible alternative instead of VMs. NFVI can reside at multiple layers from the edge, ranging from the base station to regional data centers.

- Virtual Infrastructure Manager (VIM): The edge nodes may be served by the same VIM as the core. Alternatively, the network can be split into several regions, each with its own VIM. This may be feasible if edge nodes are using a different virtualization technology (e.g. containers instead of virtual machines). At the extreme, each edge processing node provides a separate VIM.
- Scaling Function: The Scaling Function is a slice specific function managing the scaling of the VNFs in the slice. In the edge computing case, it scales a particular VNF to multiple locations. This is implemented with a common Scaling Function across slices, which can be part of the VNFO or added as an extension module to the VNFO. Another approach is to include a slice-specific Scaling Function as a part of the slice blueprint. In that case, it can be integrated as a part of the Slice Manager.
- Data plane controller: The task of the data plane controller is to set up the data plane path from the UE via the VNFs toward the services or toward the external connectivity. It has the responsibility for creating chains of VNFs for the particular user. The data plane controller can be implemented as part of WIM. The implementation is in practical cases an SDN controller.

We can separate from an architecture, where scaling to edge nodes are performed with functions within the slice, or an architecture with functions common to all slices.

6.4.4. Edge orchestration process

The edge orchestration process is illustrated in Figure 22. The UE registers to the AMF (or MME). The scaling function subscribes to information about new and leaving UEs, or to major changes in the number of UEs under a base station. The scaling function determines, using its algorithm, whether a new edge VNF should be deployed at the location near the new UE. If a new VNF is deployed, the API of the VNFO is utilized to query about the resources at the desired location and to create a new VNF. In all cases (reusing an existing VNF or deploying a new one) the data plane controller is informed about the UE and the VNF in order to connect them. The data plane controller sets up a data path from the UE to the new VNF(s).

6.4.5. Network monitoring

In order for edge orchestration to operate efficiently and to be able to fully utilize the available capacity without over-allocation, it is important to collect performance data from the processing nodes. Since the edge computing nodes are significantly more constrained than the central nodes, and because the workload is subject to fluctuations due to mobility, the ability to collect data in real-time is crucial. Performance data includes the CPU load, memory consumption, power consumption, disk space and bandwidth utilization both for data and control plane. The performance data directly affects the orchestration and the resource allocation.

6.4.6. Management of edge NFVs

Management of the high number of replicated NFVs may be challenging, both in terms of complexity and in terms of scalability. One solution is to represent the distributed instance of the NFV to the management system or OSS/BSS as a single abstract “virtual” NFV. The virtual NFV collects metering information from all the NFVs and complements it with information about the distribution. Configuration data sent to the virtual NFV is distributed to the NFV instances at the edge. This solution also supports the dynamicity in NFV

instances being created and destroyed, whereas the management system does not need to know the identity and address of each instance. When a new instance is deployed, it obtains its configuration from the virtual NFV.

6.4.7. Managing software updates

One network management operation introduced with NFV is the software updates. Updates to NFV software should be applied to all instances. For edge computing, the number of instances to update may be high. Therefore, to prevent the update load effect on bandwidth, attention should be paid to timing updates and to using caches. To minimize downtime, the load of edge VNFs being updated or redeployed can be moved to a centralized instance of the VNF temporarily. The update process can be linked to the scaling procedure. To minimize the risks in updating and testing new deployments, the updates can be applied to a few edge nodes before rolling it out on a network-wide scale.

6.5. Scalability of Appliance and Scalability of VNF

6.5.1. ICN/CDN specific VNFs

In the CDN as a Service use case, the platform consists mainly of 4 VNFs; virtual caching, virtual transcoding, virtual streaming and also a CDN-slice-specific virtual Management Function called Coordinator for the management of the slice resources.

The Virtual Management Function provides and manages a CDN-slice on top of multiple administrative domains and establishes a secure connection between NFVs belonging to the same slice running over multiple domains. Then, when the CDN-Slice deployment is successful, exchange connectivity parameters between the different NFVs to be able to stitch together the slice and communicate securely among them. As the resources are virtualized, the slices can receive dynamic resources during their runtime as well as different resources placement [17] , through this the infrastructure becoming flexible and available in a different combination on demand, and we can ensure a dynamic scaling up and down of virtual resources.

All these VNFs represent the Edge servers of a virtual CDN-slice, however, the Virtual Management Function is the brain of the slice. After Extraction-Transformation-Loading (ETL) process in the VNF level, all the access information is pulled back and stored in the brain in order to understand the end-users behavior regarding the content popularity and VNF hit ratio as well.

Statistics module enables the Coordinator to notify the Orchestrator for a more/less need for resources to ensure the availability and Scaling up and down of a CDN-slice.

The virtual Management Function would be able to regularly learn and analysis the Data in order to make decisions regarding the virtual resources optimization:

Need for more virtual resources: Scaling up the slice by adding new VNFs to lighten the traffic overload or, scale out by migrating the network functions to a bigger and more powerful hosting virtual machine.

Eliminate resource waste: Shut down/Pause non-needed VNFs based on its Hit-Ratio.

Through this, the life-cycle orchestration is able not only to deploy according to the specific resource need configuration of the specific slice but also to adapt to different usage conditions, how the users of the slice behave and to the exceptional network situations for performance optimization and security.

6.5.1.1 Transcoding and Streaming as Virtual Network Functions

Assuming a constantly growing demand for streaming services and transcoding will become compulsory and very challenging. So far, investigations have been confined to satisfy a huge number of users for ensuring the Quality of Experience (QoE).

We aim to ensure the flexibility of our virtual delivery platform that scales up/down and in/out relative to the changing demands of the end-users in order to reduce cost. We presented in IEEE Globecom on December 2017 [18] a new framework for managing the virtual live transcoding and streaming VNFs on top of multiple cloud domains for ensuring the QoE while reducing the cost. In order to develop such a framework, we have done a set of experimental benchmarking of transcoding and streaming VNFs using different flavors (i.e., in terms of CPU and Memory resources).

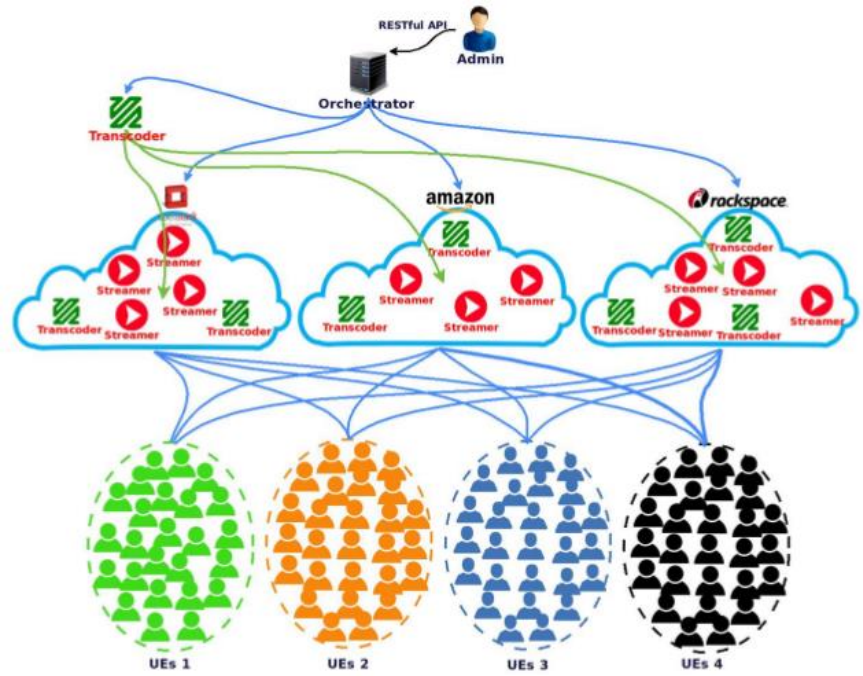

Figure 23 – Single slice across multiple cloud domains

The figure shows the different streamers and transcoders of a single slice across multiple cloud domains. This framework considers a Global Transcoder hosted in a very powerful physical machine intervenes, if needed, to help during the load balancing in case the slice knows a growing overload of user requests.

6.5.1.2 Caching as a Virtual Network Function

In order to lighten the overload on the CDN-slice VNFs, we aim to combine both ICN network with CDN as a Service slice. ICN node combines both functionalities of routing and switching and also has the ability to store contents. Every node has a buffer memory serves as a cache content. Basically, the content could be in-cached in the network. If the requested content is in the content store, the node can immediately send the data without generating further requests to the original content provider, which is the CDN-Cache server. That ensures many benefits for reducing overall bandwidth usage and latency to improve the Quality of Service (QoS).

As illustrated in the figure below, instead of requesting the CDN slice for the same content multiple times, the end-user will simply express his interest to the ICN Network, if the requested content is already in-

cached in the network, the content will be forwarded back from virtual ICN router to another to the requester.

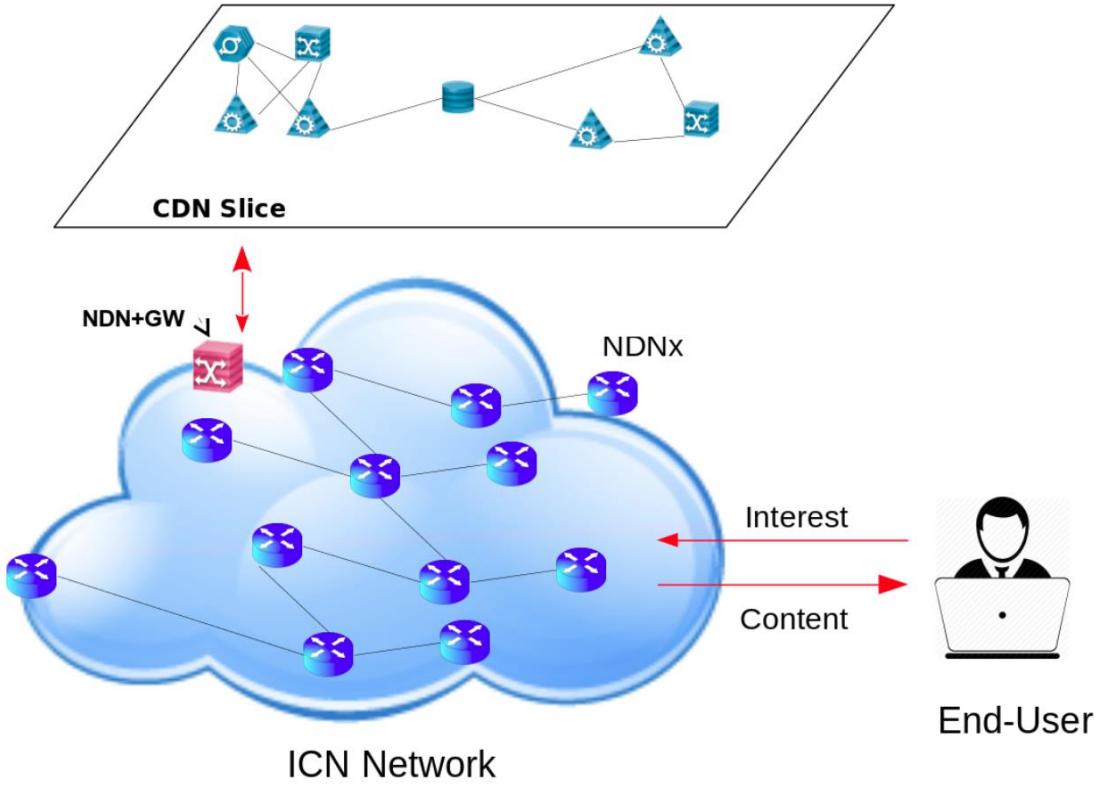

Figure 24 – Simply express

In ICN/CDN combined contents delivery service scenario, ICN slice and CDN slice will be created and linked together to provide with an efficient content delivery service. Since the linkage of networks using different protocols is requested, it is necessary to prepare the gateway VNF to be used at the boundary of a slice. There it is expected that ICN slice and CDN slice(CNDaaS) have different nature in their lifetime. ICN slice will have long life dealing with a variety of contents, but certain type of CDN slice will be created and terminated more frequently, because this type of CDN will be created based on the specific events that big number of people would like to see such as big sport events and cultural events, and the location of CDN server also changes according to the interest of the region. Because of this, from the viewpoint of ICN slice, the location of the content server will dynamically change in time, and each time the location changed, a new gateway VNF should be added. It is also expected that different CDN slice will be created while another CDN slice exists, then the new gateway VNF should be added. This requires the scalability of gateway VNF in number.

Another aspect is that when the new CDN slice is created, the variety of contents served by ICN will increase, and the user access will increase accordingly. To maintain the similar service quality, the resource assigned to the VNF is expected to scale. All the three basic node functions, content cache (storage) capacity is expected to increase, and the routing table in FIB (Forward Information Base) and the PIT (Pending Information Table) size.

As for the RAN part, it is very efficient if the multicast group of the popular content is provided as a slice. In this case, it is requested that the multicast management VNF is scalable in number.

6.5.2. RAN VNFs

To ensure slice scalability, efficient management procedures shall rely on pertinent monitoring information on the network appliance. Indeed, most of the VNF constituting the network slice may run on top of VM or containers. Therefore, many VIM, like OpenStack using ceilometer, allow gathering information on the system environment and resources that are actually assigned to a VNF, such CPU consumption, memory usage, etc. However, the management procedures require other information, which is service-related, like the number of connected UEs, average request attach from UE, etc. In the context of 5G!Pagoda, many VNFs are based on OpenAirInterface (OAI). To recall OAI is an open source implementation of 4G eNodeB and Core Network. All the EPC entities (i.e. HSS, MME, S/PGW) were run on top of a virtual environment, like VM and Container JUJU [19] . Even the eNodeB might be run on VM or Container with access to the Radio Unit (RU).

Aiming at providing monitoring interfaces for the scalability-based management algorithm developed in this task, we developed techniques to get information on the OAI VNF and OAI PNF, focusing on MME and eNodeB. Note that the eNodeB is a monolithic PNF or VNF, and no functional split is considered (i.e. Cloud RAN).

1. eNodeB: the OAI eNodeB is considered as Physical Network Function (PNF). In addition to system-level information (i.e. CPU, Memory, etc.), we have developed an API to provide information on the RAN information. The latter covers the radio quality indicators, the control plane, and the data plane.

    - Radio quality indicators cover the UE/eNodeB layer1/layer2 parameters. It includes up-to-date information regarding the configuration and status of UEs and the access network. We may mention the following: UE's configuration information (e.g., PLMN ID, C-RNTI, downlink/uplink (DL/UL) bandwidth); UE status information GNSS); eNodeB configuration information (DL/UL radio bearer configuration, tracking area code, PLMN identity); eNodeB status information (GNSS, DL/UL scheduling information, number of active UE).
    - Control-plane interface exposes information on UE/eNodeB layer3 and S1/X2 interface messages used for network control. We may mention, UE status information (mobility state, mobility history report (X2); radio link failure report); eNodeB status information (including, PRB usage per traffic class).
    - Data-plane interface provides information on the X2-U and S1-U interfaces, such as UE configuration information (EPS bearer identity, bearer type (default or dedicated), bearer context (CQI, ARP), bearer bit rate (GBR or MBR)), UE status information (QCI, CSI), and network status information (aggregated PRB usage, delay jitter of specific QCI).

The API is accessible through the FLEXRAN framework, which is composed by an agent located at the eNodeB and a remote controller. The FlexRAN southbound API is using google buffer protocol to enable the communication between the remote controller and the agent. The agent is executing the requests of the controller, which consists in GET messages to obtain the above information on RAN. It worth noting that the FlexRAN controller exposes the API to a third-tier application via a Northbound API based on REST and JSON. For more details on FlexRAN please refer to FlexRAN [20] .

2. MME: Regarding the OAI MME VNF, an API has been developed to access most relevant information that might be used by a remote application. This information is related to the number of connected eNBs, the number of attached UE, the number of connected UEs, the number of default bearer and the number of S1-U bearers. The proposed API are based on the google grpc procedures. A grpc server

is hosted in at the MME side. It replies to grpc clients with the above three information. The google grpc is the modern, lightweight communication protocol from Google. It's a high-performance, open-source universal remote procedure call (RPC) framework that works across a dozen languages running in any OS. The first version of this API is not rich as the one provided for eNodeB, but it could be easily enriched according to the application's needs.

6.6. Scalable Management for MVNO

6.6.1. Introduction

In Japan, the number of MVNOs has increased nearly 6 times in three years since 2014. (125 MVNOs in 2014, to 784 MVNOs in 2017)

This is the result of MIC's competition policy on service price and ICT service expansion. In addition, digital transformation and IoT market have been growing and it accelerated MVNO business in Japan.

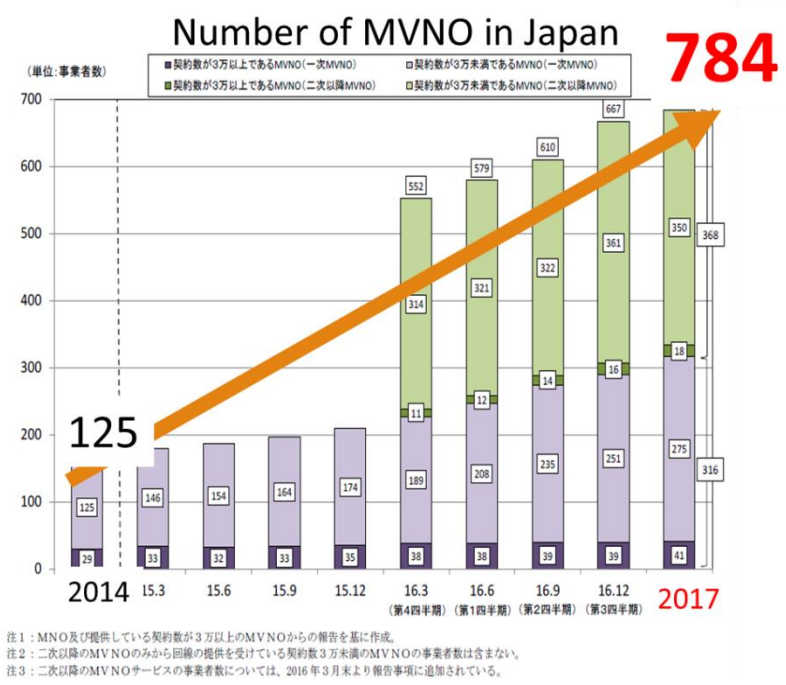

Figure 25 – Number of MVNO in JAPAN

As the number of MVNO increased, intensification of price competition and the expansion of unique services combined with communication service has been advanced. Current business model of MVNO in Japan is to lease network resources from MNO and then provide network service combined with unique applications to end users. In this business model, the efficiency of subscriber accommodation is a key factor of success.

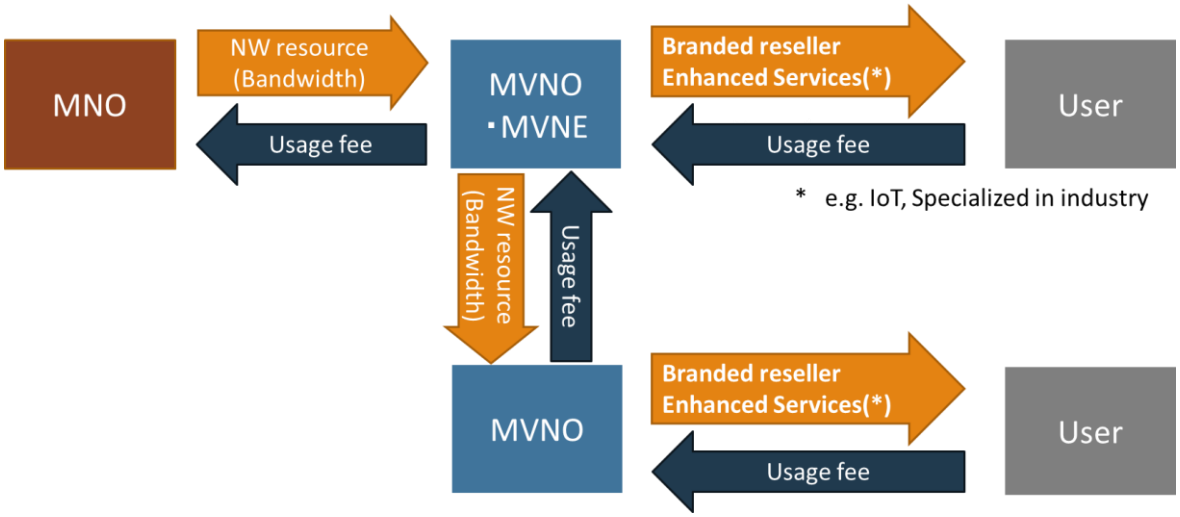

Figure 26 – Business model for MVNO

As competition is intensifying, in order to survive and differentiate in the market, it is important to maximize the usage of limited resources, to provide service quality to increase the customer satisfaction, and unique communication services differentiated from competitors. Regarding communication quality, currently, it is impossible for MVNO to control MNO's resources. Therefore, in order to secure end-to-end communication quality, it is necessary to allocate resources with a margin. It is the issues of efficient resource management by MVNO. In 5G, it is important for MVNO to be able to secure end-to-end network quality dynamically for each service. In order for MVNO to broaden the range of service creation, it is also important to enable to deploy services not only on the resources held by MVNO but also on the network resources provided from others, e.g. mobile edge resources area, depending on the service characteristics. Also, in Japan, some MVNOs are developing MVNE business by utilizing its facilities and business operation know-how. MVNE aims to expand the business by leasing its network resources to secondary MVNOs. To increase its profit MVNE needs to enrich the amount of resources and resource types and it is also necessary to assign network resources on-demand basis to MVNO when required.

6.6.2. Orchestration for MVNO

The followings are the key factors of scalable management for MVNO and MVNE.

- MVNO: To ensure a dynamic end-to-end network quality according to increasing / decreasing services, and allocation of an original service on those resources.
- MVNE: To hold cloud resources in the resource pools and assign those resources to MVNO on-demand basis, which can play an efficient role in operational management of the resource pools.

Based on the above consideration, the requirements from MVNO are as follows:

6.6.2.1 MVNO Viewpoint

- Flexible and scalable management of resources in MVNO resource pool
- Mechanism to provide MVNO's network and service resources (e.g. IoT system, specific services) on MVNO slices.
- End-to-end slice creation, configuration, and termination using resources in the resource pool

- Observation of SLA related to bandwidth, latency, priority and security level
- Allocation of applications to each slice
- Allocation of computing resources and service programs such as Mobile Edge Computing
- Slice monitoring and reconfiguration of slice
- Functions to be provided through a common API

6.6.2.2 MVNE Viewpoint

- Flexible and scalable management of resources in MVNE resource pool
- Registered resources to be provided to MVNO from MVNE resource pool
- Status of Resource pool usage to be monitored
- Aggregated resources to be provided for MVNO on-demand basis
- Functions to be provided through a common API

6.6.3. Orchestration architecture from MVNO view

The architecture of orchestration from MVNO/MVNE's viewpoint is illustrated in Figure 27.

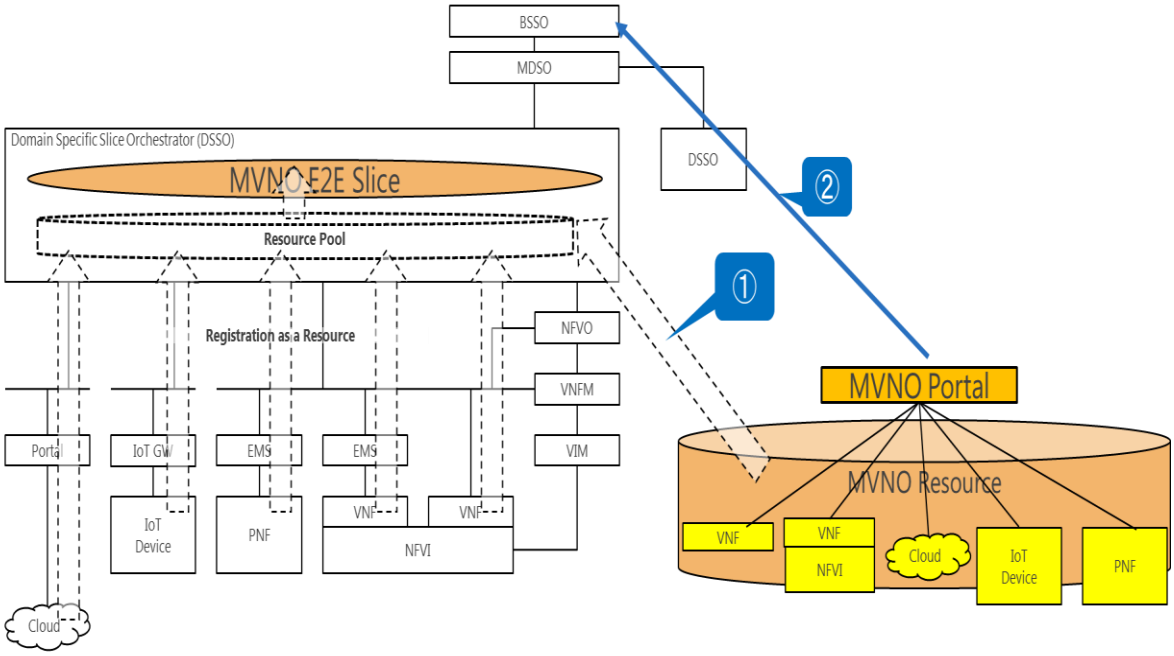

Figure 27 – System Architecture from MVNO's viewpoint

(1) To register the resources owned by MVNO / MVNE in the resource pool on the DSSO. At the registration stage, open ranges for others are defined for each resource. Resources defined as open resources can be assigned to other MNOs and MVNOs.

(2) To request the resource necessary for service deployment, network slice, and SLA for network slice from the portal held by MVNO to the BSSO. Appropriate resources are selected from the resource pools and those are provided for MVNO through MNO, MVNE, etc., and the slices for which MVNO secured the end-to-end SLA are generated.

6.6.4. Use case for MVNO

Use cases in the areas of healthcare, and the reasons why an end-to-end slice scalability management is required by MVNO are described as below:

In Japan, it is expected that the needs for medical and healthcare, and the burden on healthcare workers will increase because of the aging society. It is expected that introduction of ICT will accelerate to improve medical quality and to reduce the burden on healthcare workers.

MVNO will advance services such as data collection by sensors, voice communication, electronic medical record, wearable IoT, and so on.

In order to guarantee communication quality of those services with different communication requirements, it is required to realize an end-to-end slice. In particular, delay in telemedicine service will become a critical problem that may cause medical malpractice. In addition to that, a security whose policy is different depending on the information to be handled will become a more highlighted problem. This is because of handling of personal information, e.g. medical treatment utilizing ICT and telemedicine among other things becomes popular, and security will become crucial to control the permission of critical information.

Those delay and security problems can be solved by deploying an end-to-end slice, which satisfies those requirements. It is also considered an effective approach to realize not only an individual device security by ICT but also an integrated security by network functions. MVNO can satisfy its own service needs by creating ‘security NFV’ functions and adopting them to the network slices to realize the best security for each service.

7. Concluding Remarks

The objective of this deliverable is to define the detailed 5G!Pagoda architecture, especially to the architecture for scalable orchestration and management. We decided to start this document with extracting of the overall requirements for the purpose of scalable management and orchestration from the use-cases, which have been described in D2.1.

Although eMBB, URLLC, and mMTC are publicly known as the three major requirements for the general 5G system, but there have been so far very few discussions about concrete KGIs (Key Goal Indicators) to fulfill each use case requirement. Based on intensive discussions, we identified six keys and values to characterize eight different use cases. These keys are (1) Number of Slices, (2) Location, (3) Frequency, (4) Lifecycle Time, (5) Heterogeneous, and (6) Hierarchy Level.

Next, the 5G!Pagoda scope definition of the orchestration and the management has been fixed based on the definition presented by the ITU-T FG-2020, and we presented 5G!Pagoda Scalability-driven management and orchestration system architecture. Because the concrete six KGIs and system architecture were clarified, we decided to step forward to clearly state our scalable orchestration policy described by the following three KPIs:

(1) Number of Instances

(2) Status Change Frequency of Instances

(3) Latency of Computing Processing

We mapped the six KGIs to three KPIs based on our commercial expertise and insight regarding the scalable operation and maintenance. The 5G!Pagoda orchestrator functions and control method must always satisfy the three KPIs. We introduced a new infrastructure design concept called ‘Resource Pool’, which is the key to achieving scalable orchestration architecture. The ‘Resource Pool’ is a collection of digitalized virtual assets, which can be converted from all the physical assets. Network slice is one of the virtual assets. Using ‘Resource Pool’ as a middle-layer function of Resource Orchestrator (RO), we may quickly cope with changes and/or failures, e.g. resource demand up and down in the upper service layer, a sudden physical failure in the lower network layer and so on. Since ‘Resource Pool’ is intended to provide necessary and sufficient resources for the corresponding upper and lower tasks based on dynamic and smart forecasting. As a result, ‘Resource Pool’ based orchestrator can make the most use of limited resources and thus provide a highly agile and resilient 5G service platform for service and network operators and their end customers.

In parallel with the considerations for the 5G! Pagoda orchestration scheme, we studied and defined slice compliant, scalable management functions for each technology domain, typically the NFV domain. Finally, based on our studies and outcomes, further discussion on the “Network Slice Orchestration” in D4.2 and “End to End Network Slice” in D4.3 will follow.

Appendix A. Research activities related to orchestration and management of slices

Network slicing technology raises a lot of important issues related to slices lifecycle as well as lifetime management. Typically, in software-based networking solutions, the lifecycle of networks (or slices) is done by the network orchestrator. In some concepts, however, the orchestration is only a part of a bigger management picture. In general, the orchestration functionality is providing automated operations whereas the management is focused on the wider scope and interactions with network/slice operator. In fact, all management/orchestration related operations that are implemented in classical networks have to be implemented in sliced networks. Therefore, all existing approaches to network management (with exceptions mentioned in this sections) are applicable to network slice management.

In opposite to classical network management, the slice management has following differentiators:

- There is a need to manage not only a single network but multiple parallel networks – this issue makes management a critical part of the network slicing architecture.
- There is a need to split management functions between orchestration and management.
- There is a need to specify the slice tenant management functions.
- In cases when a slice is created for a specific service, the network management can be combined with service management.

As it has been pointed out, from the management point of view the sliced network can be treated in a similar way as classical networks; therefore, the generic scheme of Telecommunication Management Network (TMN) as defined in M.3000 recommendation can be applied in this case. It is worth to mention that M.3000 [2] splits the management into following layers:

- Business Management (BM) layer is responsible for fulfilling business goals, providing data for billing etc.,
- Service Management (SM) layer focuses on each service management. Typically, each service platform has its dedicated management platform.
- Network Management (NM) layer is responsible for the overall network management. Typically, it is split into subsystems responsible for the management of separate technological domains. Such approach is justified by a different set of features required by each domain.
- Element Management (EM) layer is responsible for the management of the individual Network Elements (NE) or network functions. In a classical hardware-based network the EM is responsible for all hardware related problems (link break, power or fan failure).

The SM, NM and EM classical functions, called commonly FCAPS, include handling of faults, performing configurations, collecting and processing data for accounting and performing security functions.

The described approach has several deficiencies. It is centralized and very complex; typically, human-operated with a low level of automated operations, therefore human error prone. Moreover, service management platforms are loosely integrated with the network management one. The NMs of different technological domains are loosely integrated. The mentioned features lead to lack of scalability and inefficient operations of the classical management.

In the context of network management, it is also necessary to describe another paradigm that is Policy-Based Management framework (PBM). The PBM enables more than individual EM management – it enables dynamic changes of a set of network/service nodes by a policy script. Thus, PBM provides to operators some level of management functions programmability and easier operations in comparison to one-by-one operations on EMs. Most of the current deployed PBM solutions are based on the Event-Condition-Action paradigm (ECA). In ECA, the PBM script for each event should provide description of low level actions (if <condition(s)> then <action(s)>). The PBM approaches typically use the PBM architecture that has been developed by IETF/DMTF [3] . The architecture is composed of the following functional elements: The Policy Management Tool (PMT), Policy Repository, Policy Decision Point (PDP), and Policy Enforcement Points (PEPs). Within last years there is a shift from ECA-based PBM to the Intent-based PBM. In the latter case the policies are described as intents (high-level goals), i.e. they describe what should be done, but not how. In such case, the Policy Engine converts the high-level goals into low-level atomic operations. The Intent-based PBM minimizes policy conflicts that can happen in case of ECA-based PBM. Moreover, a certain level of agnosticism of policies in the Intent-based PBM can be achieved; therefore, a change of network topology (for example) doesn't require a change of the policy. Such a change will be handled by the Intent Engine.

For nearly 15 years an intensive research effort has focused on the evolution of the network management systems. These efforts, so far, focused on two main directions. The first lies in distributing management functions and embedding some of them in EMs. That way each EM is able to take some local decisions, especially related to self-configuration. Such concept has been developed in the framework of the FP7 4WARD project and called in-network-management (INM) [4] . Unfortunately, this concept has gained no commercial acceptance so far.

The second direction of network management that has been developed for about 15 years is so-called Autonomic Network Management (ANM). This network management is based on the feedback control loop in which certain parameters of the networks (or objects in general) are monitored, analyzed and a decision about changes of object configuration is taken and executed. The concept (introduced by IBM in 2001 in the context of autonomic computing [5]). It is sometimes referred to as Monitor-Analyse-Plan-Execute (MAPE). The concept in network management is used for self-configuration, self-healing, self-optimization, self-protection etc. An essential part of the ANM framework is the monitoring part that collects and processes information related to the controlled object and its environment. On that basis, an algorithm that drives the behavior of the object has to take a decision. In most cases, such algorithm is not aware of its previous decisions. In the case when the algorithm has learning capabilities, it is called the Cognitive Network Management (CNM). So far there were many research projects related to autonomic and cognitive network management (FP6 ANA, FP7 SOCRATES, FP7 EFIPSANS, FP7 UNIVERSELF, FP7 SEMAFOUR, CELTIC COMMUNE) and there are many H2020 ones related to 5G network management as well. Despite intensive efforts, the concept of ANM has no market acceptance yet. The only exception is the LTE SON approach in which autonomic functions are used for handover and coverage optimization, energy-efficient operations, or plug-and-play eNodeB deployment. Despite commercial deployments, SON still has some problems; the most important one is so-called coordination of several SON functions. The problem is linked to individual autonomic functions of contradictory goals. For example, performance optimization may lead to the configuration of eNodeBs different to energy-saving one. It is worth noting that in the framework of the EFIPSANS project, a concept called GANA (Generic Autonomic Networking Architecture) [6] has been developed, which is subject to ongoing standardization by ETSI. So far there are no deployments based on GANA.

In the network softwarization era, the approach to network management has to be changed. Sometimes the proposed approaches are not in line with the main directions of the network management evolution. One of them is the SDN case. In this centralized concept, the data plane devices (called OpenFlow switches)

have no embedded intelligence and all control plane operations (at least per domain) are performed by a logically single entity, which is the SDN Controller. In some concepts (see ONF [7]) the Controller is not only responsible for the Control Plane operations, but also for the management tasks. Such approach is supported by the MCC concept of TMF [8] , in which there is no separation between control and management planes. In some SDN platforms, the NetConf protocol is used to implement this concept. Unfortunately, the approach has many drawbacks as listed in IEJCE [9] .

The greatest revolution in networks is driven by ETSI NFV which specified the VNF-based (Virtual Network Function – a software entity) networking solutions. The management and orchestration in such case (provisioning, the configuration of VNFs, resource allocation) is done according to the ETSI NFV MANO framework [10] . The NFV MANO consists of three functional blocks: VIM (Virtual Infrastructure Manager), NFVO (NFV Orchestrator) and VNFM (VNF Manager) as well as four data repositories: NS Catalogue, VNF Catalogue, NFV Instances and NFVI Resources. The VNFO can orchestrate both resources and services. It has to be noted that MANO is not replacing classical management systems. In fact, in the MANO approach, the OSS/BSS exchange information with EMs of VNFs and VNFM. According to [11] the split between OSS/BSS and MANO is following: MANO entities deal with virtualization mechanisms, whereas OSS/BSS functions are responsible for specific network services being provided by a collection of VNFs.

So far, the MANO approach has been not yet validated on a large scale and has some problems. It is a centralized solution, which implies scalability problems, especially in the context of multiple administrative or technical domains and in case of network slicing, where multiple network instances should coexist. The problem has been already discovered with the suggestion of distribution of the MANO intelligence to allow taking local decisions and shorting the round-trip time of management related messages Diego Lopez. Moreover, there are no MANO solutions on the market yet that provide full support for FCAPS. The NFV standardization is progressing and recently new functions related to fault or performance management have been added. Recently more details about the interactions between the OSS/BSS and NFVO has been provided. It is worth to emphasize that MANO is a centralized solution that raises scalability issue of the approach. Additionally, most of the communication between the OSS/BSS and the MANO-based network has to be done via NFVO. In the context of network slicing, this approach can't be accepted as it is not scalable, and represents a single point of failure. Unfortunately, the ETSI working group dealing with the distribution of MANO functions has suspended its activity due to the lack of satisfactory progress. So far, no orchestration in multiple administrative domains has been defined.

In MANO, there is a separation between OSS/BSS functions and MANO specific orchestration; but in many approaches, the OSS/BSS functions are not defined. However, according to the context of IMT-2020 [1] the orchestration goes beyond MANO and includes processes aiming at the automated arrangement, coordination, instantiation and use of network functions and resources for both physical and virtual infrastructures by optimization criteria. Contrariwise, the IMT-2020 group has defined management as a set of processes aiming at fulfillment, assurance, and billing of services, network functions, and resources in both physical and virtual infrastructure including compute, storage, and network resources.

The described evolution of network management and orchestration shows that there is so far, no solution of choice for network slicing. The existing concepts collide sometimes and still a good mix of the mentioned ones has to be found. There is no doubt that the scalability issue linked with a potentially huge number of network slicing is the most challenging. It seems that some changes to the existing MANO approach cannot be avoided due to the need of distribution of orchestration/management functionalities in line with the in-network management paradigm as well as with autonomic or cognitive network management approaches.

It is worth to emphasize that in order to achieve scalable management; the following factors have to be linked together:

- the management/orchestration architecture that allows a distribution of management functions,
- a set of management protocols that allows efficient management information exchanges (for example gossiping based),
- algorithms that drive the management functions and their decomposition.

Moreover, the management should be programmable, giving that way possibility for modification and addition of management services during the network (or slice) runtime.

Appendix A.1. 5G PPP

In the framework of 5G PPP projects the aspects of slicing and their management is addressed by several projects. In many cases, slice management and orchestration are combined with the overall network slicing architecture. Some of them have already been analyzed in D2.3 [21] . The most important synthetic achievement is the 5G PPP white paper describing the overall system architecture. Its first version was published on July 2016 (version 1.0) and an updated version (2.0) [22] has been published in July 2017. The overall architecture is presented in Figure 28

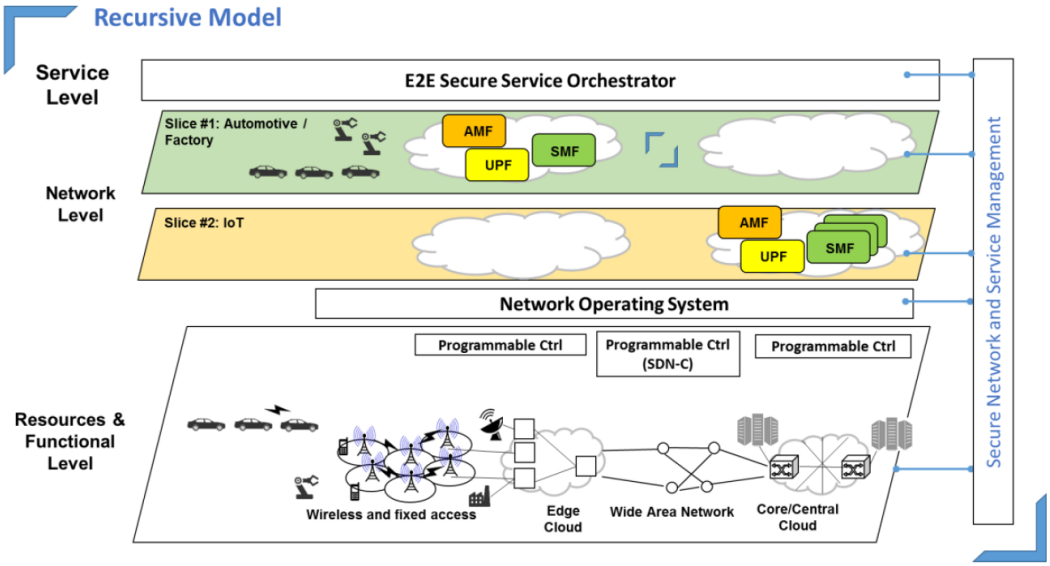

Figure 28 – 5G PPP Overall architecture [23]

The existence of two orchestrators in the picture has to be noted – one of them is responsible for network and service management whereas the second one is responsible for service orchestration. The architecture functional layers are presented in Figure 29. In the context of management and orchestration, the most important one is the Management and Orchestration layer. It includes ETSI NFV MANO functional entities (VIM, VNFM, and NFVO) as well as domain-specific management, which comprises Element Management (EM) and Network Management (NM) functions, including Network (Sub-)Slice Management Function (N(S)SMF). For slice stitching, the Inter-Slice Broker has been introduced. It interacts with the Service Management function. In the presented approach the Service Management is an intermediary function between the Service layer and the Inter-Slice Broker. It transforms service descriptions into resource-oriented ones. The Service Management is driven by BSS and service-oriented policies.

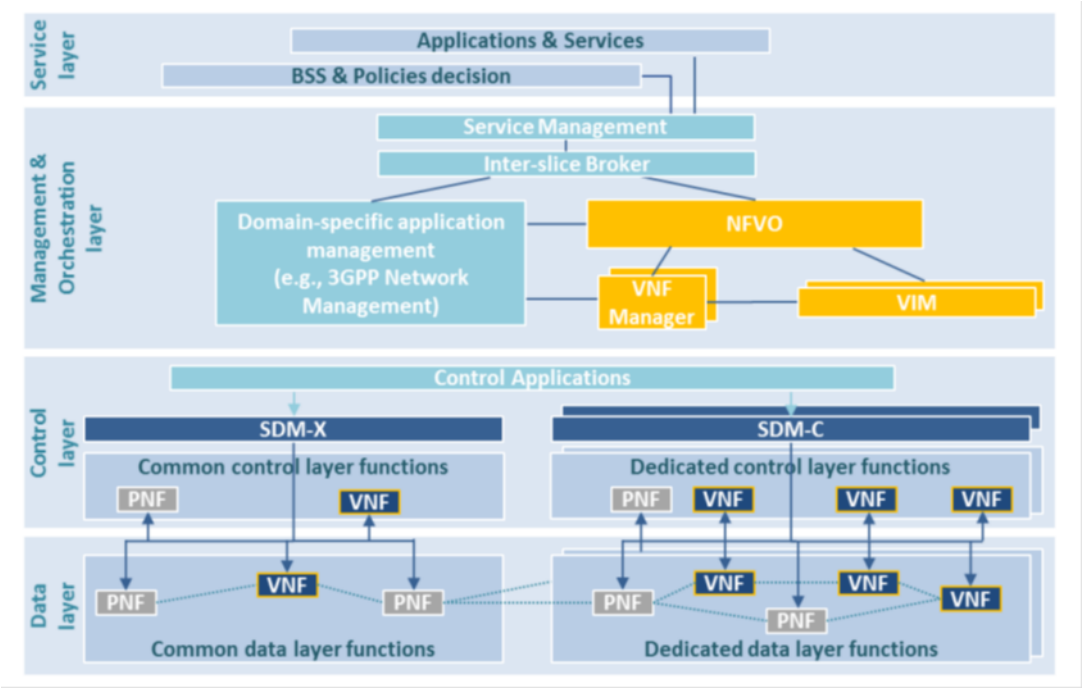

Figure 29 – 5G PPP architecture layers for single domain [23]

In the analyzed concept also, inter-slice MANO is presented. In the approach, each slice domain has Software-Defined Mobile Network Controller (SDM-C) as a part of slice operations specific VNF. The SDMC, based on slice performance reports received by QoS/QoE Monitoring and Mapping module, may adjust the network slice configuration either by reconfiguring some of the VNFs in a network slice or by reconfiguring data paths in an SDN-like style.

For the inter-slice operation, an additional set of VNFs is created. The inter-slice operations are performed via the Inter-Slice Broker that uses OS-Ma-NFvo interfaces to interact with NFVOs and OSSes/BSSes of each domain in order to perform its tasks, moreover, the Software Defined Mobile Network Coordinator (SDM-X) is used to interact with SDM-Cs.

The inter-slice management and orchestration are seen as a key feature of the proposed 5G architecture that supports multi-service and multi-tenancy operations but no details about service nor tenant management are provided. In the discussed approach it is assumed that the architecture shall allow each tenant to deploy and have a full degree of control of its slice. It led to the recursive architecture that allows for the building of recursive slices over the virtual infrastructure. In this approach, each tenant may have its own orchestrator what solves nicely the issue of orchestration scalability.

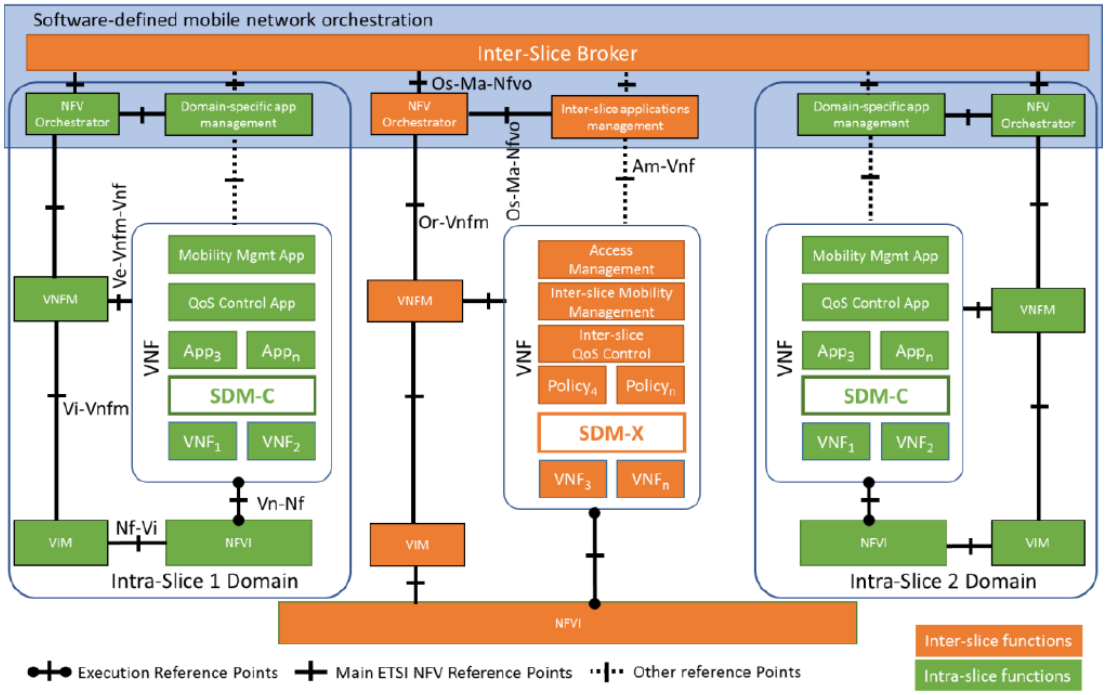

Figure 30 – 5G PPP Inter-domain approach [23]

The analyzed document includes also a chapter on cognitive network management. In Figure 31 there are presented additional components that are required in order to implement the autonomic or cognitive management in the MANO framework.

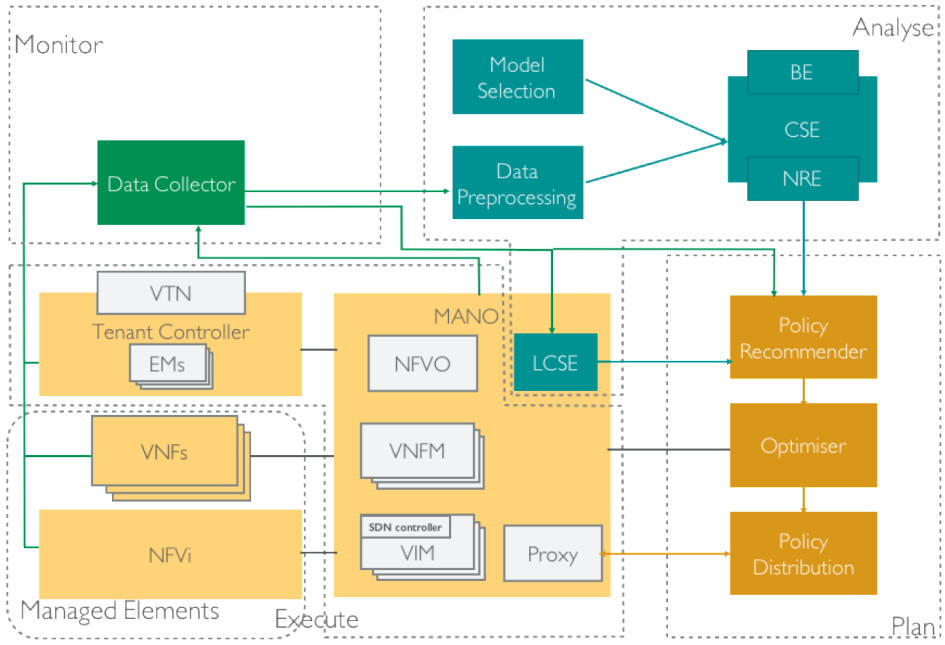

Figure 31 – 5G PPP approach for integration of the MANO framework with autonomic/cognitive management (a single domain case) [23]

As it can be seen from the picture additional to MANO functional entities are used in order to implement the MAPE approach (Monitor-Analyse-Plan-Execute). The Data Collector and Data Pre-processing functional entities are responsible for the monitoring part (SLA focused), the Cognitive Smart Engine (CSE) uses machine learning; Policy Recommender, Optimiser, and Policy Distribution functional entities implement

policy-based management (as a MAPE "Plan' function). The Execute part is based on NVFI, VNFS and MANO functional entities. A more detailed picture (Figure 32) shows additional the MANO framework functional components.

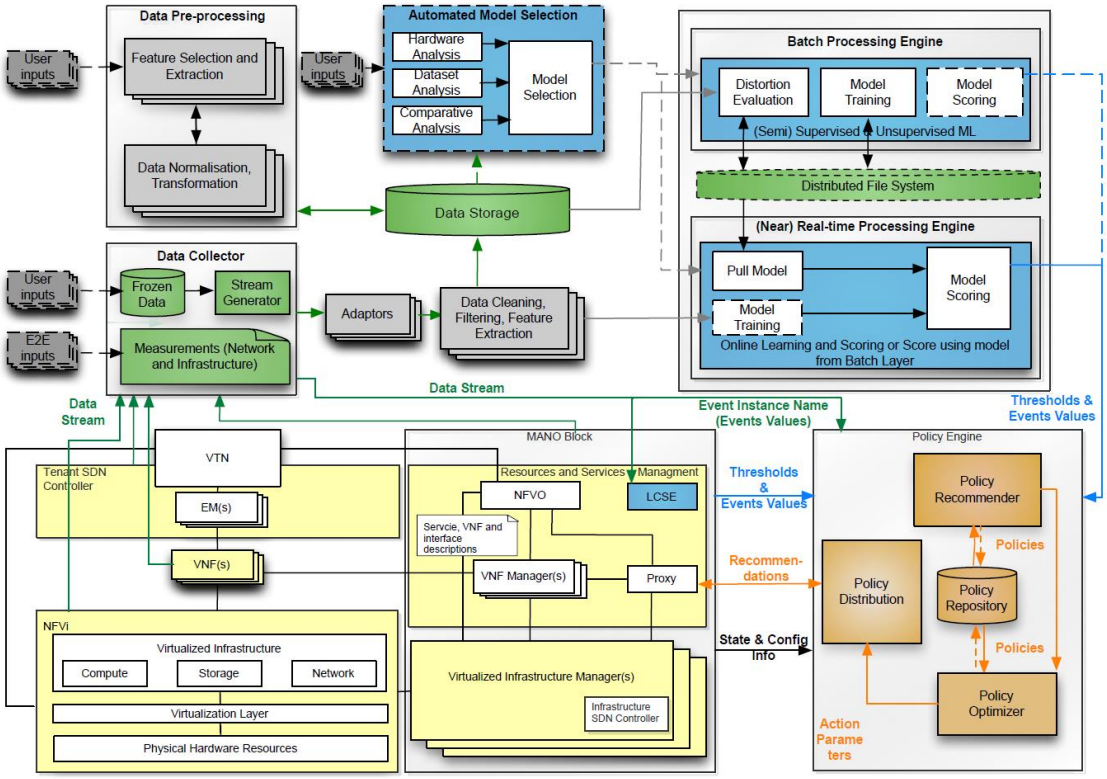

Figure 32 – 5G PPP autonomic/cognitive management operational architecture [23]

In the 5G PPP White Paper about Software Networks [24] there is included a figure showing resource and service orchestration (Figure 33). In this concept, the resource orchestration is performed by MANO (being a slave to OSS/BSS) and service orchestration is done by OSS/BSS supported by EMs of the MANO framework. It is worth to emphasize that EMs allows for distribution of some management functions in the in-network-management spirit. In the concept a generic NFV network is a subject of resource and service orchestration – no slice-specific issues are addressed.

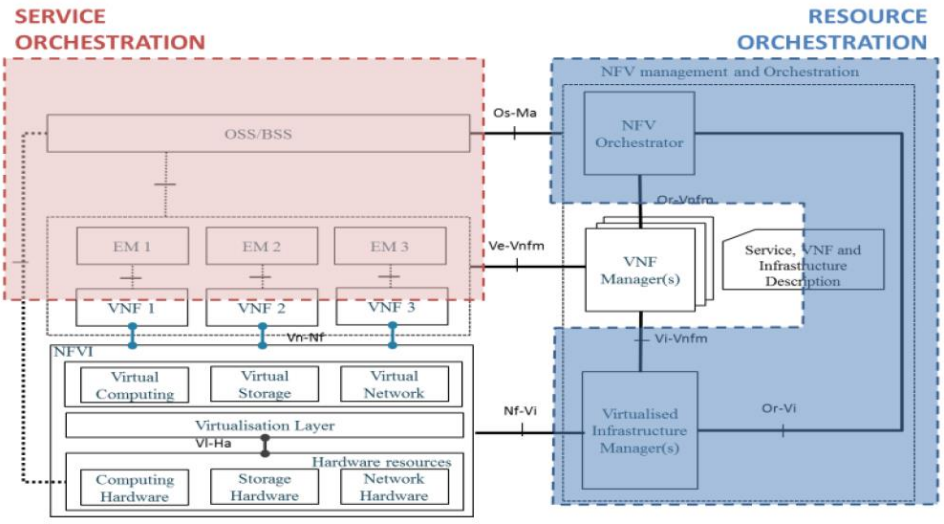

Figure 33 – 5G PPP autonomic/cognitive management operational architecture [24]

In this worth mentioning that at present there are several 5G PPP projects that work on the orchestration and management of 5G networks and some of them address also network slicing management and orchestration. Some of them are in a very early stage.

Appendix A.2. 3GPP

The 3GPP has recently published several documents referring to miscellaneous aspects of management of the lifecycle of network slices and 5G network management and orchestration. The documents related to network slicing are:

- TS 28.530 – Management of network slicing in mobile networks; Concepts, use cases, and requirements;
- TS 28.531 – Provisioning of network slicing for 5G networks and services;
- TR 28.801 – Study on management and orchestration of network slicing for the next-generation network;
- TS 29.531 – 5G System; Network Slice Selection Services; Stage 3;
- TR 33.811 – Study on the security aspects of 5G network management slicing.

Almost all of the listed documents, except TR 28.801 (version 15.0.0, 25-09-2017) are at very initial stage, what makes impossible to analyse them, moreover TS 29.531 and TR 33.811 are out of scope of this analysis (slice selection is not a part of the management and 5G!Pagoda is not dealing with security issues).

The mentioned 3GPP Technical Report TR 28.801 is accompanied with two other complementary TRs creating together the full picture of 5GS management and orchestration:

- TR 28.800 – Study on management and orchestration architecture of next-generation network and service (version 0.8.0, 05-09-2017);
- TR 28.802 – Study on management aspects of next-generation network architecture and features (version 1.0.0, 07-09-2017).

The maturity of those documents is relatively low, but already it is sufficient to determine the requirements and future directions. Especially, the management architecture design and definition of management interfaces is still a green field. It has to be additionally noted, that TR 28.801 (recognized by 3GPP as mature, by labeling it with version number 15) still contains a number of open questions left for further study.

The 3GPP approach is based on two basic concepts: Network Slice Instance (NSI) and Network Slice Subnet Instance (NSSI). The first one is comparable with 5G!Pagoda term ‘E2E Slice’ and its completeness mean having all functionalities and resources necessary to support a certain set of communication services (serving a business purpose). NSI at the fundamental level is composed of NFs (both AN and CN ones), realized with corresponding physical and logical resources, and its composition is described by NS template that can be individually enriched with some instance-specific information (parameters, policies). NSI may be fully/partially isolated from other NSIs both at physical and logical level. NSSI concept is introduced due to NSI management logic and means a constituent of NSI being a specific subset of atomic NFs or other NSSIs. Such NSSI may be shared by more than one NSI or NSSI in the same logic, as some NF may be shared by more than one NSI or NSSI. The NSSI concept, although having a wider meaning, maybe to some extent compared to 5G!Pagoda ‘(sub-)slice’ term, especially ‘common’ and ‘dedicated’ slice, although they were previously used in the context of administrative (e.g. related to ownership, vendor, organization, organic

architecture, topology, orchestrating system) or technology-driven (AN, CN, transmission, etc.) partitioning to slicing domains.

TR 28.801 [25] lists the following issues related to the management that are in scope of it:

- Issue 1. Aligning of slice instance related terms,
- Issue 2. Creation of the network slice to support services provided by the operator,
- Issue 3. NSI FCAPS management,
- Issue 4. SON evolution for network slice management,
- Issue 5. Orchestration of network slice,
- Issue 6. Shared (multiple services) slice instance management,
- Issue 7. Orchestration of Slice across multiple administrative domains.

The lifecycle phases of a Network Slice Instance are presented in Figure 34.

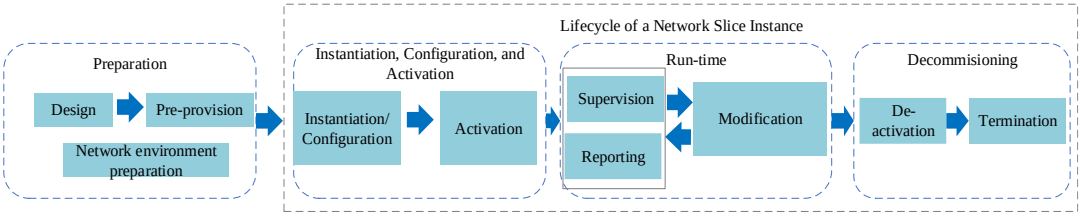

Figure 34 – The lifecycle phases of a Network Slice Instance [25]

The 3GPP approach defines the following management functions related to NSSI that are listed below in order of corresponding hierarchy:

- Communication Service Management Function (CSMF). The CSMF manages Communication Services provided by the Network Operator according to the requirements from the Communication Service Customer, converts these requirements to the NS requirements (e.g. network type/capacity, QoS requirements, etc.) and delegates management of NSIs to NSMFs;
- Network Slice Management Function (NSMF). The NSMF manages NSIs, according to the requirements from CSMF and further converts/splits them to NSS requirements and delegates management of NSSIs to NSSMFs;
- Network Slice Subnet Management Function (NSSMF). NSSMF manages NSSIs based on the requirements received from the NSMF.

In terms of NS lifecycle management, the 3GPP goal is to provide management of the lifecycle of the following classes of services: CS, NSaaS (i.e. NaaS) and NSSaaS (i.e. IaaS). Despite the fact that these classes of services may be mapped to the corresponding management functions (CSMF, NSMF, and NSSMF respectively), the entry point of creation process of CS or NSaaS service will start at the level of CSMF – the difference between these two scenarios is in the location of ownership border (in case of CS, the service provider owns the CSMF; in case of NSaaS the service customer owns the CSMF). In case of NSSaaS, the process entry point is NSMF of the customer. After being created, the NSSI may be activated, modified, deactivated or terminated. All of these actions will be managed by its NSMF with the involvement of subordinate NSSMFs. The run-time phase management/orchestration-related operations include supervision/reporting (e.g. for KPI monitoring), as well as modification tasks, (e.g. upgrade, reconfiguration,

NSI scaling, changes of NSI capacity, changes of NSI topology, association and disassociation of network functions with NSI).

In the context of 5G network management there are four groups of objects to be managed:

- 5G services,
- 5G slice instances,
- 5G network functions and
- 5G resources (both virtual resources and physical).

The handling of faults and performance in the described approach is following:

- Fault Management data (performance alarms) are sent from NFs to NSMF;
- Network fault management data (event notifications) are generated at NF, NSSI;
- Performance Management data are generated/aggregated at NF, NSSI to show performances of objects at these levels;
- Communication Service performance data are generated by CSMF based on service-related measurement data retrieved from NSMF;
- Communication Service fault management data (alarms) are CSMF based on use NSI alarms associated with the service and retrieved from NSMF.

The use of 4G SON-like paradigm is expected and should allow for per slice policy based NSI automated configuration and reconfiguration, automated optimization and automated self-healing of slices. In case of self-healing, the distribution of policies to NSSMFs enables triggering of local actions even at the level of single NSSIs.

Appendix B. References

[1] ITU, "Y.3100: Terms and definitions for IMT-2020 network", Y.3100 (2017-09); http://www.itu.int/rec/T-REC-Y.3100/en.

[2] ITU-T, "M.3000: Telecommunications management network, Overview of TMN Recommendations", M.3000 (02/2000), source: https://www.itu.int/rec/T-REC-M.3000-200002-I/en

[3] IETF, "Terminology for Policy-Based Management", RFC 3198, 2001-01; source: https://tools.ietf.org/html/rfc3198.

[4] FP7-4WARD project, "D-4.2 In-Network Management Concept", 2009-03-31; source: http://www.4ward-project.eu/index9d96.html?s=Deliverables.

[5] IBM, "Autonomic Computing White Paper: An architectural blueprint for autonomic computing", 3rd edition, 2005-06; source: https://www-03.ibm.com/autonomic/pdfs/AC%20Blueprint%20White%20Paper%20V7.pdf.

[6] ETSI, "Generic Autonomic Network Architecture (An Architectural Reference Model for Autonomic Networking, Cognitive Networking and Self-Management)", ETSI GS AFI 002, V1.1.1 (2013-04); source: http://www.etsi.org/deliver/etsi_gs/AFI/001_099/002/01.01.01_60/gs_afi002v010101p.pdf.

[7] Open Networking Foundation, https://www.opennetworking.org/.

[8] TMF, "IG1118 OSS/BSS Futures: Preparing the Future Mode of Operation, R15.5.1 Standard", 2016-03-18; source: https://www.tmforum.org/resources/exploratory-report/ig1118-ossbss-futures-architecture-r15-5-1/.

[9] S. Kukliński, P. Chemouil, "Network Management Challenges in Software-Defined Networks", in IEICE Transactions on Communications, vol. E97-B, no. 1, 2014, pp. 2-9.

[10] ETSI, "Network Functions Virtualization (NFV); Management and Orchestration", ETSI GS NFV-MAN 001 V1.1.1 (2014-12); source: http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf.

[11] R. Mijumbi, J. Serrat, J.-L. Gorricho, S. Latré, M. Charalambides, D. Lopez, "Management and Orchestration Challenges in Network Functions Virtualization", in IEEE Communications Magazine, 2016-01, pp. 98-105.

[12] ETSI, "Network Functions Virtualization (NFV); Management and Orchestration; Os-Ma-Nfvo reference point - Interface and Information Model Specification", ETSI GS NFV-IFA 013, V2.1.1 (2016-10); source: http://www.etsi.org/deliver/etsi_gs/NFV-IFA/001_099/013/02.01.01_60/gs_NFV-IFA013v020101p.pdf.

[13] ETSI, "Network Functions Virtualization (NFV); Management and Orchestration; Report on Architectural Options", ETSI GS NFV-IFA 009 V1.1.1 (2016-07); source: http://www.etsi.org/deliver/etsi_gs/NFV-IFA/001_099/009/01.01.01_60/gs_NFV-IFA009v010101p.pdf.

[14] ETSI, "Network Functions Virtualization (NFV); Management and Orchestration; Ve-Vnfm reference point - Interface and Information Model Specification", ETSI GS NFV-IFA 008 V2.1.1 (2016-10);

source: http://www.etsi.org/deliver/etsi_gs/NFV-IFA/001_099/008/02.01.01_60/gs_NFV-IFA008v020101p.pdf.

[15] http://portal.etsi.org/NFV/NFV_White_Paper2.pdf.

[16] Network functions virtualization and software management, Ericsson, White paper, Uen 284 23-3248, December 2014.

[17] S. Retal, M. Bagaa, T. Taleb, and H. Flinck, “Content Delivery Network Slicing: QoE and Cost Awareness,” in Proc. IEEE ICC 2017, Paris, France, May 2017.

[18] B. Mada, M. Bagaa, and T. Taleb, “Efficient transcoding and streaming mechanism in multiple cloud domains,” in Proc. IEEE Globecom 2017, Singapore, Singapore, December 2017.

[19] https://jujucharms.com/u/navid-nikaein/oai-epc

[20] X. Foukas, N. Nikaein et al. “FlexRAN: A flexible and programmable Platform for Software-Defined Radion Access Networks”, in proc. of ACM CoNEXT 2016, Irvine, CA, 2016.

[21] 5G!Pagoda, “D2.3: Initial report on the overall system architecture definition”, 2017-06-30; source: https://5g-pagoda.aalto.fi/assets/demo/attachement/delivrables/5G!Pagoda%20-%20D2.3%20-%20architecture.pdf.

[22] 5G PPP, 5G PPP Architecture Working Group, “View on 5G Architecture”, Version 1.0, 2016-07; source: https://5g-ppp.eu/wp-content/uploads/2014/02/5G-PPP-5G-Architecture-WP-July-2016.pdf.

[23] 5G PPP, 5G PPP Architecture Working Group, “View on 5G Architecture”, Version 2.0, 2017-07; source: https://5g-ppp.eu/wp-content/uploads/2017/07/5G-PPP-5G-Architecture-White-Paper-2-Summer-2017_For-Public-Consultation.pdf.

[24] 5G PPP, 5G PPP Software Networks Working Group, “Vision on Software Networks and 5G”, White Paper, Version 2.0, 2017-01; source: https://5g-ppp.eu/wp-content/uploads/2014/02/5G-PPP_SoftNets_WG_whitepaper_v20.pdf.

[25] 3GPP, “Telecommunication management; Study on management and orchestration of network slicing for next generation network”, 3GPP TR 28.801 V15.0.0 (2017-09-25), http://www.3gpp.org/ftp//Specs/archive/28_series/28.801/28801-f00.zip.

[26] http://culturedigitally.org/2014/09/digitalization-and-digitization/

[27] IEC Market Strategy Board "Factory of the future" project whitepaper, http://www.iec.ch/whitepaper/pdf/iecWP-futurefactory-LR-en.pdf