D2.2: Consolidated definition of use cases, business models and requirements analysis

Deliverable

Document ID	NECOS-D2.2
Status	Final
Version	2.0
Editors(s)	Luis M. Contreras (TID), Sergio Vivas (TID)
Due	30/11/2018
Delivered	03/12/2018

Abstract

This document contains the results of the techno-economic analysis that has been conducted in the NECOS project to better understand the opportunities and benefits that the existence of the NECOS platform will provide. It is a complement of its predecessor, namely D2.1, but it is presented as self-contained document.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF CONTRIBUTORS

Participant	Short Name	Contributors
Telefónica Investigación y Desarrollo	TID	Luis Miguel Contreras
Telefónica Investigación y Desarrollo	TID	Sergio Vivas Pleite
University of Macedonia	UOM	Lefteris Mamatas
University of Macedonia	UOM	Ilias Sakellariou
University of Macedonia	UOM	Panagiotis Papadimitriou
University of Macedonia	UOM	Sofia Petridou
Federal University of Uberlândia	UFU	Rafael Pasquini
Federal University of Uberlândia	UFU	Raquel Fialho de Queiroz Lafetá
Federal University of Pará	UFPA	Antônio Jorge Gomes Abelém
Federal University of Pará	UFPA	Billy Anderson Pinheiro
Federal University of Sâo Carlos	UFSCar	André Luiz Beltrami Rocha
Federal University of Goias	UFG	Leandro Alexandre Freitas
Federal University of Goias	UFG	Kleber Vieira Cardoso
University of Campinas	UNICAMP	Christian E. Rothenberg
University of Campinas	UNICAMP	David Moura
Federal University of Rio Norte	UFRN	Felipe Dantas
Federal University of Rio Norte	UFRN	Marcilio Lemos
Federal University of Rio Norte	UFRN	Alisson Medeiros
University College London	UCL	Alex Galis
Telecom Research and Development Center	CPqD	Douglas Salles Viroel
Universitat Politècnica de Catalunya	UPC	Javier Baliosian
Universitat Politècnica de Catalunya	UPC	Joan Serrat

REVIEWERS

Reviewer	Short Name	Institution
Paulo Ditarso	UFSCar	Federal University of Sâo Carlos
Javier Baliosian	UPC	Universitat Politècnica de Catalunya

Acronyms

Acronym	Description
API	Application Programing Interface
AR	Augmented Reality
BPMN	Business Process Model and Notation
C3 PSC	Command, Control and Communications Public Safety Center
CAT	Cache Allocation Technology
CDN	Content Distribution Network
CloudCO	Cloud Central Office
CORD	Central Office Re-architected as a Datacenter
CMC	Chassis Management Controller
CMT	Cache Monitoring Technology
DC	Data Center
ECP	Entry Cloud Provider
ECPS	Entry Cloud Provider Services
eMBB	enhanced Mobile Broadband
FCP	Federated Cloud Provider
FS	Federated Services
IC	Infrastructure Cost
MEC	Multi-access Edge Computing
mMTC	massive Machine-Type Communications
MTC	Metropolitan Tourist Centre
NS	Number of servers
PDT	Partially Defined Template
PPS	Price Per Service
SC	Server Cost
SLA	Service Level Agreement
SO	Service Orchestrator
SOA	Service Orchestrator Adaptor
SRA	Service Resource Alternatives
SSP	Slice Specification Processor
TNS	Total Number of servers
TS	Total services
uRLLC	ultra-Reliable and Low Latency Communications
vCPE	Virtual Customer Premises Equipment
VIM	Virtual Infrastructure Manager
vRAN	Virtual Radio Access Network
WIM	Wide Area Network Infrastructure Manager

Executive Summary

The document starts summarizing seven scenarios. These scenarios are service oriented and are used to understand the NECOS platform requirements in order to support the deployment of such services. The virtual Radio Access Network (vRAN) scenario takes advantage of the virtualization trends starting to be considered for mobile networks, specifically in the radio access components to facilitate 5G services like uRLLC, eMBB or mMTC. The 5G Services scenario is properly focused on the requirements to provision the above-mentioned types of services. The Virtual Customer Premises Equipment scenario claims to address many issues related to the current model by turning the Customer Premises Equipment into a very simple standard device while moving the network functions to a cloud infrastructure. The Network Slicing for Touristic Content Distribution is meant for offering tourist information facilities to those visitors aiming at enriching their visiting experience, by offering state-of-the-art location-aware cultural content available in the cities. The Multi-Domain Network Slicing for Next Generation Touristic Applications is oriented to enable novel technologies in these contexts, like augmented reality applications that offer even enhanced travelers 'experience. The Network Slicing for Smart Cities Data Content Distribution scenario addresses a public body responsible for enhancing metropolitan authorities' interaction with civilian population in a metropolitan area. Finally, the Network Slicing for Metropolitan Integrated Monitoring scenario aims at leveraging the information flows between distinct public agency teams and major staff authorities.

This document progresses by providing the results of a refinement process of the requirements that were initially reported in the deliverable D2.1. The refinement methodology consisted in correlating the initial sets of requirements (both functional and non-functional) with the workflows that were elaborated in WP5. These workflows are coping with the main processes in the slice lifecycle and specifically on slice creation, decommission, monitoring and elasticity (upgrade and downgrade). These workflows discovered more in-depth requirements, moving towards a set of scenario-agnostic requirements that are able to support key-functionalities of NECOS.

Another contribution of this document is the refinement of the NECOS stakeholders. An initial analysis of the NECOS stakeholders was presented in D2.1. Based on the business models provided by the 3GPP, NGMN and TM forum, three main groups of roles were highlighted. Later on, once the NECOS architecture reported in D3.1 was designed, a more specialized stakeholders' model was proposed to support the Slice as a Service approach. The model contains one stakeholder associated to each one of the main building blocks of the architecture, namely the NECOS (LSDC) Slice Provider, the Marketplace Provider, and the Resource Providers. These three entities are supporting the NECOS Tenant who is making use of the Slice as a Service approach.

The last section of this document is the techno-economic analysis, where NECOS represents an ecosystem where multiple actors or stakeholders interact with each other to provide an integral service (i.e., a slice) to different tenants. The analysis carried out provides both analytical and experimental results to characterize the behavior of the NECOS federation. Those results support some potential business models, which are described in the final part of the analysis.

1. Introduction

The NECOS project aims to address the challenging problem of realizing slicing in cloud networking - native integration of cloud computing and advanced networking -environments in a lightweight manner. This is aimed to be achieved in a unique manner, characterized with the following key properties: being service-agnostic (but adapting the slices to the desired service characteristics), automating the process of optimal cloud configuration in multi-domain federated environments, and providing a uniform management with a high-level of automaticity.

This deliverable is based on the two use cases that were specified in deliverable D2.1, namely the Telco Cloud and the Multi-Access Edge Computing (MEC). These use cases act as platforms supporting seven scenarios, controlled and managed through the NECOS platform, leveraging on the concept of slicing as a form of segregating multiple services in the same (federated) cloud computing substrate.

Key on the evolution of the networks and cloud environments is the concept of multi-tenancy, enabling novel network slicing ideas on top of the existing telecom networks. The paradigms of network virtualization, mainly based on the Network Functions Virtualization (NFV) approach, and network programmability through Software Defined Networking (SDN), have tremendously fostered this evolutionary view, appearing as tools leveraging the implementation of slicing. However, these two paradigms are not sufficient for network slicing. Especially for the latter, there is a prevalent NFV-centric view in the industry, which is not necessarily satisfying all the cases and scenarios. Overloading the NFV orchestration and management artefacts with slicing mechanisms could lead to inefficient cloud or network partitioning. NECOS intends to address such a gap with a new slicing architecture that is valid for multiple scenarios and a wide-range of requirements, i.e., a first relevant analysis is described here.

The focus of WP2 is to describe the use cases and associated service scenarios, the requirements deriving from them, and their associated business models. This document provides the final results of this work package that ends with this deliverable.

1.1. Structure of this document

This deliverable summarizes the work performed during the lifetime of WP2. The structure reflects the main aspects that have been treated in depth after the release of D2.1. The work during that period have been focused on having a better understanding of the requirements that the NECOS platform should satisfy for supporting the scenarios of usage defined in D2.1, as well as to perform the techno-economic analysis of the NECOS as a system. These main aspects are complemented with some update of the previous work in D2.1.

First, a brief refinement on NECOS use cases and scenarios is provided in Chapter 2, in order to reflect the latest considerations with the execution environments and the usage of them that will be also taken by other work packages in the project. Chapter 3 provides an in-depth analysis of the requirements that the NECOS platform has to satisfy, consequence of identifying the different workflows needed for accomplishing the scenarios identified as relevant in the project. Chapter 4 details the refinements produced for the NECOS stakeholders, as evolution of the primary approach followed in D2.1. The techno-economic analysis is reported in Chapter 5, which focuses on the business analysis of the project proposition. Finally, Chapter 6 summarizes the work and provides some remarks and an outlook for future steps and evolutions in the project implementation.

1.2. Contribution of this Deliverable to the project and relation with other Deliverables

The objectives covered by WP2 are as follows:

- **O2.1** To define realistic and challenging use cases, driven from industry, using relevant situations for evaluating the NECOS concepts.
- **O2.2** To explore possible business roles and models in different ecosystem scenarios, and evaluation of the incentives, risks and opportunities related to the adoption of the NECOS solution for all the relevant stakeholders
- **O2.3** To identify general requirements for the NECOS platform as derived from specific requirements for selected use cases and applications

This deliverable provides an update of use cases and scenarios as well as a refinement of the requirements with focus on NECOS platform, addressing objectives O2.1 and O2.3. Additionally it provides experimental and analytical studies related to the relation between actors and stakeholders, covering the techno-economic dimension in the project, and addressing objective O2.2.

1.2.1. Incremental value provided by this deliverable with respect to D2.1

The main incremental work refers to the techno-economic analysis. Both analytical and experimental studies are reported, deriving from them some business guidelines with respect the model to be followed by the cloud providers forming NECOS federation in the provision of Slice as a Service.

1.2.2. Relation with other deliverables

This deliverable will feed the other work packages with refined view of use cases, scenarios, requirements and economic dimension. Specifically, this document is expected to influence D3.2, on the refinement of the NECOS architecture, and D4.2, on the full specification of the information model applicable for soliciting slices in NECOS.

2. Refinement on use cases and scenarios

This section provides a refined view of the use cases and scenarios initially considered in D2.1. The relevance of the use cases and the scenarios of applicability is clear since they are used to derive the requirements that the NECOS solutions has to comply with.

2.1. Overview of NECOS use cases

NECOS primarily identified the Telco Cloud and the MEC as use cases for defining a cloud federation architecture supporting slice as a service. These two use cases are tacking traction in the industry with development of standards related to them and increasing commercial offerings.

From the Telco Cloud perspective, a number of initiatives are progressing, such as the CORD efforts in ONF[1] and the CloudCO approach in BBF[2], as reported in D2.1. The architectures there defined continue evolving and it is expected that could be incorporated in commercial offerings in short-term.

Similarly, the edge computing environment is expanding, multiplying the number of solutions. Additional to ETSI MEC some other initiatives like EdgeX Foundry[3], Facebook's TIP WG on Edge Computing[4], and Akraino[5], to mention a few of them are flourishing.

All of these initiatives reinforce the scope of NECOS as cross-over slicing solution that could potentially be deployed in order to federate such different options in both Telco Cloud and MEC.

On top of the use cases, NECOS has defined some scenarios of applicability, as reported initially in D2.1 and updated in this deliverable, which can use such kind of execution environments. The scenarios have permitted to provide the notion of service to be run on the slice provided by the system. Such services impose some needs to the platform, which basically help to define requirements are reported later on in the following section.

2.2. Refinement on scenarios

At the time of the D2.1 writing, four scenarios were created to elicitate the initial requirements of the NECOS platform and to showcase its application over different network contexts. These scenarios have grown and started to generate sub-scenarios that share common features. In the following, we summarize the scenarios and sub-scenarios present in D2.1:

- 5G Networks
    - 5G Networks
        - It has all the textual elements to become a scenario, including specific requirements (vRAN);
    - 5G Services
        - It has all the textual elements to become a scenario, including specific requirements (5G);
- vCPE
    - It is consistent, not presenting sub-scenarios;
- Touristic Services
    - Network Slicing for Touristic Content Distribution

[1] https://www.opennetworking.org/cord/
[2] https://www.broadband-forum.org/cloudco
[3] https://www.edgexfoundry.org/
[4] https://telecominfraproject.com/edge-computing/
[5] https://www.akraino.org/

        - It has all the textual elements to become a scenario, including specific requirements (Touristic - CD);
    - Multi-Domain Network Slicing for Next Generation Touristic Applications:
        - It has all the textual elements to become a scenario, including specific requirements (Touristic - APP);
- Emergency Scenario
    - Network Slicing for Smart Cities Data Content Distribution:
        - This sub-scenario is strongly coupled with the Emergency Scenario. It has almost all the textual elements to become an independent scenario, only the requirement set is missing since this one shares the same set of requirements of the Emergency scenario;
    - Network Slicing for Metropolitan Integrated Monitoring
        - It has all the textual elements to become a scenario, including specific requirements of the scenario (Emergency);

In order to increase the number of showcases for the NECOS platform, and to have a consolidated version to be referenced by other deliverables, we refactored, based on the previous analysis, the initial four scenarios from D2.1 in seven scenarios. In general, we rewrote the text to decouple it from the original scenario and updated it by using feedback from other work packages. The seven scenarios are introduced as following and the complete refactoring is presented in Appendix A.

2.2.1. Virtual RAN (vRAN)

The virtual RAN (vRAN) scenario takes advantage of virtualization trends for providing network functions associated to the RAN and can also be extended to support different service providers. In this context, vRAN presents potential benefits for multiple stakeholders as Network Connectivity Providers and tenants that have an infrastructure to provide localized or contextualized services. Additionally, this scenario involves the sharing of a common infrastructure among tenants with different needs.

2.2.2. 5G services

This scenario addresses the requirements of Network Operators to adapt existing networks in order to be able to provide forthcoming 5G services such as the enhanced Mobile Broadband (eMBB), that encompasses the challenge of providing an unprecedented volume of data delivery, associated with e.g., high-definition video sharing; the massive Machine-Type Communications (mMTC) that focuses on applications where a large number of IoT devices, such as sensors, collectively creates a significant data volume passing through the network; and the ultra-Reliable and Low Latency Communications (uRLLC), which refers to services that need extremely low end-to-end latency, such as Tactile Internet, Interactive Gaming, Virtual Reality, Automotive, Industry, and Automation.

2.2.3. Virtual Customer Premises Equipment (vCPE)

This scenario exploits the Fixed Telco environments targeted for the use of cloud capabilities such as a virtual Customer Premises Equipment (vCPE). The virtual CPE enables the evolution of Central Offices by applying the key concept of Telco Clouds: the integration of both non-virtualized equipment (bridge CPE) and virtualized elements (virtual network functions), enabling new services to be deployed in an automated fashion. The vCPE services can also be deployed at the Edge Cloud, near to the end-user, in order to reduce latency and improve the user experience.

2.2.4. Network Slicing for Touristic Content Distribution

This scenario addresses the need of a Metropolitan Tourist Centre (MTC) for delivering location-aware touristic content to a large number of visitors in a high-profile Metropolitan Area. In order to achieve the task, the MTC relies on the Slice-as-a-Service model to set up an elastic content delivery network (CDN), where the combination of edge and core cloud servers allows the CDN to adapt to the dynamic patterns of visitor demands. In order to be effective, such a CDN must target areas with a high concentration of visitors, such as public transport vehicles (buses, subway, etc.) and city tourist attractions (e.g., squares, museums), possibly taking advantage of public Wi-Fi infrastructure.

2.2.5. Multi-Domain Network Slicing for Next Generation Touristic Applications

Augmented Reality (AR) applications are expected to enhance travellers' experience and increase the value of the cultural product offered by municipal authorities. However, annotation of the device's camera video is a CPU intensive task, that might not be feasible due to visitor's mobile device resource constraints (e.g. computing capacity, battery, etc.). In this specific scenario, the MTC AR service architecture takes advantage of the slice-as-a-service approach and service function chaining. CPU intensive tasks are delegated to geographically distributed edge clouds, residing near high profile areas, which in turn communicate with core cloud servers to deliver such next-generation services.

2.2.6. Network Slicing for Metropolitan Integrated Monitoring

This scenario addresses a Command, Control and Communications (C3) Public Safety Centre (PSC), responsible for enhancing incident management and resolution to a significant number of first responders (e.g., police department, fire & rescue services, emergency medical services, and public works), as well as their interaction with civilian population in a metropolitan area. The C3 PSC aims at leveraging the information flow level between citizens, responders, and agencies, by quickly offering the ability of receiving, correlating and sharing information.

2.2.7. Network Slicing for Smart Cities Data Content Distribution

This scenario addresses the need to integrate different infrastructure providers in a city in order to provide data content delivery in a Smart City through the use of different infrastructure providers. It also presents how the NECOS platform could fit the aforementioned requirements due to its distributed and federated architecture, and the natural QoS feature present in the slice that enables the end-to-end SLA according to the user needs.

3. Requirements analysis

NECOS considers two Use Cases, MEC (Multi-access Edge Computing) and Telco Cloud, from which four scenarios were initially defined in deliverable D2.1 and restructured in the seven scenarios presented in Section 2 of this current deliverable.

The first set of requirements of NECOS was derived from the four scenarios in deliverable D2.1, using a scenario-driven methodology. Basically, deliverable D2.1 describes the functional and nonfunctional requirements in a per scenario basis. According to this methodology, some requirements were present in more than one scenario and they were highly related to service aspects expected to run in each scenario.

Departing from the scenario-driven methodology to elicit requirements, it was possible to conduct the first round of actions in all the other work packages of NECOS. Deliverable D3.1 details the NECOS Architecture, deliverable D4.1 presents the required APIs and information models, and deliverable D5.1 describes the internal of main modules of the architecture together with key workflows. At this stage in the project, the architecture and APIs support key-functionalities described as workflows for slice creation, decommission, monitoring and elasticity (upgrade and downgrade).

During the process of workflows specification in deliverable D5.1, a new methodology was adopted in order to elicitate more in-depth requirements, moving towards a set of scenario-agnostic requirements that are able to support key-functionalities of NECOS. The methodology adopted was the specification of five workflows using BPMN (Business Process Model and Notation). The workflows are slice creation, slice decommission, monitoring, elasticity upgrade and elasticity downgrade. Each one of the workflows were assigned to two specialists within NECOS, that were asked to individually present a step-by-step list of, so called, tasks required to perform the respective workflow, considering any scenario of D2.1 they prefer.

Once the lists of tasks to realize a workflow were presented by the two specialists, the second activity was to check whether both lists could easily be converted in a single complementary list of tasks. For all the five workflows it was possible to easily converge the independent lists of tasks in the five consolidated lists of tasks.

From the consolidated lists of tasks, the specialists were asked to assign such tasks to architectural modules present in the NECOS Architecture of D3.1, also considering the APIs of deliverable D4.1. After this per-module-assignment of tasks, the workflows presented in D5.1 were created using BPMN.

From the five BPMN diagrams it was possible to perform a cross-check of tasks among workflows, standardizing tasks per module, which were present in more than one workflow. As a result, we were able to consolidate the functional requirements per NECOS architectural module, all of them describe later in this section. Such a new set of requirements is essential for the second-year activities in all work packages of NECOS.

3.1. Functional Requirements

The output generated from the methodology explained above was a list of requirements, described by different specialists from this work package, grouped by NECOS main functionalities. The last task executed by this process was to group these requirements together, organizing them by components, i.e., the element that must implement the described action, in order to identify similar functionalities and duplicated actions. In the end, the list of requirements decreased around 50%, since many functionalities are very similar and can be presented here as a single functional requirement. Afterwards, during the development, these requirements can be split into implementation tasks that shall be developed to fulfill the actions described.

The functional requirements that will be presented in the following shall be consumed by other WPs during the development and validation of the NECOS Platform. For each requirement described we also provide a brief analysis on related Cloud-to-Cloud and Client-to-Cloud APIs, which should be implemented by the component or be called during the execution of the associated actions. For requirements that could not be mapped to existing APIs from D4.1, we provide a suggestion for the elaboration of new APIs.

3.1.1. Slice Specification Processor

The Slice Specification Processor (SSP) is a fundamental component in the interaction between Tenants and the NECOS Slice Provider, acting as the entry-point for all slice creation process. The SSP is responsible for processing different types of slice specifications into a slice request that is sent to the Slice Builder. The SSP requirements are the following:

FR-1.1 - Creation of PDT messages from slice descriptions

The SSP shall be able to create Partially Defined Template (PDT) messages that define the general requirements of a slice. These messages are created based on different types of slice specifications provided in the tenant's request. The PDT messages will act as input to the resource discovery procedure, triggered by the Slice Builder.

Workflows

This requirement was originated from the Slice Creation and Elasticity Upgrade workflows.

APIs

The related APIs that shall be implemented by the SSP are:

- *create_slice (Slice Specification, [Start Time], [End Time]): Slice ID*
- *create_slice (Slice Requirements): Slice ID*
- *create_slice (Service Specification): Slice ID*

FR-1.2 - Provide slice creation interface

The SSP shall implement an interface that allows the Tenant to request slices based on different types of slice specifications. This interface shall reflect different levels of abstraction of a slice. The SSP component is the NECOS Slice Provider entry-point for the slice creation process.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

The related APIs that shall be implemented by the SSP are:

- *create_slice (Slice Specification, [Start Time], [End Time]): Slice ID*
- *create_slice (Slice Requirements): Slice ID*
- *create_slice (Service Specification): Slice ID*

FR-1.3 - Provide slice management interface

The SSP shall provide interfaces for the management of running slices. These management operations shall be forwarded to the Slice Resource Orchestrator that will be responsible for starting the related operations. The management operations are: slice modification, slice removal and different types of queries.

Workflows

This requirement was originated from the Slice Decommission, Elasticity Upgrade and Elasticity Downgrade workflows.

APIs

The related APIs that shall be implemented by the SSP, and presented to the tenant domain, are:

- *delete_slice (Slice ID)*
- *get_slice (Slice ID): Slice Specification*
- *get_slice_parts (Slice ID): Slice Part ID [ ]*
- *get_slice (Slice ID): Slice Specification*
- *get_element_handle (Element ID): Element Management URL*
- *add_element (Slice Part ID, Element Specification)*
- *delete_element (Element ID)*
- *update_slice (Slice ID, [Slice Specification], [Service Specification])*
- *add_resources (Slice ID, Resource Descriptor)*
- *delete_resources (Slice ID, Resource Name)*

3.1.2. Slice Resource Orchestrator

The Slice Resource Orchestrator is a component that aggregates many operations regarding the slice instantiation and maintenance. It takes care of the slice lifecycle management by continuously checking if the allocated slice satisfies the requirements that were initially requested by the tenant. It provides interfaces for managing the running slices and supporting operations such as slice modification and decommission. It is also responsible for the allocation of resource elements that are required for the instantiation of the service instances. The main functional requirements that shall be implemented by the SRO are the following:

FR-2.1 - Management of connections between slice parts to have a end-to-end slice

The management among slice parts must be performed in order to ensure the existence of an end-to-end slice. Each DC Slice Controller and WAN Slice Controller has a slice part that must be glued together with one another to have an end-to-end slice.

Workflows

This requirement was originated from the Slice Creation and Elasticity Upgrade workflows.

APIs

It was identified the need of an API that shall be called by the Slice Resource Orchestrator, in order to manage connection among slice parts. This API should be handled at infrastructure provider level:

- *connect_slice_parts (Slice Part ID []);*
- *disconnect_slice_parts (Slice Part ID []);*

FR-2.2 - Interact with Slice Database for query and to provide slice updates

The Slice Resource Orchestrator shall interact with Slice Database to retrieve information about slices and its slice parts for management operations such as the elasticity procedure, where it can check whether the slice can be expanded or shrunk.

Workflows

This requirement was originated from the Slice Creation, Slice Decommission, Elasticity Upgrade and Elasticity Downgrade workflows.

APIs

Since the interaction is only with an internal component, there are no APIs associated to it.

FR-2.3 - Continuously update IMA regarding VIM, WIMs and Monitoring pointers

The updating of the IMA resources should take VIMs, WIMs and Monitoring pointers into account, as these elements are aware of the available resources.

Workflows

This requirement was originated from the Slice Creation, Slice Decommission, Elasticity Upgrade and Elasticity Downgrade workflows.

APIs

Since the interaction is only with an internal component, there are no APIs associated to it.

FR-2.4 - Provide operational functions for slice management

The Slice Resource Orchestrator shall provide functions for management of slices that allows, for example, decommissioning and modifying slice resources. These functionalities can be triggered by the tenant or by the component itself.

Workflows

This requirement was originated from the Elasticity Upgrade and Elasticity Downgrade workflows.

APIs

The related APIs that should be called by the Slice Resource Orchestrator are:

- *add_element (Slice Part ID, Element Specification)*
- *delete_element (Element ID)*
- *update_slice (Slice Part ID, Slice Part Descriptor)*
- *get_slice_part_elements (Slice Part ID): Slice Part Element ID []*
- *get_element_handle (Element ID): Element Management URL*

FR-2.5 - Request slice parts removal

The Slice Resource Orchestrator shall manage the removal of slice parts from DC Slice Controller and WAN Slice Controller, originated from elasticity downgrade and decommission processes.

Workflows

This requirement was originated from the Slice Decommission and Elasticity Downgrade workflows.

APIs

The related APIs that should be called from Slice Resource Orchestrator are:

- *request_slice_part (Slice Part Descriptor, Slice Part Requirements): Slice Part ID*
- *delete_slice_part (Slice Part ID)*

FR-2.6 - Process slice monitoring information provided by the IMA

The IMA provides a uniform abstraction layer above the heterogeneous VIM/WIM and monitoring subsystems that are part of an end-to-end slice. The Slice Resource Orchestrator shall be able to process metrics associated to a slice, provided by this monitoring infrastructure. The Slice Resource Orchestrator shall also have mechanisms for detecting loss of communication within slice parts that can be associated to an infrastructure failure or with a temporary connectivity issue.

<u>Workflows</u>

This requirement was originated from the Elasticity Downgrade and Elasticity Upgrade workflows.

<u>APIs</u>

The associated API that should be called by the Slice Resource Orchestrator is:

- *get_VNFs_info (VNF_ID [], Infrastructure Manager Handle): VNF status information*

FR-2.7 - Evaluate the need of upgrade or downgrade of resources within a slice

The evaluation of upgrade or downgrade needs is done from the monitoring that is carried out in the infrastructure. If necessary, the slice will be expanded or shrunk to meet the requirements set for it.

<u>Workflows</u>

This requirement was originated from the Elasticity Downgrade and Elasticity Upgrade workflows.

<u>APIs</u>

Since the requirement is an internal process, there are no APIs associated to it.

FR-2.8 - Evaluate requests for upgrade or downgrade of resources within a slice

Slice upgrades and downgrades requests, originated from tenant's domain, should be evaluated by the Slice Resource Orchestrator in order to identify if there are available resources to meet the request and if it does not violate any previously policy established during the service creation.

<u>Workflows</u>

This requirement was originated from the Elasticity Downgrade and Elasticity Upgrade workflows.

<u>APIs</u>

The associated APIs that shall be implemented for this requirement are:

- *add_resources (Slice ID, Resource Descriptor)*
- *delete_resources (Slice ID, Resource Name)*
- *add_element (Slice Part ID, Element Specification)*
- *delete_element (Element ID)*
- *update_slice (Slice ID, [Slice Specification], [Service Specification])*

FR-2.9 - Selection of resources that will be allocated or freed in the elasticity process

The Slice Resource Orchestrator is the component responsible for selecting resources that will be allocated and deallocated during the elasticity upgrade and downgrade processes.

<u>Workflows</u>

This requirement was originated from the Elasticity Downgrade and Elasticity Upgrade workflows.

<u>APIs</u>

Since the requirement is an internal process, there are no APIs associated to it.

FR-2.10 - Request the addition or removal of resources within slice parts

The addition and removal of resources from slice parts must be performed by the Slice Resource Orchestrator in order to meet vertical elasticity demands.

<u>Workflows</u>

This requirement was originated from the Elasticity Downgrade and Elasticity Upgrade workflows.

APIs

The associated API that should be called by the Slice Resource Orchestrator is:

- *update_slice (Slice Part ID, Slice Part Descriptor)*

FR-2.11 - Provide an elasticity process that prioritizes vertical elasticity over a horizontal approach

The Slice Resource Orchestrator shall have an elasticity mechanism that prioritizes the vertical elasticity approach over the horizontal one. The vertical elasticity shall not demand slice topology changes that may be needed in the horizontal one.

Workflows

This requirement was originated from the Elasticity Downgrade and Elasticity Upgrade workflows.

APIs

The related APIs that can be called by the Slice Resource Orchestrator are:

- *request_slice_part (Slice Part Descriptor, Slice Part Requirements): Slice Part ID*
- *delete_slice_part (Slice Part ID)*
- *add_element (Slice Part ID, Element Specification)*
- *delete_element (Element ID)*
- *update_slice (Slice Part ID, Slice Part Descriptor)*

FR-2.12 - Implement horizontal elasticity by requesting the creation or removal of slice parts

The horizontal elasticity implementation should be performed creating or removing slice parts in DC Slice Controller and WAN Slice Controller. This procedure shall trigger the addition of new slice parts through the interaction with the Slice Builder, similar to the procedure of creating a slice.

Workflows

This requirement was originated from the Elasticity Downgrade and Elasticity Upgrade workflows.

APIs

The related APIs that can be called by the Slice Resource Orchestrator are:

- *request_slice_part (Slice Part Descriptor, Slice Part Requirements): Slice Part ID*
- *delete_slice_part (Slice Part ID)*

3.1.3. Slice Builder

The Slice Builder is responsible for building the end-to-end slice, composed of slice parts located at different domains. The Slice Builder starts the slice creation process by searching for suitable resources in the Marketplace, by communicating with the Slice Broker. After receiving and analyzing different resource alternatives, the Slice Builder defines the final slice specification and starts the instantiation of the slice. The instantiation is followed by the activation, by contacting both the corresponding DC and WAN Slice Controllers. The Slice Builder is also responsible for handling requests from Slice Resource Orchestrator in order to discover additional resources for augmenting slices at runtime.

FR-3.1 - Analysis of specific policies and rules for slice requests

The Slice Builder shall be able to receive PDT messages with slice requests from the SSP and analyze them regarding specific policies and rules before starting the slice resource discovery process.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

The related APIs that called at the SSP are:

- *create_slice (Slice Specification, [Start Time], [End Time]): Slice ID*
- *create_slice (Slice Requirements): Slice ID*
- *create_slice (Service Specification): Slice ID*

FR-3.2 - Request resources for slice parts

The Slice Builder shall be able to request resources for slice parts, by communicating with the Slice Broker, and by sending PDT messages through the Marketplace interface. This operation is part of the resource discovery process described in D3.1 and D4.1.

Workflows

This requirement was originated from the Slice Creation and Elasticity Upgrade workflows.

APIs

The related API that is called by the Slice Builder is:

- *locate_slice_resources (Partially Defined Template): Service Resource Alternatives*

FR-3.3 - Provide different mechanisms for defining the final slice specification

The Slice Builder shall provide different mechanisms to define the final slice specification, comprising reasoning mechanisms for automated selection based on slice request, and queries for tenant's approval/selection.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

There are no APIs related to the final slice specification. A new API is proposed to be implemented in the tenant's domain in order to query him about the selection of the final slice specification:

- *handle_slice_specification_selection(): API called by NECOS Slice Provider during the selection of resource alternatives.*

FR-3.4 - Creation of contracts for resources reservation among resource providers, NECOS platform and tenants

The Slice Builder shall create a common resource reservation contract among the NECOS Slice Provider, the Slice Activator and the Slice Controller before continues with the reservation process.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

There are no APIs associated to this requirement.

FR-3.5 - Perform reservation and activation of slice parts

The Slice Builder shall perform reservation and activation of slice parts, by sending PDT messages to request the proper resources for each relevant Slice Controller (DC and/or WAN).

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

The related APIs that shall be called by the Slice Builder are:

- *request_slice_part (Slice Part Descriptor, Slice Part Requirements): Slice Part ID*
- *activate_slice_part (Slice Part ID): {Slice Part ID, Infrastructure Manager Handle}*

FR-3.6 - Provide slice and slice parts information

The Slice Builder shall provide slice and slice parts information, by sending all slice parts pointers and information to SRO after completing its operations.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

Not applicable since this is a domain's internal operation.

3.1.4. Slice Controller

The Slice Controllers are the components that reside in the infrastructure provider domain and are in charge of creating DC and WAN slices by allocating the related slice resources. The allocation of the required resource can end-up with the deployment of VIM on-demand to latter support the deployment of the services within the slice. The Slice Controller can be classified in DC Slice Controller, which handles allocation of compute and storage resources; and WAN Slice Controller, responsible for instantiating the network resources that connects DC slices together.

FR-4.1 - Support allocation, removal and modification of resources for slice parts

The Slice Controller shall support the following slice part operations:

- allocation: the Slice Controller shall allocate resources for an existing or a new slice part;
- removal: the Slice Controller shall be able to remove existing resources from a slice part. This operation can result on removing the entire slice part;
- update: the Slice Controller shall be able to update existing resources for a slice part, following a new slice part description.

Workflows

This requirement was originated from the Slice Creation, Slice Decommission, Elasticity Upgrade and Elasticity Downgrade workflows.

APIs

The related APIs that shall be implemented by the Slice Controller are:

- *request_slice_part (Slice Part Descriptor, Slice Part Requirements): Slice Part ID*
- *delete_slice_part (Slice Part ID)*
- *get_slice_part_elements (Slice Part ID): Slice Part Element ID []*
- *get_element_handle (Element ID): Element Management URL*
- *add_element (Slice Part ID, Element Specification)*
- *delete_element (Element ID)*

- *update_slice (Slice Part ID, Slice Part Descriptor)*

FR-4.2 - Accept or reject contracts for resource reservation

The Slice Controller shall analyze the contracts and answer if the resource reservation can be fulfilled. These contracts are provided by the Slice Builder during the slice creation process.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

A new API for the Slice Controller is proposed for handling slice contracts for resource reservation:

- *handle_slice_contract*: called to handle contracts among Tenant, NECOS Slice Provider and Resource Provider.

FR-4.3 - Instantiation and removal of VIMs and WIMs for slice parts

The Slice Controller shall support the following operations related to VIMs and WIMs:

- allocation: the Slice Controller shall be able to allocate a new VIM/WIM for a slice part and provide all the information needed to access it;
- removal: the Slice Controller shall be able to remove allocated VIMs and WIMs for a given slice part.

Workflows

This requirement was originated from the Slice Creation, Slice Decommission, Elasticity Upgrade and Elasticity Downgrade workflows.

APIs

There are no APIs associated to this requirement since it is a domain-internal process.

FR-4.4 - Support requests for connecting and disconnecting slice parts together

The Slice Controller shall handle the connection between slice parts in order to glue parts together, having a full end-to-end slice. In the same way, it shall support the disconnection between slice parts by removing parts from a slice. This procedure shall be triggered by the Slice Builder and by the Slice Resource Orchestrator after the creation, decommission or modification of a slice.

Workflows

This requirement was originated from the Slice Creation and Slice Decommission workflows.

APIs

There are no APIs associated to this requirement. It is expected that new APIs need to be defined and implemented to fulfil this operation.

- *connect_slice_parts((Slice Part ID []): glue slice parts together in order to have a full end-to-end slice;*
- *disconnect_slice_parts((Slice Part ID []): disconnect slice parts from a slice.*

FR-4.5 - Process requests for upgrade and downgrade of resources within slice parts

The Slice Controller shall analyze, when requested, the feasibility of a vertical elasticity upgrade or downgrade within a slice part. The slice Resource Orchestrator will issue this kind of request in order to fulfil elasticity demands for a given slice. When available, the vertical scaling shall trigger the allocation/deallocation of resources by communicating with the proper VIM and WIM instances. If the request for resources cannot be fulfilled, the Slice Controller shall return a negative answer to the Slice Resource Orchestrator that will manage alternative ways of modifying the related slice.

Workflows

This requirement was originated from the Elasticity Upgrade and Elasticity Downgrade workflows.

APIs

There related APIs that shall be implemented by the Slice Controller are:

- *add_element (Slice Part ID, Element Specification)*
- *delete_element (Element ID)*
- *update_slice (Slice Part ID, Slice Part Descriptor)*

3.1.5. Infrastructure & Monitoring Abstraction

The Infrastructure & Monitoring Abstraction (IMA) provides a uniform abstraction layer on top of different types of VIMs and WIMs, allowing other components to interact with different infrastructure providers without taking care of implementation details.

FR-5.1 - Process requests for setup, removal and update of relevant adaptors for VIMs, WIMs and Monitoring endpoints

The IMA shall process and confirm requests for setup, removal, and update of relevant adaptors for VIMs, WIMs and monitoring endpoints. These adaptors are specific wrappers that translate generic calls to specific methods in the context of different types of VIMs and WIMs.

Workflows

This requirement was originated from the Slice Creation, Slice Decommission, Elasticity Upgrade, and Elasticity Downgrade workflows.

APIs

There are no APIs associated to the setup of adaptors inside the IMA. This is an internal operation that shall be triggered by the Slice Resource Orchestrator, when needed.

FR-5.2 - Provide slice monitoring metrics

The IMA shall provide slice monitoring metrics to the SRO in an aggregated manner, as well as in a per-slice basis. In order to provide those metrics, the IMA shall interact with VIMs, WIMs and monitoring entry points, collecting metrics from both physical and virtual elements running in the target infrastructure.

Workflows

This requirement was originated from the Elasticity Upgrade and Elasticity Downgrade workflows.

APIs

The related APIs that shall be implemented by the IMA are:

- *get_VNFs_info (VNF_ID [], Infrastructure Manager Handle): VNF status information*

The following API is a new interface proposal for retrieving aggregated metrics to be used by the Slice Resource Orchestrator:

- *get_aggregated_service_info(ElementID, ResourceElementID, InfrastructureManagerHandle)*

FR-5.3 - Provide management operations for deployment of virtual functions

The IMA shall provide an interface for management operations (start, stop, and/or retrieve VNFs), for deployment and usage of virtual function.

Workflows

This requirement was originated from the Slice Creation, Slice Decommission, Elasticity Upgrade and Elasticity Downgrade workflows.

APIs

The related APIs that shall be implemented by the IMA are:

- *start_VNFs (VNF Descriptor, Infrastructure Manager Handle): VNF_ID []*
- *delete_VNFs (VNF_ID [], Infrastructure Manager Handle)*
- *get_VNFs_info (VNF_ID [], Infrastructure Manager Handle): VNF status information*

3.1.6. Slice Database

The Slice Database stores all the information associated to a slice during its lifecycle. It lives in the NECOS Slice Provider and provides the interfaces needed for querying and managing such data.

FR-6.1 - Provide an interface for slice management

The Slice Database shall provide an interface for the management of all slice-related information. The Slice Database is responsible for storing information about the topology of each slice and all the services that runs in each slice part.

Workflows

This requirement was originated from the Slice Creation, Slice Decommission, Elasticity Upgrade and Elasticity Downgrade workflows.

APIs

Since the Slice Database is an internal component of the architecture, there are no APIs associated to it.

FR-6.2 - Provide information about slices, slice parts and services

The Slice Database shall provide an interface for querying information about a slice. This information can be related to the slice topology and its associated services and shall be consumed by the tenant or other components of the architecture.

Workflows

This requirement was originated from the Slice Creation, Slice Decommission, Elasticity Upgrade and Elasticity Downgrade workflows.

APIs

Since the Slice Database is an internal component of the architecture, there are no APIs associated to it.

3.1.7. Slice Broker

The Slice Broker is the component responsible for processing requests for slice parts and for locating resources providers that are candidate of satisfying the requests. It performs a resource discovery process that is composed of two stages, the first one is for discovering DC resource options and the second one is for WAN resource options.

FR-7.1 - Support a search mechanism for requesting resources from infrastructure providers

The Slice Broker shall process slice requests, triggered by the NECOS Slice Provider, and start a mechanism for searching resource offers in the registered infrastructure providers. It shall implement a two-step search mechanism that first queries DC Slice Agents about DC resource offers and analyses their responses in order to identify potential network resource providers. The second step is to query the potential WAN Slice Agents in order to fulfil the connectivity demand for the slice request.

Workflows

This requirement was originated from the Slice Creation and Elasticity Upgrade workflows.

APIs

The related API that shall be implemented by the Slice Broker is:

- *locate_slice_resources (Partially Defined Template): Service Resource Alternatives*

The related API that shall be called by the Slice Broker is:

- *pull_resource_offer (Slice Description): Resource Element ID [], Cost*

FR-7.2 - Capability of identify potential network resource providers based on DC resource offers

As part of its search mechanism, the Slice Broker shall be able to analyse DC resource options, provided by DC Slice Agents, and identify potential WAN Slice Agents that can accomplish the slice request, considering connectivity demands.

Workflows

This requirement was originated from the Slice Creation and Elasticity Upgrade workflows.

APIs

The related API that shall be implemented by the Slice Broker is:

- *locate_slice_resources (Partially Defined Template): Service Resource Alternatives*

FR-7.3 - Provide resource offers in the form of alternative resources

The Slice Broker shall respond to the Slice Builder Partially Defined Template (PDT) message requests with the appropriated Service Resource Alternatives (SRA) messages, which shall contain a list of providers that are offering resources to fulfil the slice requirements.

Workflows

This requirement was originated from the Slice Creation and Elasticity Upgrade workflows.

APIs

The related API that shall be implemented by the Slice Broker is:

- *locate_slice_resources (Partially Defined Template): Service Resource Alternatives*

EUB-01-2017

FR-7.4 - Accept registration of Slice Agents

The Slice Broker shall provide an interface for the registration of Slice Agents to maintain an updated list of potential resource providers. This process should be initiated by DC and WAN Slice Controllers that want to announce their resource offers to the Marketplace.

Workflows

This requirement wasn't originated from any specific workflow.

APIs

The related API that shall be implemented by the Slice Broker is:

- *register_provider (Agent Entry Point): Provider ID, Location*

3.1.8. Slice Agent

The Slice Agent resides at the infrastructure provider and its main responsibility is to answer Slice Broker resource requests, by determining if the slice part request can be satisfied, providing a list of resources options as alternatives. In order to analyses and provide resource offers to the Slice Broker, the Slice Agent needs to constantly communicate with its DC and WAN Slice Controllers, in order to identify the available resources within the domain.

FR-8.1 - Process resource requests and check local availability

The Slice Agent must be able to process PDT messages sent by the Slice Broker, extracting from them all the necessary information like constraints, preferences and resources, in order to initiate the resource discovery process, which involves querying its own DC/WAN controller about the availability of resources.

Workflows

This requirement was originated from the Slice Creation and Elasticity Upgrade workflows.

APIs

The related API that shall be implemented by the Slice Agent is:

- *pull_resource_offer (Slice Description): Resource Element ID [], Cost*

FR-8.2 - Provide resource options as answers for resource requests

After the process of analyzing the availability of resources within the domain, the Slice Agent shall provide an answer to Slice Broker request in the form of resource options and their associated costs.

Workflows

This requirement was originated from the Slice Creation and Elasticity Upgrade workflows.

APIs

The related API that shall be implemented by the Slice Agent is:

- *pull_resource_offer (Slice Description): Resource Element ID [], Cost*

FR-8.3 - Constantly check the availability of resources in the local domain

The Slice Agent shall constantly check the availability of resources in the local domain, by querying its own DC or WAN Slice Controllers, in response of a request received or by constantly pushing the availability of resources to the Slice Broker.

Workflows

This requirement was originated from the Slice Creation and Elasticity Upgrade workflows.

APIs

The related API that shall be implemented by the Slice Agent is:

- *pull_resource_offer (Slice Description): Resource Element ID [], Cost*

The related API that can be called by the Slice Agent is:

- *push_resource_offer (Resource Descriptor): Resource Element ID [], Cost*

FR-8.4 - Registration with the Slice Broker

The Slice Agents shall be able to announce their presence to the Slice Broker, enabling the provider to offer their resources for slice parts.

Workflows

This requirement wasn't originated from any specific workflow.

APIs

The related API that shall be called by the Slice Agent is:

- *register_provider (Agent Entry Point): Provider ID, Location*

3.1.9. Slice Activator

The Slice Activator is a component that resides in Tenant's domain and is the central point of interaction between the Tenant and the NECOS Slice Provider for the management of slices. Its main responsibility is to provide an interface for the creation and management of slices, at different levels of abstraction. The Slice Activator must forward Tenant's requests to the NECOS Slice Provider via Client-to-Cloud APIs and provide the answers back to the Tenant. It shall support different management operations on running slices and handling the interaction with NECOS Slice Provider, in order to accept or reject contracts for resource reservations. The main functionalities that must be supported are the following:

FR-9.1 - Creation of slice description from tenant's slice specification, slice requirements and service descriptors

The Slice Activator shall implement an interface that allows the Tenant to request slices, following the different types of specifications provided by the NECOS Slice Provider. These specifications, presented in deliverable D4.1, reflect different levels of abstraction when requesting a slice: (i) the Slice Specification, (ii) the Slice Requirements, and (iii) the Service Specification. The slice specification must be forwarded to the Slice Specification Processor via Client-to-Cloud APIs, which will be in charge of starting the slice creation process. During the full lifecycle of the slice, the Slice Activator must provide ways of querying the status of the slice request made by the Tenant.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

The related APIs that should be called by the Slice Activator are:

- *create_slice (Slice Specification, [Start Time], [End Time]): Slice ID*
- *create_slice (Slice Requirements): Slice ID*

- *create_slice (Service Specification): Slice ID.*

FR-9.2 - Provide a catalog system to support the description of slice requests

The Slice Activator shall provide a catalog system to support the description of Tenant's requests during the slice specification process. The catalog system will provide templates to help on creation of slice requests in all the proposed abstraction levels.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

Not applicable since this is a domain's internal operation.

FR-9.3 - Start service deployment

After a successful slice creation process, the Slice Activator shall trigger the service deployment process by communicating with the Service Orchestrator, at the Tenant's domain. The Slice Activator shall provide all slice information to the Service Orchestrator, which was received from NECOS Slice Provider at the end of the slice creation.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

Not applicable since this is a domain's internal operation.

FR-9.4 - Support management functions for running slices

The Slice Activator shall implement management interfaces for running slices where the Tenant shall be able to request different types of operations, at different levels of abstraction. Some of the main operations are the following:

- Slice deactivation: will trigger the decommission of a slice and the termination of running services;
- Slice update: based on an updated slice or service specification;
- Slice resources update: by adding/removing resources on/from a specific slice;
- Low-level operations on slice parts: interfaces where the Tenant will be able to modify specific elements of a slice part.

Workflows

This requirement was originated from the Slice Decommission, Elasticity Upgrade and Elasticity Downgrade workflows.

APIs

The related APIs that should be called by the Slice Activator are:

- *delete_slice (Slice ID)*
- *get_slice_parts (Slice ID): Slice Part ID [ ]*
- *get_slice_part_infrastructure_management_handle (Slice Part ID): Slice Part Management URL*
- *get_slice_part_elements (Slice Part ID): Slice Part Element ID []*
- *get_element_handle (Element ID): Element Management URL*

- *add_element (Slice Part ID, Element Specification)*
- *delete_element (Element ID)*
- *get_slice (Slice ID): Slice Specification*
- *update_slice (Slice ID, [Slice Specification], [Service Specification])*
- *add_resources (Slice ID, Resource Descriptor)*
- *delete_resources (Slice ID, Resource Name)*

FR-9.5 - Handle contracts for resource reservation

The Slice Activator shall be able to handle a slice contract analysis for slice requests. The Slice Activator should decide if the contract must be signed or rejected based on the slice specification provided by the Tenant, and the slice configuration selected by the NECOS Slice Provider. In case of disagreement with the provided contract, the slice creation is aborted, otherwise the creation process continues.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

A new API is expected to be implemented by the Slice Activator in Tenant's domain, which will be called by the NECOS Slice provider during the slice creation process:

- *handle_slice_contract*: called to handle contracts among Tenant, NECOS Slice Provider and Resource Provider.

FR-9.6 - Support a mechanism for selection of the final slice specification

During the slice creation process, the Tenant may be asked to select the final slice configuration, among different slice alternatives provided by the Slice Builder. The Slice Activator shall provide a mechanism for selecting a slice configuration and returning back the answer to the Slice Builder, in order to continue or abort the slice creation process.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

A new API is expected to be implemented by the Slice Activator in Tenant's domain, which will be called by the NECOS Slice provider during the slice creation process:

- *handle_slice_specification_selection()*: API called by NECOS Slice Provider during the selection of resource alternatives.

3.1.10. Service Orchestrator Adaptor

The Service Orchestrator Adaptor (SOA) is the component that talks with tenant's Service Orchestrator to provide an interface for deployment and management of a service that is associated to a slice. The SOA component interacts with other NECOS components to trigger different tasks issued by the Service Orchestrator.

FR-10.1 - Process service deployment requests

The Service Orchestrator Adaptor (SOA) shall interact with the Service Orchestrator from Tenant's domain to provide an interface for the deployment of a service. It shall adapt every task into internal calls that will be performed by other components in the NECOS Slice Provider. The SOA should keep the Service Orchestrator up-to-date about the instantiation status of all service deployment tasks.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

The related API that should be implemented by the SOA is:

- *start_service ([Service Specification, Service Name], Slice ID): Service ID*

FR-10.2 - Provide service access and monitoring interfaces to the tenant

The Service Orchestrator Adaptor is the interface between tenants and NECOS Slice Provider, from the perspective of service orchestration and management. In this way, it shall provide interfaces for service access and monitoring, in which tenants will interact with its services.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

The related API that should be implemented by the SOA is:

- *start_service ([Service Specification, Service Name], Slice ID): Service ID*

3.1.11. Service Orchestrator

The Service Orchestrator (SO) is a component that resides in tenant's domain and is responsible for the deployment and management of the services that run into the slices. It also provides interfaces for the tenant to interact with the service and to manage its lifecycle.

FR-11.1 - Provide service deployment within a slice

After a successful slice creation process, the Service Orchestrator shall interact with the Service Orchestrator Adaptor in order to manage the deployment of the service within the slice. The Service Orchestrator will perform the service deployment directly to the slice and will provide to the tenant the interfaces for interacting and monitoring the running service.

Workflows

This requirement was originated from the Slice Creation workflow.

APIs

The related APIs that can be called by the Service Orchestrator are:

- *start_service ([Service Specification, Service Name], Slice ID): Service ID*
- *stop_service (Service ID)*
- *get_service_info (Service ID): Service Status Information*

3.2. Non-Functional Requirements

This section presents the non-functional requirements related to scenarios handled by NECOS solution, which were grouped by characteristics and elicited from the requirements presented in NECOS document D2.1. Standards such as ITU-T IMT-2020, ITU - RM.2083-0, NGMN [1][2][9][13] and related work as 5GPPP, 3GPP, METIS [4][5][6][14], were evaluated to analyze the requirements being addressed for the 5G, IoT and network slicing scenarios.

3.2.1. Service diversity

NFR-1.1 - Network should support diversified services accommodating a wide variety of traffic characteristics and behaviours, to support a huge number and wide variety of communication objects, such as user devices, peripheral devices, sensors, and IoT/M2M devices [1].

NFR-1.2 - Network should support services that have different end-to-end QoS (data rate, reliability, latency, location accuracy etc.) requirements. There may be some latency critical applications while there are others that tolerate long end-to-end latency [1]. The network architecture needs to guarantee the overall QoS of a network slice instance served for both operators and third parties (MVNOs, enterprises, service providers, content providers, etc.) on shared infrastructure [1].

3.2.2. Service Level Agreement

NFR-2.1 - The system must assure all the performance metrics (e.g., bandwidth, latency, CPU) negotiated for each slice.

NFR-2.2 - Low latency. The Virtual service should minimize the impact as much as possible on connection’s latency.

NFR-2.3 - High throughput. The resources allocated to the slice should provide a high throughput when processing network packets, considering a large number of subscribers with high bandwidth.

NFR-2.4 - High availability. The service should operate continuously for a long time, so failures in the slice level should be automatically handled without impacting the service.

The IMT-2020 vision assists the development of various industry sectors [2], introducing the following targets for research and innovation:

- Low latency, i.e. 1ms over-the-air, and high reliability;
- User density with area traffic capacity of 10 Mbps/m^2;
- Peak data rate of 10 Gbps with particular scenarios supporting up to 20 Gbps;
- Service continuity under high mobility with speeds of 500 km/h;
- Connection density with 106 devices per km^2;
- 100 Mbps user experienced data rates for wide area;
- Coverage, three times higher spectral efficiency (i.e., in comparison to 4G).

The 5GPPP [3] brought light into the requirements of 5G through the flagship project METIS introducing the following target network capabilities [4]:

- *Amazingly fast:* A feature that shall enable instantaneous network connectivity for all applications by providing 10 Gbps data rates.
- *Great service in a crowd:* A feature that shall enable a broadband experience, regardless of the user density, assuring a traffic volume of 9 GB/h and a data rate up to 20 Mbps per user.
- *Best experience follows you:* A feature that allows a fixed line network experience for users on the move, with at least 100 Mbps in the downlink and 20 Mbps in the uplink.
- *Super real-time and reliable connection:* A feature that aims at supporting mission-critical machine type communications, ensuring 99.999% reliability and less than 5ms end-to-end latency.

- *Ubiquitous things communicating:* A feature that aims at providing wireless connectivity for sensors and actuators, supporting 300,000 devices per cell, while prolonging the battery lifetime of devices in the order of a decade.

The NGMN [5] also anticipates a number of emerging 5G use cases focusing on:

- *Enhanced broadband access everywhere:* envisions a minimum amount of bandwidth, at least 50Mbps, ensuring a connected global society, via high speed Internet. This asset can serve a default general purpose usage.
- *Enhanced broadband access in dense areas:* provides broadband access with up to 10 Gbps bandwidth in densely populated areas, e.g. stadiums or open-air festivals, enabling multimedia services, e.g. ultra-high definition video streaming.
- *High user mobility:* offers broadband support for mobile users in extremely fast-moving vehicles such as high-speed trains.
- *Massive Internet of Things:* supports broadband access for ultra-dense networks of sensors and actuators, considering devices in need of super low-cost, long range and low power consumption, e.g. providing utility measurements.
- *Extreme real time communication:* assuring ultra-low latency connectivity, e.g. for interactive tactile Internet.
- *Ultra-reliable communication:* provides ultra-low latency [10], reliability and availability of network connectivity supporting, e.g. autonomous driving.
- *Lifeline communication:* supports connectivity in case of natural disasters and emergencies capable to accommodate flexibly a sudden tremendous traffic increase, while assuring resilient connectivity.
- *Broadcast-like service:* provides network connectivity for broadcasting, e.g. news or firmware updates for instance to improve the breaking system of a car or braise up a detected security hole in cars [11].
- *Light-weight communication* [12]: provides network connectivity for supporting essential instantiation, configuration and maintenance service information.
- *Multi-connection:* assures network connectivity for users with different smart devices, e.g. smart glass and smartphones, using multiple access technologies.

3.2.3. Mobility

NFR-3.1 - The network architecture is required to support seamless and consistent user experience while moving across different access networks, and also steer mobile devices to choose the most suitable access technology in a seamless way [1].

NFR-3.2 - The network architecture is required to support enhanced mobility management such as "context-aware mobility" considering device types, application characteristics, etc. [1].

NFR-3.3 - The network architecture is required to support the mobility management aligning with those architectural changes that the core network is envisioned to be a flat distributed network, which is composed of the multiple distributed gateways to cope with traffic explosion and latency requirements of applications [1].

NFR-3.4 - Network should provide mobility that facilitates high-speed and large-scale network in an environment where a huge number of UE can dynamically move across heterogeneous networks [1].

3.2.4. Functional Flexibility and programmability

NFR-4.1 - The network architecture is required to support the capability of creating the dedicated network, i.e., the capability of composing functions on-demand into a dedicated slice, according to the requirement of the 3th party. It is envisioned to be flexible enough for realizing network components as software components and simplifying the process of creating the dedicated network [1].

NFR-4.2 - Network architecture should be designed to have a common core that supports on-demand composition of variety of multiple network slices [1].

NFR-4.3 - Programmability allows third parties to control the allocated slice resources, i.e. networking and cloud resources, via open APIs that expose network capabilities facilitating on-demand service-oriented customization and resource elasticity [13][6][7][8].

NFR-4.4 - High scalability (variable and dynamic number of users), with a short-time response for the deployment on the field. Metrics - Provisioning time and decommission time.

3.2.5. Network-agnostic architecture and technology-agnostic

NFR-5.1 - Network is envisioned to be an access network-agnostic architecture where core network will be a common unified network for emerging new radio access technologies as well as existing fixed and wireless network [1].

NFR-5.2 - The access technology-agnostic unified core network should be accompanied by common control mechanisms, which are decoupled from access technologies [1].

NFR-5.3 - Heterogeneity handling. The service provisioning should be transparent with respect to details of the physical infrastructure, VIM used, load balancing and slice management.

NFR-5.4 - Resource federation and intelligent multi-domain orchestration, to enable novel services that were not possible before.

3.2.6. Isolation of network Slice and virtual resource and security

NFR-6.1 - Network architecture should support virtualization of resources associated with network functions, which should support isolation of any network slice and virtual resource from all others [1]. Isolation is a fundamental property of network slicing that assures performance guarantees and security (to defend network openness to third parties) for each tenant, even when different tenants use network slices for services with conflicting performance requirements [13][6][7][8].

NFR-6.2 - The network architecture is required to provide softwarization capabilities with enhanced performance for wired and wireless network [1].

NFR-6.3 - The network architecture is recommended to provide softwarization capabilities with enhanced performance for mobile access networks [1].

NFR-6.4 - The network architecture is required to support programmability of network functions in data plane for easier provisioning of new emerging service [1].

NFR-6.5 - The network architecture is required to support the separation of control and data plane functions in the network [1].

NFR-6.6 - The network architecture is required to have the capability to meet the service-specific security assurance requirement in a single network slice, rather than the whole network slice [1].

NFR-6.7 - Fairness. Operator must be able to optimize the resource usage without negative impact on any tenant.

3.2.7. Unified intelligent network management

NFR-7.1 - The network architecture is required to create, operate and manage network slice [1].

NFR-7.2 - The network architecture is recommended that network functions are virtualized, and it should support dynamic scale-in/scale-out per operator’s policies [1].

NFR-7.3 - Network should be designed to simplify operations and management of the network with increased complexity due to flexible and extensible network softwarization [1].

NFR-7.4 - Procedures should be automated as far as possible, with well-defined open interfaces to mitigate multivendor interworking problems, as well as interoperability issues [1].

NFR-7.5 - Standardized management protocol is desirable [1].

NFR-7.6 - Also, enhanced end-to-end QoS management and security/privacy models should be designed [1].

NFR-7.7 - Customization assures that the resources allocated to a particular tenant are efficiently utilized in order to meet best the respective service requirements [13][6][7][8].

3.2.8. Distributed architecture

NFR-8.1 - Separation of control and data planes and the enabling technologies are the basis to make the network flexible and extensible [13][6][7][8].

NFR-8.2 - Network should support a highly scalable distributed architecture to avoid signalling congestion and to minimize the signalling overhead for diverse UE/RAT/service requirements [13][6][7][8].

NFR-8.3 - Require the gateways to the core networks expected to be located closer to the cell sites resulting in distributed a network architecture [13][6][7][8].

3.2.9. Reliability and resilience

NFR-9.1 - Network should be designed, operated, and evolved with reliability and resilience, considering congestion and disaster conditions, and to be designed for safety and privacy of their users [13][6][7][8].

NFR-9.2 - The network architecture is required to protect negative impact in one network slice offered by another network slice [1].

NFR-9.3 - The network architecture is recommended to maintain the established QoS of the network slice after creation regardless of the status of other network slices [1]. So failures in the slice level should be automatically handled without impacting the service.

3.2.10. Optimization

NFR-10.1 - The network should provide sufficient performance by optimizing network equipment capacity based on service requirement and user demand [1].

NFR-10.2 - The network should provide dynamic data routing mechanisms that respond to changing conditions of network segments [1].

NFR-10.3 - Network should be designed and implemented for optimal and efficient handling of huge amounts of data [1].

NFR-10.4 - The slice resources should be efficiently utilized, i.e., no over-provisioning of resources.

NFR-10.5 - Fast service deployment. Metrics - Service provisioning time.

3.2.11. Improve energy efficiency

NFR-11.1 - Network should be designed to reduce UE power consumption and to improve energy efficiency in overall network operation. Device-level, equipment-level, and network-level technologies should cooperate with each other in achieving a solution for network energy saving. 5GPPP [3] proposes 100 times more energy-efficient networking, and energy lifetime for sensors to be greater than 10 years.

3.2.12. Automation

NFR-12.1 - Automation enables an on-demand configuration of network slicing without the need of fixed contractual agreements and manual intervention [13][6][7][8].

NFR-12.2 - Such convenient operation relies on signalling-based mechanisms, which allow third parties to place a slice creation request indicating besides the conventional SLA, which would reflect the desired capacity, latency, jitter, etc., timing information considering the starting and ending time, and duration or periodicity of a network slice [13][6][7][8].

NFR-12.3 - Transparent end-user performance. End-user performance transparency, with respect to service usage, i.e. visitors should be unaware of changes due to load-balancing actions (introduction of new VMs, changes in routing, etc.), and experience minimal service delays and interruptions.

NFR-12.4 - Real-time monitoring. Provision of monitored data to the operator and the tenant. Metrics - Monitoring-data availability.

3.2.13. End-to-end

NFR-13.1 - End-to-end is an inherent property of network slicing for facilitating a service delivery all the way from the service providers to the end-user/customer(s) [13][6][7][8].

NFR-13.2 - Such a property has two extensions, *(i)* it stretches across different administrative domains, i.e. a slice that combines resources that belong to distinct infrastructure providers, and *(ii)*, it unifies various network layers and heterogeneous technologies, e.g. considering RAN, core network, transport and cloud [13][6][7][8].

3.2.14. Hierarchical abstraction

NFR-14.1 - Hierarchical abstraction is a property of network slicing that has its roots on recursive virtualization, wherein the resource abstraction procedure is repeated on a hierarchical pattern with each successively higher level, offering a greater abstraction with a broader scope. In other words, the resources of a network slice, allocated to a particular tenant, can be further traded either partially or fully to yet another third player, which relates to the network slice tenant facilitating in this way another network slice service on top of the prior one [13][6][7][8].

NFR -14.2 - The network architecture is required to support that the 3rd parties can create and manage a network slice configuration via suitable APIs, within the limits set by the network operator [1].

EUB-01-2017

4. Refinement on NECOS stakeholders

In this section a refinement on NECOS stakeholders is described. Firstly, the initial NECOS stakeholders approach created in D2.1 is explained. Secondly, final NECOS stakeholder's architecture is showed according to the model explained in D3.1.

4.1. Initial NECOS stakeholder's model proposed in D2.1.

An initial analysis of the NECOS stakeholders was presented in D2.1. Based on the business models provided by the 3GPP, NGMN and TM forum, three main groups of roles were highlighted: infrastructure/resource provider, service provider and customer. Each of these groups were composed of more specialized actors, where a single entity could play different roles.

In NECOS, the most important element is the slice, which is a composition of a partition of connectivity, compute and storage resources within services. On this wise, we could not adopt the business models provide by those associations and it was needed to include new roles.

The NECOS ecosystem stakeholders provided in D2.1 is depicted in Figure 1.

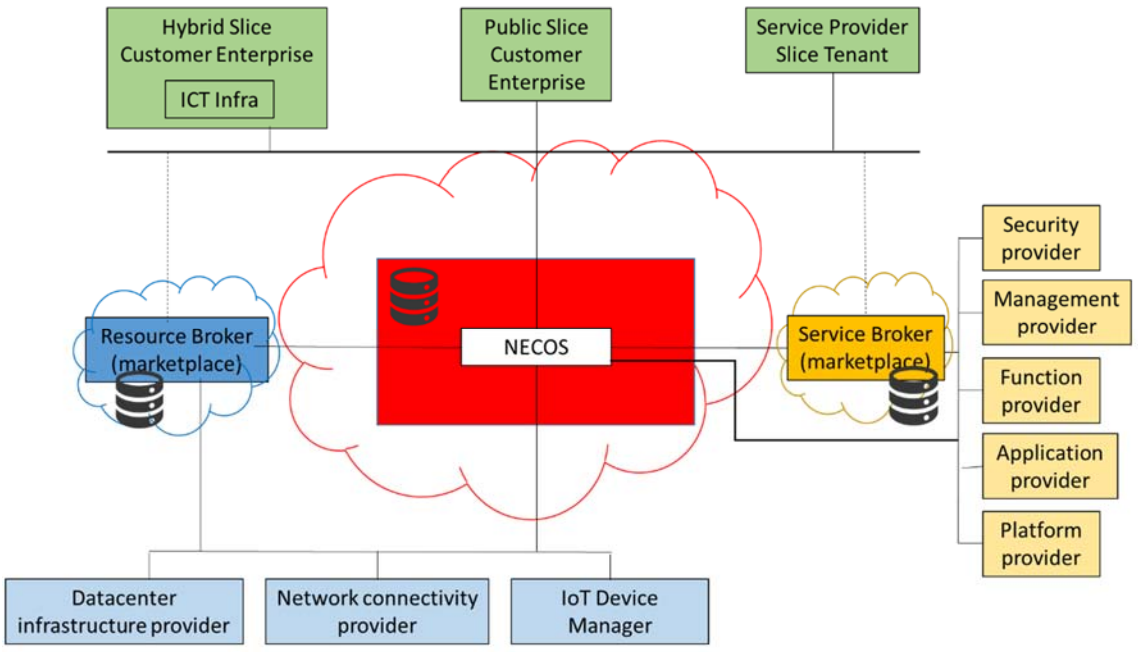

Hybrid Slice Customer Enterprise
ICT Infra
Public Slice Customer Enterprise
Service Provider Slice Tenant
Resource Broker (marketplace)
NECOS
Service Broker (marketplace)
Security provider
Management provider
Function provider
Application provider
Platform provider
Datacenter infrastructure provider
Network connectivity provider
IoT Device Manager

Figure 1. NECOS ecosystem stakeholders in D2.1

The whole stakeholder's ecosystem was classified into four main categories:

- *Slice Provision*: consisted of the entities that offered slice services to customers and tenants (red box in Figure 1), i.e., it was basically the NECOS orchestrator and its internal elements. Elements in this category only existed if both elements in the other categories existed and collaboration or commercial agreements among them are established. The unique stakeholder inside this group was the NECOS system. This entity provided slice services to customers upon requests.
- *Infrastructure Resource Provision*: consisted of the entities that provided physical and virtual resources to slices (blue box in Figure 1). Elements in this category had no dependency on elements in the other categories. That said, they only made sense to exist to serve customers

and tenants, the exception being infrastructure resource owners (not providers) offered idle capacity. Within this category there were four stakeholders: the resource broker provided a resource directory service to NECOS and to Slice Customers to support slice requests or increase/reduction of resources in an existing slice; the datacenter infrastructure provider supplied data center resources to support building of slices; the network connectivity provider offered wired and wireless connectivity resources to support building of slices; and the IoT Device Manager provided shared or exclusive access to IoT devices to support building of slices.

- *Service Provision*: consisted of the entities that provided services to slices (yellow box in Figure 1). Elements in this category had no dependency on elements in the other categories. That said, they only made sense to exist to serve customers and tenants. Within this category, the following stood out: the service broker provided a services directory to the NECOS and to Slice Customer to add to slices or on top of slices; the Security as a Service provided security services (e.g., Intrusion Detection) to slices as a service; the Management as a Service provided management services (e.g., monitoring) to slices as a service; the Function as a Service provided functions (e.g., VNFs) to slices as a service, the Software as a Service provided software applications to slices as a service; and the Platform as a Service provided platforms (e.g., SDN Controller) to slices as a service.
- *Slice Consumption*: consisted of the customers and tenants that consumed slices (green box in Figure 1). Elements in this category did not necessarily depend on the Slice Provision services to exist, but had their needs greatly facilitated by the automation provided by those services. This group included: The Public Slice Consumer requested to NECOS a slice using external resources and services; the Hybrid Slice Customer requested to NECOS a slice using both internal resources and services and external resources and services; and the Service Provider Slice Tenant requested to NECOS a slice using external resources and services.

In this architecture, a Slice Customer requested a slice to NECOS. NECOS attempted to meet the request by first verifying its internal database for available slices ready to use. If no slice met the requirements, then the NECOS checked with the Resource Broker for resources and with the Service Broker for services to build the slice. The slice to be delivered to the Slice Costumer consisted of resources only or both resources and services. For elasticity purposes, a Slice Costumer requested to NECOS or directly to the Resource Broker to add resources to or remove resources from the slice. Likewise, a Slice Costumer requested to NECOS or directly to the Service Broker to add services to or remove services from the slice (e.g., VNF) or from the top (e.g., MaaS) of the slice.

4.2. Final NECOS stakeholder's architecture

As it has been explained in other deliverables, the idea of this project is to develop a specific slicing model based on a process that separate the underlying infrastructure into constituent slice parts and later blend them into a full end-to-end slice. To achieve the aim of service provisioning, it is needed a combination between right abstractions, layering and separation of concerns in the architectural diagram. If this combination is well done, NECOS will create a powerful and flexible Slice as a Service mechanism.

In particular, the Slice as a Service approach provides an adaptable control plane, having features for creating, growing, shrinking, and closing slices, as well as adapting slices at the run-time, while considering service requirements and current cloud resource conditions.

The stakeholder's architecture explained in D2.1 was a preliminary approach to obtain this Slice as a Service model. In that diagram, there was not yet a stakeholder who can give NECOS the needed slices parts to build up a full slice. For this reason, the final stakeholder's architecture grows the preliminary approach.

In this section, we present the architecture that has been devised to support the Slice as a Service approach. The architecture contains three main categories. These are (1) the NECOS (LSDC) Slice Provider (colored in blue), (2) the Marketplace Provider (colored in yellow), and (3) the Resource Providers (colored in green). These sub-systems are provided in order to support the tenants of NECOS who wish to use Slice as a Service. The tenant of NECOS is expected to be an organization that requires slices for running their own services. Figure 2 presents these main elements, how they are grouped and the way they interact with the tenants (colored in red) who use NECOS.

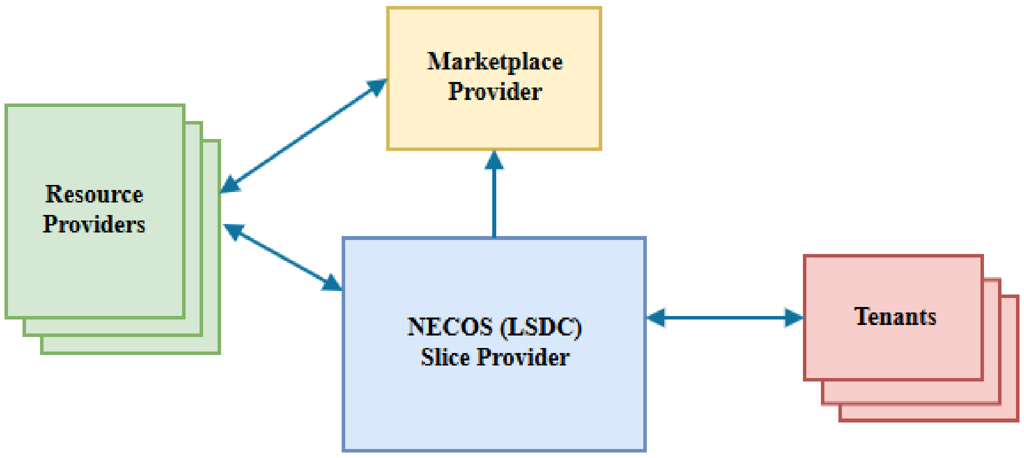


Figure 2. Final NECOS stakeholder's architecture

The details of the NECOS stakeholders' architecture are presented hereunder.

4.2.1. The NECOS Tenant

The tenant of NECOS is expected to be an organization that requires slices for running their own services. An example of such a tenant could be a CDN company. The tenant is not expected to be a small organization that needs a handful of Virtual Machines nor an end-user who wishes to use an online service. Both kinds of user already have multiple options available to them.

The NECOS tenant is depicted in the right part of Figure 2 – it can be an organization that is expected to request to a NECOS Slice Provider the instantiation of heterogeneous resources including computation, storage and network in the form of end-to-end slice, and to run their own Service Orchestrator for deploying and managing their own services on that allocated slice. The tenant is likely to have a range of options for choosing a Slice Provider, and such could be based on factors such as commerce, geography, and contractual obligations.

4.2.2. The NECOS Slice Provider (LSDC)

The NECOS Slice Provider (LSDC) is the sub-system that allows for the creation of full end-to-end Slices from a set of constituent Slice Parts. In NECOS, a Slice looks the same as the full set of federated resources, with the main attribute being that the domains look a lot smaller, as they have been sliced. Therefore, we call the Slice, the LSDC - the Lightweight Software Defined Cloud.

The NECOS Slice Provider (LSDC) receives the requests from the tenants, who can operate on the full infrastructure or choose to interact with Slice a Service providers, using NECOS. When NECOS requests a Slice, it goes out to a specially designed and configured Resource Marketplace (from a marketplace provider) that, based on a Slice Specification, can find Slice Parts across various participating resource providers.

The NECOS Slice Provider (LSDC) oversees combining the Slice Parts that make up a slice into a single aggregated Slice and is responsible for the orchestration of the Slices, including their management at the run-time of their lifecycle. It is also responsible to orchestrate the service elements across the Slice parts that make up the full end-to-end slice. Finally, it is the stakeholder that is responsible for the actual placement and embedding of VMs and virtual links for the services into the resource domains.

4.2.3. The Marketplace Providers

The Marketplace Providers deliver resource marketplaces that provide the way for the NECOS (LSDC) Slice Provider to find the slice parts to build up a slice. Rather than having a pre-determined set of providers that have been configured in a federation, we chose to use a more flexible model of a marketplace from which we could provision slice parts. This was done for two main reasons: (i) in order to build end-to-end slices we need a mechanism to reach and interact with providers in multiple geographic places - including mobile edge and sensor networks - which are often not covered with existing federation approaches; and (ii) slice creation is generally more dynamic than federation agreements, so we need a highly dynamic run-time mechanism for finding providers. We observed that there are online marketplaces for flights and for hotel rooms that provide a good operational model to follow. There is a broker, which can dynamically go to multiple agents and get a collection of offers for the required resource. The end-user can then make a choice as to which offer is best suited. The agents can be a division of the resource provider, or they may be an external aggregator that provides compound offers. From the perspective of the broker and the end-user, the agent just provides offers at a price point with certain conditions. Such an approach fitted with the NECOS Slice as a Service system very well.

The NECOS (LSDC) Slice Provider is responsible to build a full end-to-end multi-domain slice from the relevant constituent slice parts. In order to do so, it must find available resources from the marketplace. This search involves communication with a Marketplace Provider, the entity responsible for contacting the Resource Providers. In the NECOS approach, there is expected to be multiple Marketplace Provider in various locations - maybe multiple per country. Each one will have access to many Resource Providers across many geographical domains, which are able to provide offers for the required slice parts that match a set of request constraints. For example, the slice may need DC slice parts in Spain, Greece, the UK, and in Brazil, with network slice parts that connect the DC slice parts.

4.2.4. The Resource Providers

The Resource Providers are those organizations that can provide the resources required for the slice parts - namely, Data Centre resources in the form of servers, storage, and Network resources. Further resources can be provided by organizations that have Mobile Edge, Sensor Networks, Wireless, etc. Each resource provider will be capable of providing slice parts, which will be part of a full end-to-end slice. The Resource Provider also needs to provision the relevant manager for each slice part, meaning a VIM for a DC slice part, or a WIM for a network slice part.

5. Business analysis

This section covers the techno-economic analysis performed in the project. NECOS represent an ecosystem where multiple actors or stakeholders interact with each other to provide an integral service (i.e., a slice) to different tenants.

The analysis carried out provides both analytical and experimental results to characterize the behaviour of the NECOS federation. Those results support some potential business models which are described in the final part of the analysis.

5.1. General considerations

The NECOS system enables the trading of resources among different actors for the dynamic provision of slices (then following a slice as a service, SlaaS, model). For the consideration of the techno-economic analysis, it is assumed that a Tenant will request some infrastructure capabilities (i.e., compute and networking resources) to a cloud provider, identified as Entry Cloud Provider, since is the provider facing the Tenant demand. This Entry Cloud Provider is part of a broader system that involves a federation with other cloud providers that we refer as Federated Cloud Providers.

More than one cloud provider in the NECOS federation can be involved in the slice as a service provision, depending if the Entry Cloud Provider is able to satisfy the tenant demand with its own resources on the time of arrival. In case this Entry Cloud Provider has not enough resources to satisfy the Tenant request, then it can request additional resources to the federation, in such a way that part of the demand is passed over Federated Cloud Providers. This will be transparent to the Tenant which will make use of the allocated slice as a unified service good.

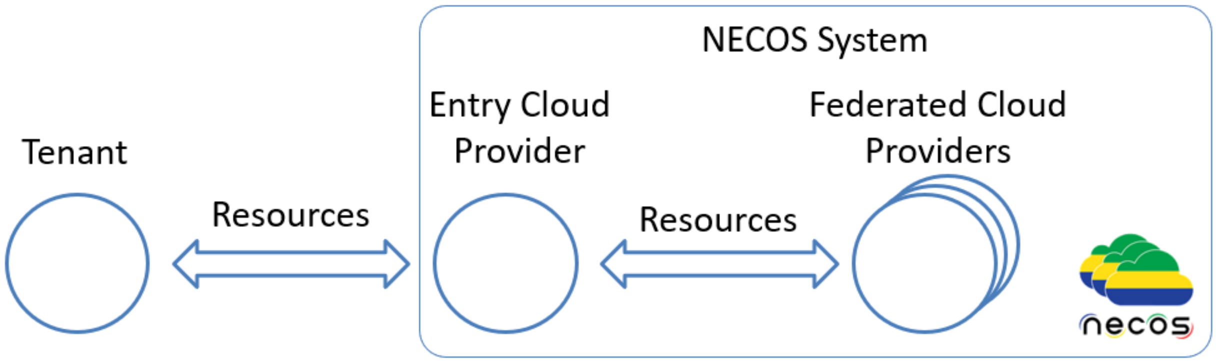


Figure 3. NECOS System diagram

The kind of resources forming the slice to be considered as traded in the NECOS system are detailed in the NECOS information model, as reported in D4.1. These are basically computing resources (CPU, memory, storage) and network resources (bandwidth).

The cloud systems have been approached from a techno-economic perspective in some previous works.

This is the case of [16] that provides an overview of the economics involved in the provision of applications in public cloud infrastructures. The study considers commercial cloud offerings by some real providers detailing different pricing models (subscription, on-demand, etc.). There elasticity is also reported as one of the aspects that can vary the costs (from the tenant perspective) to adapt the cloud offering to the real tenant need.

In [17] a complete survey of economic and pricing models for cloud is provided, including some insight into federation schemas. To exercise a techno-economic analysis a number of variables have to be taken into account such as different pricing models, cost of energy, space, bandwidth, etc., covering both CapEx and OpEx costs. All those considerations can lead to complex modelling and require of sophisticated mathematical artefacts. From all the models there described, the cost-based analysis will be considered in NECOS. Additionally, in order to simplify the analysis but yet obtaining significant insights, only a simple price model and infrastructure cost will be taken into consideration, as explained afterwards.

Interestingly, [18] analytically assess the profitability of a cloud federated scenario. One of the main arguments for it is that the federation reduces the blocking of service requests, then leading to higher utilization rates on the participants of the federation.

The next sub-section introduces the methodology chosen by NECOS to analyse the project architecture from a techno-economic perspective.

5.2. Methodology adopted

In order to analyse the system, two different approaches have been followed. In one hand, an analytical work has studied some variables for delimiting to what extent the federation can generate benefits to its participants. On the other hand, a simulation tool has been developed in MATLAB allowing for understanding the behaviour of the systems for different kind of workloads for the slice being requested by the tenant.

In both cases, the followed approach is a cost-based analysis of the slice as a service in a federated environment. Figure 4 illustrates the concept.

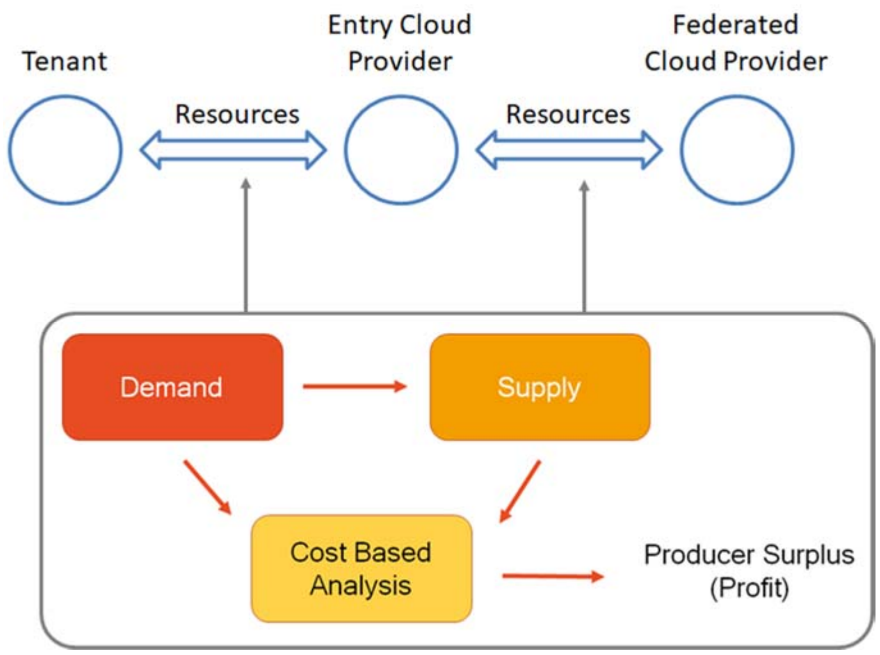


Figure 4. Cost based model

All this analysis is based on the following equation:

$$Benefits = Revenues - Costs \quad (1)$$

In Eq. (1) benefits are the monetary units that NECOS system will earn for offering some resource slices to the tenants; Revenues are the monetary units that the tenants will pay for obtaining the desired slice; and Costs represent the monetary units that the NECOS system providers expend for having the resources to fulfil these slice requests.

For completing the analysis, some references for revenues (i.e., pricing) and costs (i.e., infrastructure costs) have been considered:

- *Revenues*: it is assumed that the NECOS provider pricing will be similar to the ones from current cloud providers existing in the market. An initial search of market prices was carried out among the main infrastructure providers nowadays in the market with the objective of finding providers who price the flexible offering of resources depending on the volume of them needed as well as the amount of time the tenant requires to use them. Following this idea, pricing from [15] is taken to model the revenues.
- *Costs*: another important topic is how to model the infrastructure costs. Generically for the analysis done, it has been considered the usage of servers with 24 cores each, focusing the analysis on the impact of the CPU usage. This is because other parameters (such as RAM or memory) seem to be less relevant as driver for infrastructure costs, as described in [19][LMCM1]. Doing a survey among the main server vendors, the average price of servers with 24 cores is about 4000€, so this is the chosen server price. As public available reference for commercial offering of servers the reader can check [20][LMCM2]. Additionally, it has been considered for the amortization time of a physical server a period of 5 years (as value suggested in [21][LMCM3]), which will be taken as the total simulation time. In this way, the infrastructure cost could be assumed as the total cost of the servers for simplification.

The distinct tenants approaching the NECOS platform will generate a total aggregated demand that has to be honoured by the overall system. Such an aggregate demand will be a composition of individual slice demands showing different characteristics in terms of duration, number of resources needed, arrival time and pattern, etc.

To carry out an adequate techno-economic study of this architecture, the services received in the Entry Cloud Provider are divided into different types according to the expected kind of workload characterizing the service, in order to characterize the system behaviour according to the supported services. Workload characterization is a usual manner of segmenting the demand in cloud environments, as in [22][LMCM4].

In this analysis performed here two services dimensions are used: requested CPUs (in number of CPUs) and workload duration (in number of days). The requested CPUs represent the number of CPUs that each slice needs. Duration is the time that each slice lasts. Therefore, services are split into these 6 different types of services (i.e., types of slices) as showed in Table 1.

Table 1. Services characterization

Services	CPU usage (# CPUs)	Duration (days)
Service 1	2	15
Service 2	4	15
Service 3	8	15
Service 4	2	120

Service 5	4	120
Service 6	8	120

Furthermore, it has been considered that each slice requires some CPUs to be allocated for slice control and management purposes. This assumption has been implemented by adding two CPUs to the number of cores that the service needs to be carried out, independently of the characteristics of the slice. Therefore, these additional cores will not be directly used for generating revenues to the provider but will imply resource consumption, and then, indirectly, costs.

For example, in Figure 5, it is showed the CPU usage in NECOS system for five years, where each column summarizes the CPUs for the minimum number of servers that avoid any rejection of a slice demand during the period. There are three types of CPU there reflected: services CPUs, which are the ones used to run services as demanded by the tenant request, and hence directly providing revenues; management CPUs, which are the ones allocated for slice control and management; and, finally, empty CPUs which represent the vacant resources. Each bar represents the accumulated figures for two months intervals, in which the sum of the three type of CPU usage result the total number of available resources (CPUs) made available by the servers on a daily basis[6].

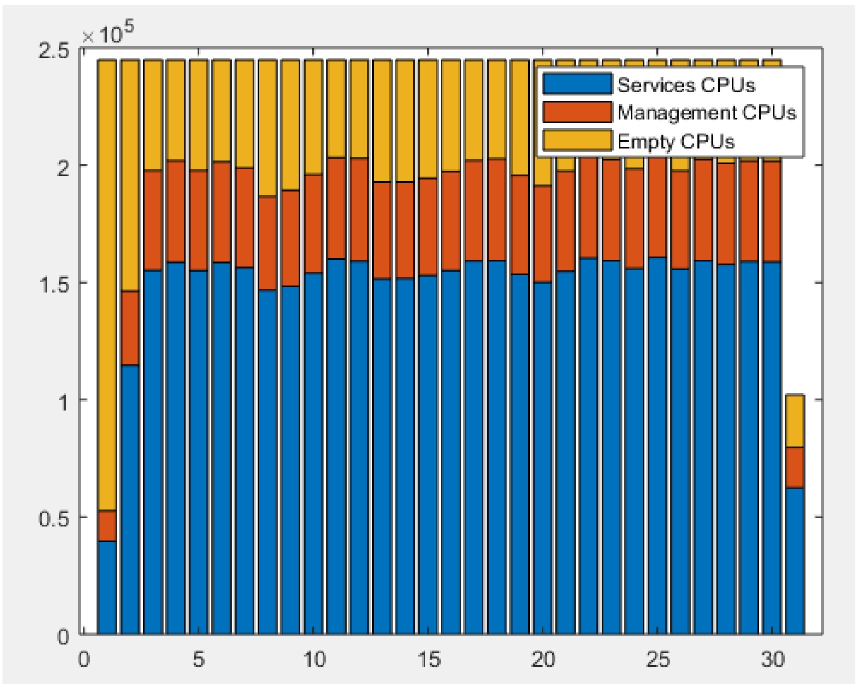


Figure 5. CPU usage in NECOS system

This business analysis is then performed in two ways. First, an analytical study is done, based on the equations extracted from this architecture and then, an experimental study is done, based on the simulations carried out by the developed MATLAB tool.

[6] The simulation calculates in a daily basis the CPUs devoted to services and management as well as the ones vacant. The graph represents the accumulated view each two months. As result, the figure illustrates the percentage of CPUs for each category.

5.3. Analytical study

The analytical study aims to establish a relationship among the main variables involved in the provision of multiple slices in a federated cloud like NECOS. Thus, it is possible to fix some of the variables and determine the implications of some others for guaranteeing sustainability of the federated systems.

The main variables considered in the analysis are the following:

- *ECPS* (Entry Cloud Provider Services): number of service requests (i.e. slice demands) received and served by the entry cloud provider.
- *FS* (Federated Services): number of service requests (i.e. slice demands) received by the entry cloud provider but served by the federated cloud providers due to the scarcity of resources in the entry cloud provider at the time of the request.
- *TS* (Total services): total number of service requests (i.e. slice demands) received by the entry cloud provider (*TS* = *ECPS* + *FS*)
- *PPS* (Price Per Service): it is the price the tenant is willing to pay to the NECOS system (i.e., to the entry cloud provider being part of NECOS) for fulfilling the request. In the analysis it has been considered pricing values as offered today by commercial companies in the market (based on [15], and expressed in €):

$$PPS = (18.24 \; + \; 6.48 \cdot \# \, CPUs) \, \frac{Duration(days)}{30} \tag{2}$$

- *IC* (Infrastructure Cost): it is the price the NECOS members pay for their resources. For the sake of simplicity, it has been considered a common cost for all the participants in the federation.
- *SC* (Server Cost): it is the cost of each individual server.
- *NS* (Number of servers): it is the number of servers that each NECOS participant has.

$$IC = SC \cdot NS \tag{3}$$

- *K*: it is the percentage over the *PPS* that the entry cloud provider retains from the tenant payment when transferring the service (i.e., the requested slice) to a federated provider.
- (1 - *K*): it is the percentage over the *PPS* that a federated cloud provider earns for performing a service.
- *TNS* (Total Number of servers): it is the minimum total number of servers in the NECOS system necessary to not deny any service (i.e., no blockage of service requests).

This analysis has been done from three different points of view: the view of the Entry Cloud Provider, the one of the Federated Cloud Providers (one or more), and the general view of the NECOS system as a whole. This study provides a high degree of freedom to understand the impact of different parameters in the system, according to their values. For instance, in this deliverable, it is provided an equation that indicates the minimum duration (minimum days) a service must endure so that the corresponding entities do not have a negative benefit. It is delivered according to the total number of services that are carried out, the number of available servers that the corresponding entity has, the cost per server, the number of CPUs that the service needs to be performed, and the k, which is the percentage of PPS earned by the ECP if you federate a service.

<u>**Entry Cloud Provider (ECP)**</u>

The benefit of the ECP is composed of the incomes due to the services requests (slices) that can fulfil by its own, plus the incomes due to the incomes for the services which are passed to other providers in the federation for the cases in which the ECP has not enough resources for satisfying the requests.

The second income will be usually a mark-up for the mediation in the process of commercializing the service, being subtracted from the total price of the service (slice) that is not modify for the tenant. Note that the tenant is not (necessarily) aware of the federation, then the tenant is requesting a service at a given price independently if it requires from the federation for allocating the necessary resources.

The benefit in the entry cloud provider is then defined by:

$$\begin{gathered} benefit_{ECP} = revenues_{ECP} - Costs_{ECP} \rightarrow \\ \rightarrow benefit_{ECP} = ECPS \cdot PPS + K \cdot FS \cdot PPS - IC_{ECP} \end{gathered} \tag{4}$$

To make the system sustainable for the ECP, it is mandatory to have a positive benefit:

$$\begin{gathered} benefit_{ECP} > 0 \rightarrow \\ \rightarrow ECPS \cdot PPS + K \cdot FS \cdot PPS - IC_{ECP} > 0 \rightarrow ECPS \cdot PPS + K \cdot FS \cdot PPS > IC_{ECP} \rightarrow \\ \rightarrow PPS\,(ECPS + K \cdot FS) > SC \cdot NS_{ECP} \rightarrow PPS > \frac{SC \cdot NS_{ECP}}{ECPS + K \cdot FS} \rightarrow \\ \rightarrow PPS\,(ECPS + K \cdot FS) > SC \cdot NS_{ECP} \rightarrow PPS > \frac{SC \cdot NS_{ECP}}{ECPS + K \cdot FS} \rightarrow \\ \rightarrow (18.24 + 6.48 \cdot \#\,CPUs) \frac{Duration(days)}{30} > \frac{SC \cdot NS_{ECP}}{ECPS + K \cdot FS} \rightarrow \\ \rightarrow Duration(days) > \frac{SC \cdot NS_{ECP} \cdot 30}{(ECPS + K \cdot FS) \cdot (18.24 + 6.48 \cdot \#\,CPUs)} \end{gathered} \tag{5}$$

Federated Cloud Providers (FCP)

The FCPs will receive slice requests from the ECP in the situations where the ECP has not enough resources for satisfying the request by its own. The FCP will not be able to capture all the payment of the tenant due to the mediation of the ECP, which will retain some amount of money for the fact of mediating in this service requests. Note that in the extreme case, when the ECP has no resources at all, this situation will be equivalent to that of a broker in a federation.

The benefit in the federated cloud providers is:

$$\begin{gathered} benefit_{FCP} = revenues_{FCP} - Costs_{FCP} \rightarrow \\ \rightarrow benefit_{FCP} = (1 - K) \cdot FS \cdot PPS - IC_{FCP} \end{gathered} \tag{6}$$

To make the system sustainable for the FCP, it is mandatory to have a positive benefit:

$$\begin{gathered} benefit_{FCP} > 0 \rightarrow \\ \rightarrow (1 - K) \cdot FS \cdot PPS - IC_{FCP} > 0 \rightarrow (1 - K) \cdot FS \cdot PPS > IC_{FCP} \rightarrow \\ \rightarrow (1 - K) \cdot FS \cdot PPS > SC \cdot NS_{FCP} \rightarrow PPS > \frac{SC \cdot NS_{FCP}}{(1 - K) \cdot FS} \rightarrow \\ \rightarrow (18.24 + 6.48 \cdot \#\,CPUs) \frac{Duration(days)}{30} > \frac{SC \cdot NS_{FCP}}{(1 - K) \cdot FS} \rightarrow \\ \rightarrow Duration(days) > \frac{SC \cdot NS_{FCP} \cdot 30}{(1 - K) \cdot FS \cdot (18.24 + 6.48 \cdot \#\,CPUs)} \end{gathered} \tag{7}$$

Eqs. (6) and (7) are equivalent for all the providers in the federation. Here it is described in general terms as if it was just one FCP in the federation, but the results are equivalent if more than one is also considered (in that case, if all the FCPs share the same values, the result per FCP would be the same divided by the number of FCPs).

NECOS System

Once addressed the benefit of the ECP and the FCPs, it is easy to build the global benefit of the federation system by adding both benefits.

The benefit in the entire NECOS system is:

$$\begin{gathered} benefit_{NS} = benefit_{ECP} + benefit_{FCP} \rightarrow \\ \rightarrow benefit_{NS} = ECPS \cdot PPS + K \cdot FS \cdot PPS - IC_{ECP} + (1 - K) \cdot FS \cdot PPS - IC_{FCP} \rightarrow \\ \rightarrow benefit_{NS} = PPS\ (ECPS + FS) - (IC_{ECP} + IC_{FCP}) \rightarrow benefit_{NS} = PPS \cdot TS - IC_{NS} \end{gathered} \tag{8}$$

To make the system sustainable for the NECOS federation as a whole, it is mandatory to have a positive benefit:

$$\begin{gathered} benefit_{NS} > 0 \rightarrow \\ \rightarrow PPS \cdot TS - IC_{NS} > 0 \rightarrow PPS \cdot TS > IC_{NS} \rightarrow \\ \rightarrow PPS \cdot TS > SC \cdot NS_{NS} \rightarrow PPS > \frac{SC \cdot NS_{NS}}{TS} \rightarrow \\ \rightarrow (18.24 + 6.48 \cdot \#\, CPUs) \frac{Duration(days)}{30} > \frac{SC \cdot NS_{NS}}{TS} \rightarrow \\ \rightarrow Duration(days) > \frac{SC \cdot NS_{NS} \cdot 30}{TS \cdot (18.24 + 6.48 \cdot \#\, CPUs)} \end{gathered} \tag{9}$$

5.4. Experimental study

The experimental analysis has been carried out by using a tool developed in MATLAB. As it was explained before, the total simulation time period is 5 years, which is assumed to be the depreciation period of the servers in all the providers. Also, in this case, simulations have been carried out considering the perspective of different actors like the entry cloud provider and federated cloud providers, as well as the overall view of the NECOS system with a different number of services (i.e., slice requests), in order to analyse if the system behaves in the same way, under different circumstances. All these simulations have been done with *K* = 0,1, which implies that the ECP retains a 10% of the incomes due to the slice request passed to the FCPs.

The slice requests associated to the services in Table 1 are assumed to arrive following a Poisson distribution. Each slice is static in the sense that the resources requested on arrival do not vary along the time. Finally, in case the ECP has not enough resources for fulfilling the entire slice request, such request is passed entirely to another provider in the federation (i.e., the slice request is not divided even in the case that it could be partially satisfied by the ECP).

Again, this study has been done from the same three different points of view.

Entry Cloud Provider (ECP)

Two kind of simulations were carried out from the perspective of the entry cloud provider.

A first block of simulations intended to study how the ECP infrastructure is consumed according to the received demand, for the different kinds of workloads. With that purpose, and in order to understand the behaviour for each of the different service classes, the same number of slice requests is considered for each service. The simulation then varies the total number of servers available at the entry cloud provider, reporting the benefits obtained by the ECP as a function of the number of servers deployed (more servers will made available more CPUs for the slice demands). All the simulations are performed considering that there is no blocking for any demand (i.e., all the demands are served by the ECP or by other provider in the federation in case the ECP has not enough resources).

First, a sensibility analysis with respect the number of demands received by the entry cloud provider was performed, in order to understand what the behaviour of the ECP is and what are the dependencies with respect to scalability issues. The outcomes are depicted hereunder in Figure 6.

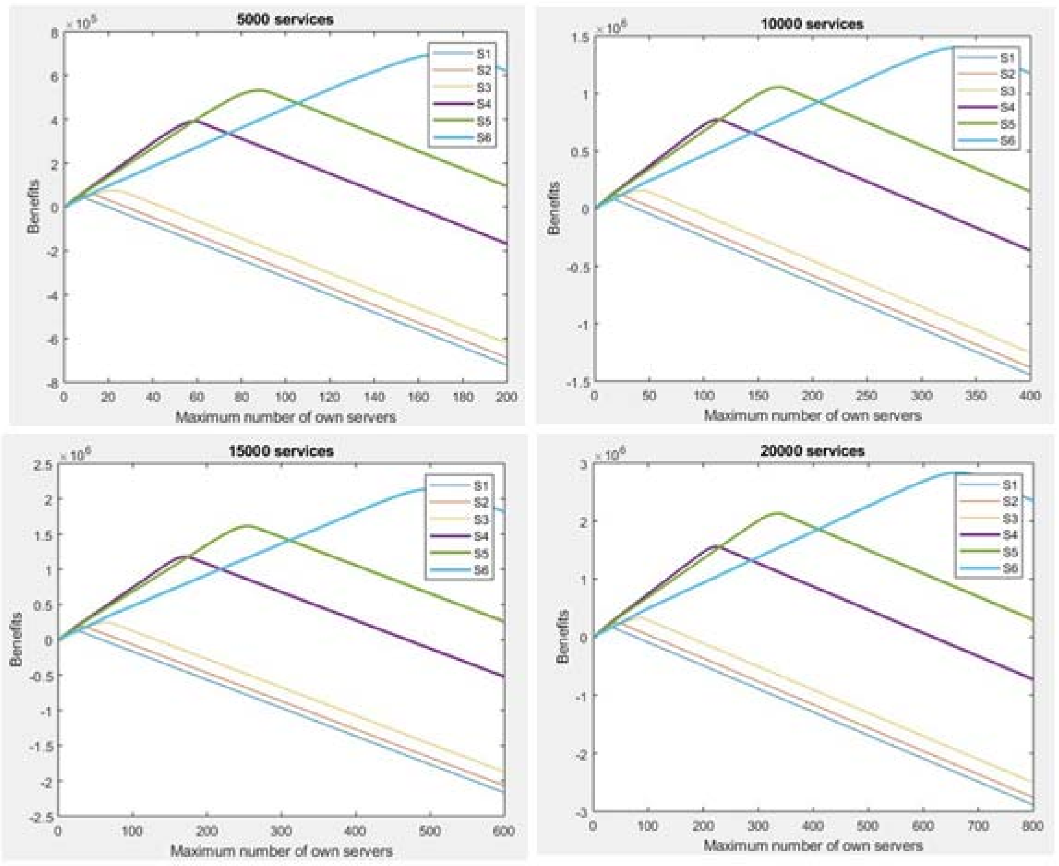


Figure 6. Entry Cloud Provider Benefits according to the number of services

As can be noted, the higher the number of service requests, the higher the number of servers required in the ECP for capturing the maximum benefit. The trend reported is similar independently of the number of service requests. The benefit is increasing up to the point that the costs of the infrastructure erosion the benefits. The number of servers necessary for getting the maximum benefit is slightly less than the number of servers needed for not passing over any demand to another federated provider. At that point the benefits become decreased because the cost of having servers occasionally used. Since the benefits behaviour and the relation between kinds of services are always the same, next simulations consider a total demand of 5000 slices. Note that 5000 slices demanded in a 5 years period mean an average demand of 2,7 slices per day to the federation.

Taking profit of the different characterization of services as indicated in Table 1, the next step is to study by means of these graphs what is the type of service more profitable for the entry cloud provider under the same number of service demands. In other words, to check how the ECP behaves as long as the number of servers increases. At the beginning, with a low number of servers, those will be fully booked for each type of service, and part of the demand will be passed over other providers in the federation. As long as the number of server increases, the number of services transferred to other providers in the federation will decrease. Note that the simulation considers that all the work is kept (that is, service blocking is null). For that, it is assumed that the other providers in the federation can absorb all the demand that cannot be attended by the entry cloud provider. Figure 7 presents how the ECP behaves per type of slice service and minimal number of servers.

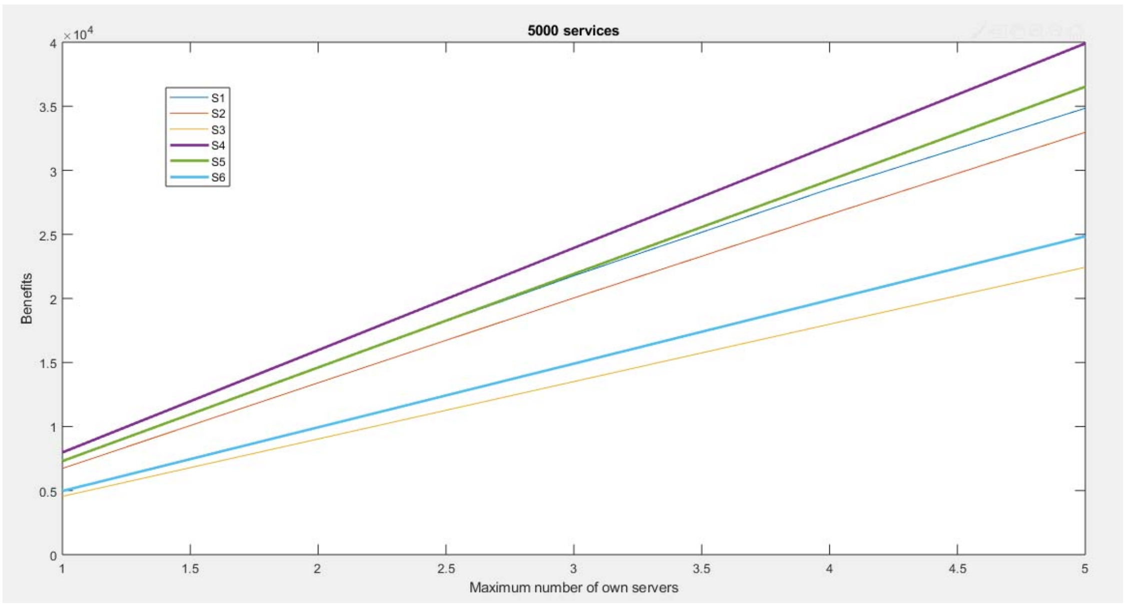


Figure 7. Services profitability comparison

Some insights can be extracted from this graph. Firstly, the most profitable service is type 4. This means that services with less CPU usage and long duration are the most affordable ones. Secondly, the worst services are service 3 and service 6, which are the ones demanding more resources. This insight may seem contradictory, because in Figure 6, service 6 appears as the service type with the highest benefit. This behaviour is due to the fact that, given an equal number of servers, slices requiring more CPUs imply the need of deploying much more servers. The following conclusions can be obtained:

- By fixing the number of slices to be served, the higher benefit is obtained from services of long duration and requiring high number of CPUs (i.e., long and heavy slices), such as type 6 services. This is at the cost of deploying a large infrastructure.
- By fixing the number of servers, the higher benefit is obtained from services of long duration, where the slices requiring less CPU resources perform better (i.e., long and light slices), such as type 4 and 5 services. This is at the cost of passing services over other providers in the federation.

In another rounds of simulation, it has been previously calculated the exact number of servers required for not denying any service for a given demand. Based on this, the percentage of servers that the entry cloud provider has been varied and, as consequence of that, the percentage of servers that the federated cloud provider honours is also varied. Then it is possible to compute the entry cloud provider benefits, and compare them against the benefits of the federation as a whole, the NECOS system (which basically equals to the incomes corresponding to the maximum number of honoured services minus the costs of the minimum servers for attending such demand).

In the following four figures (one for each demand), it can be seen six lines (one each type of service) that are the entry cloud provider normalized benefits as well as line which is the NECOS system normalized benefit.

- 5000 services

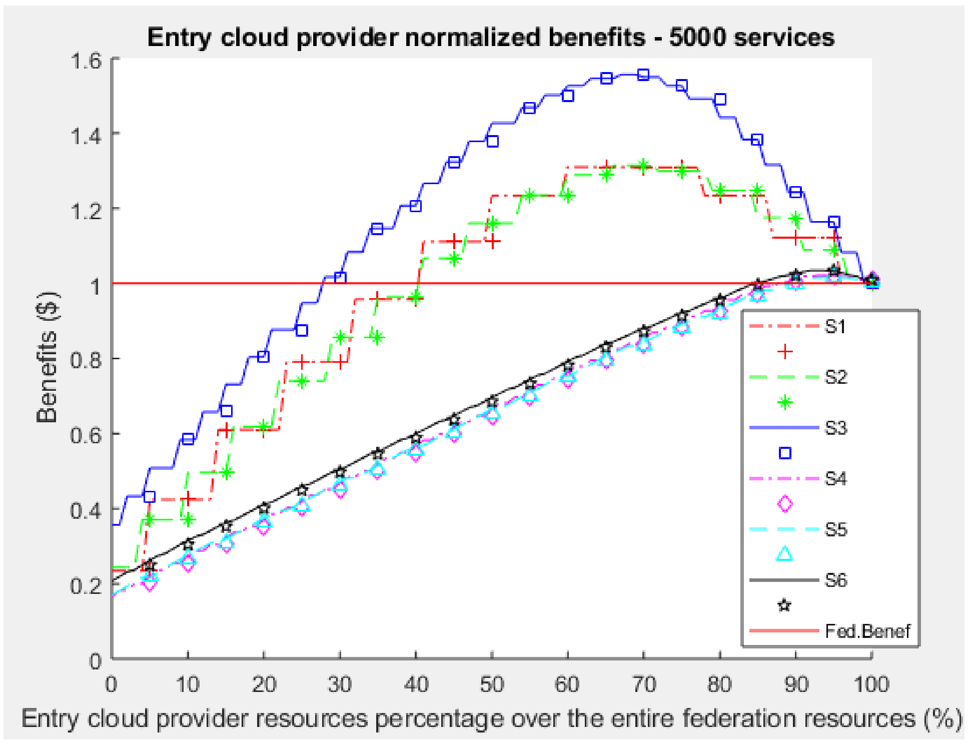


Figure 8. Entry cloud provider benefits – 5000 services

- 10000 services

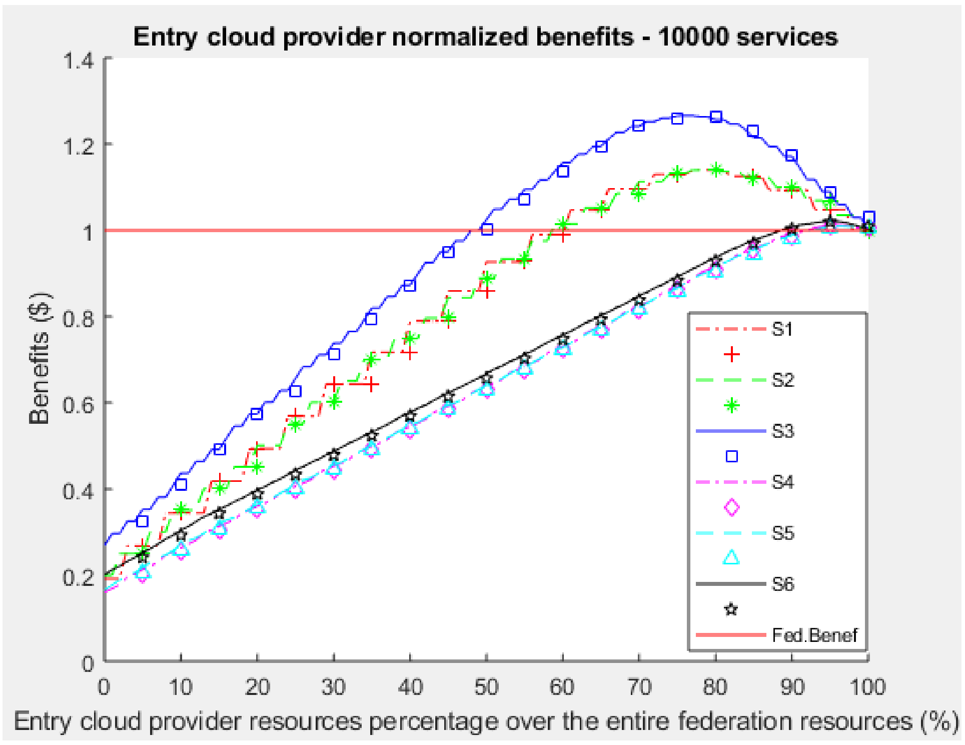


Figure 9. Entry cloud provider benefits – 10000 services

- 15000 services

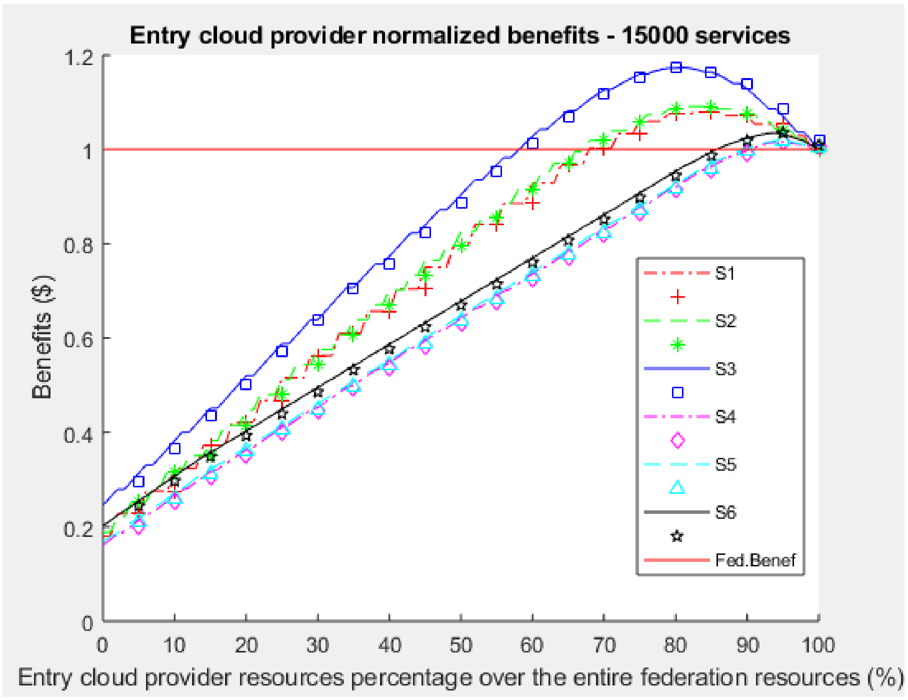


Figure 10. Entry cloud provider benefits – 15000 services

- 20000 services

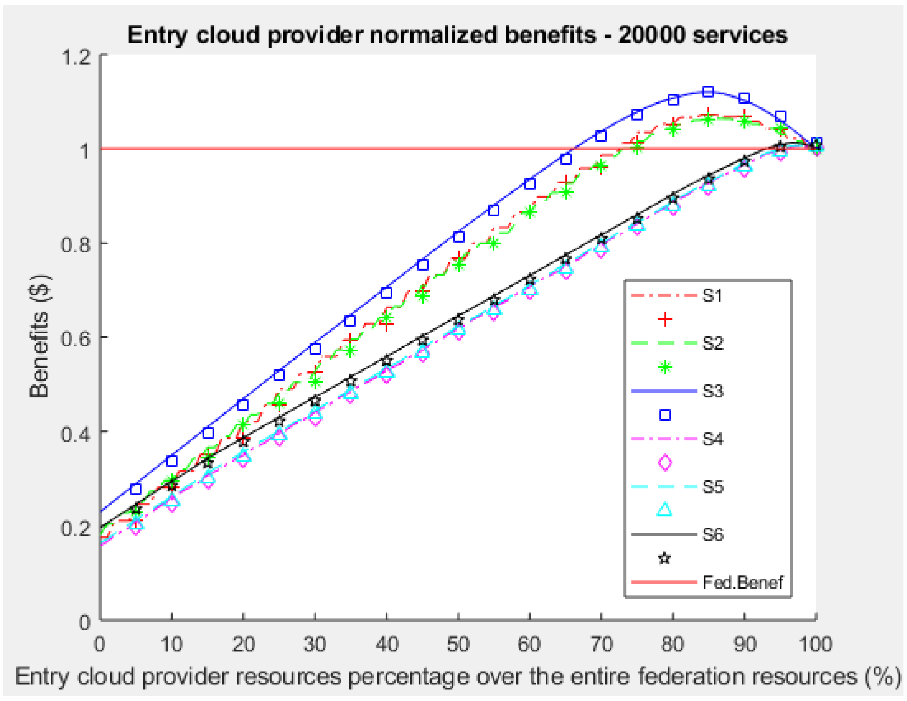


Figure 11. Entry cloud provider benefits – 20000 services

The conclusion here is that in some situations the ECP is able to capture more benefit that the federation as a whole (relative benefit). This is because the fact that when the ECP is able to satisfy the majority of the service requests only a few portions is passed over the federated providers, which have to allocate a dedicated infrastructure (with is corresponding cost) for a small volume of services. Additionally, for those services transferred to the other providers in the federation, the ECP retains a mark-up, which implies that the federated providers' incomes are reduced if compared as if they were being paid by the tenants.

To this respect it seems more beneficial for the ECP to focus on short-lived slices, independently of the number of CPUs required for the slice (relative benefit with respect to the benefit of the federation). Additionally, as the number of service requests increases, the closer can be the maximum benefit of the ECP to the maximum benefit of the federation.

<u>Federated Cloud Providers (FCP)</u>

As it was explained before, the whole NECOS system follows the same equation:

$$benefit_{NS} = benefit_{ECP} + benefit_{FCP} \tag{10}$$

Therefore, if sometimes the entry cloud provider benefit is higher than the NECOS system benefit is because the federated cloud providers' benefit is negative, which means that the federated cloud providers are losing money.

To confirm this assertion, a specific simulation was performed, presenting the benefits of all the parties (for simplicity, a single federated provider is considered). In Figure 12, it can be seen the resulting comparison.

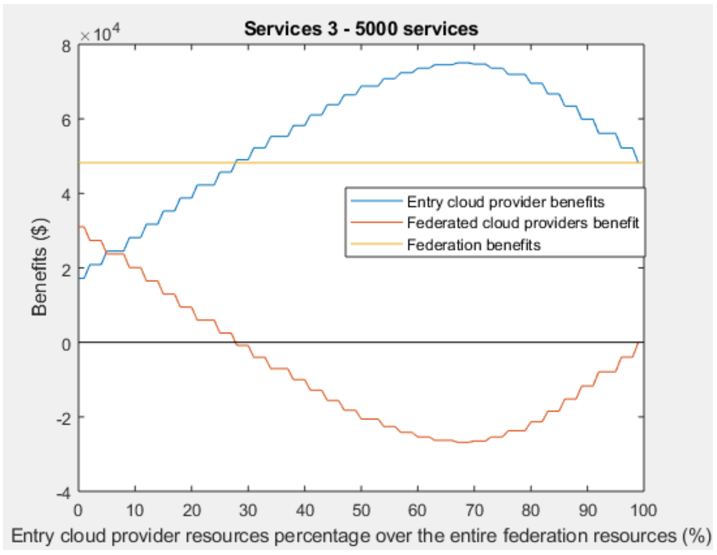


Figure 12. Federated cloud providers benefit

Therefore, it is confirmed that sometimes the federated cloud providers' benefits can be negative. To extract in what scenarios this situation can occur, the following analytical study was done, departing from the definition that the federated cloud providers' benefit is not desired to be negative:

$$\begin{gathered} benefit_{FCP} > 0 \rightarrow \\ \rightarrow (1-K) \cdot FS \cdot PPS - IC_{FCP} > 0 \rightarrow (1-K) \cdot FS \cdot PPS > IC_{FCP} \rightarrow \\ \rightarrow FS > \frac{SC \cdot NS_{FCP}}{(1-K) \cdot PPS} \rightarrow TS \cdot \%_{Federated\ Services} > \frac{SC \cdot NS_{FCP}}{(1-K) \cdot PPS} \rightarrow \\ \rightarrow \%_{Federated\ Services} > \frac{SC \cdot TNS \cdot \%_{Servers\ in\ FCP}}{(1-K) \cdot PPS \cdot TS} \rightarrow \\ \rightarrow \frac{\%_{Federated\ Services}}{\%_{Servers\ in\ FCP}} > \frac{SC \cdot TNS}{(1-K) \cdot PPS \cdot TS} \end{gathered} \quad (11)$$

If this inequality is met, the federated cloud providers' benefit is not negative.

Exemplifying with type 3 services (i.e., short-lived and heavy-weight slices), the values of this index are calculated for these four scenarios.

$$\left.\frac{\%_{Federated\ Services}}{\%_{Servers\ in\ FCP}}\right|_{5000\ services\ 3} = 0.8 \quad (12)$$

$$\left.\frac{\%_{Federated\ Services}}{\%_{Servers\ in\ FCP}}\right|_{10000\ services\ 3} = 0.7 \quad (13)$$

$$\left.\frac{\%_{Federated\ Services}}{\%_{Servers\ in\ FCP}}\right|_{15000\ services\ 3} = 0.66 \quad (14)$$

$$\frac{\%_{Federated\ Services}}{\%_{Servers\ in\ FCP}}{}_{20000\ services\ 3} = 0.63 \tag{15}$$

Observing at the results of the previous figures and of these four operations, it can be concluded that as this index increases, the possibilities of these benefits being negative increase too. This implies that the FCP requires an important percentage of services from the ECP to be profitable, and such percentage decreases the larger is the total service demand.

Looking at the variables in Eq. (11), the only ones that vary are *TNS* and *TS*. In consequence, as the number of services increases, this ratio decreases. However, the relation between *TS* and *TNS* is not linear. This is because *TS* grows faster than *TNS*, that is, the total number of services increases more than the total number of servers needed to serve all of them. For this reason, it will always be more convenient for the Federated Cloud Providers that there is a greater number of total services.

On the other hand, if the ratio in Eq. (11) is 1, it means that there must be at least the same percentage of federated services as of servers in the Federated Cloud Providers over the total number of servers in NECOS system. To study this specific scenario, a concrete simulation is carried out. For this, it is considered the simulation of 5000 type 3 services, varying the variable *K*.

Furthermore, the relation in Eq. (11) depends on *K*, the larger the mark-up retained by the ECP, the higher the percentage of services needed to reach profitability. Interestingly, from Eq. (11) it can also be determined the maximum mark-up to apply in the federation. When the total cost of the servers equals the incomes due to the transfer of services from ECP to FCP, the benefit of the FCP is 0. Less incomes (because of higher mark-up) impacts on the FCP sustainability. Then,

$$\frac{SC \cdot TNS}{(1 - K) \cdot PPS \cdot TS} = 1 \rightarrow K = 1 - \frac{SC \cdot TNS}{PPS \cdot TS} \tag{16}$$

Solving this equation, with *K* = 0.28. In consequence, if the ECP retains more than 28% of the price paid by the tenant in the services passed to the FCP, the FCP will have a negative financial balance. Attending to Eq. (16) the higher the price of the slice as paid by tenant, or the higher the number of service requests, the higher can be the mark-up to be applied by the ECP.

The following considerations can be obtained:

- The larger the number of total services requested, the lower the number of services required by the FCP for being profitable.
- In any case, the FCP requires a large number of services diverted from the ECP for being profitable, if it only counts on federated services for its business model. Then it is required to complement its incomes with own services (slices) directly traded with tenants, that is, it has to become an ECP by itself for guaranteeing sustainability.

<u>Overall NECOS system</u>

In the previous simulations, the perspectives of both the ECP and the FCP have been analyzed. For doing that, it has been taken into account the exact number of servers required for avoid denying any service request (that is, no service blocking).

Now the analysis will consider the percentage of servers that the overall NECOS system (i.e., ECP plus FCPs) has when compared to the number of servers required for not denying any service request, which implies that in these simulations the system denies services (except when the system has 100% of resources). In this respect, there are computed the overall NECOS system benefits, revenues, costs and the percentage of services not denied.

In the following eight figures (two for each number of service requests), it can be seen six different lines (one per each type of service) that are the NECOS system benefits (normalized and not normalized, respectively).

- 5000 services

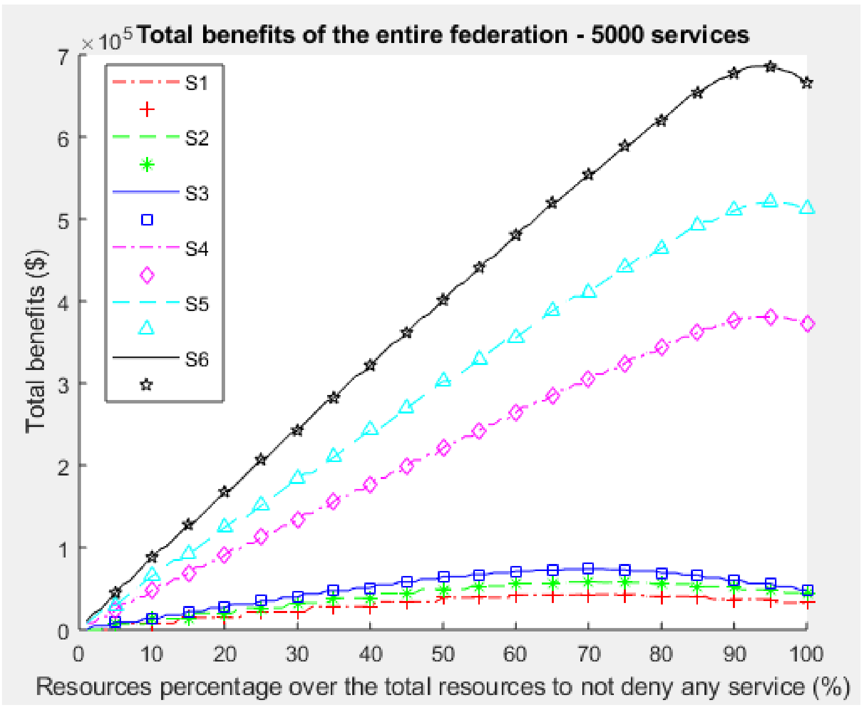


Figure 13. Total benefits of the NECOS system – 5000 services

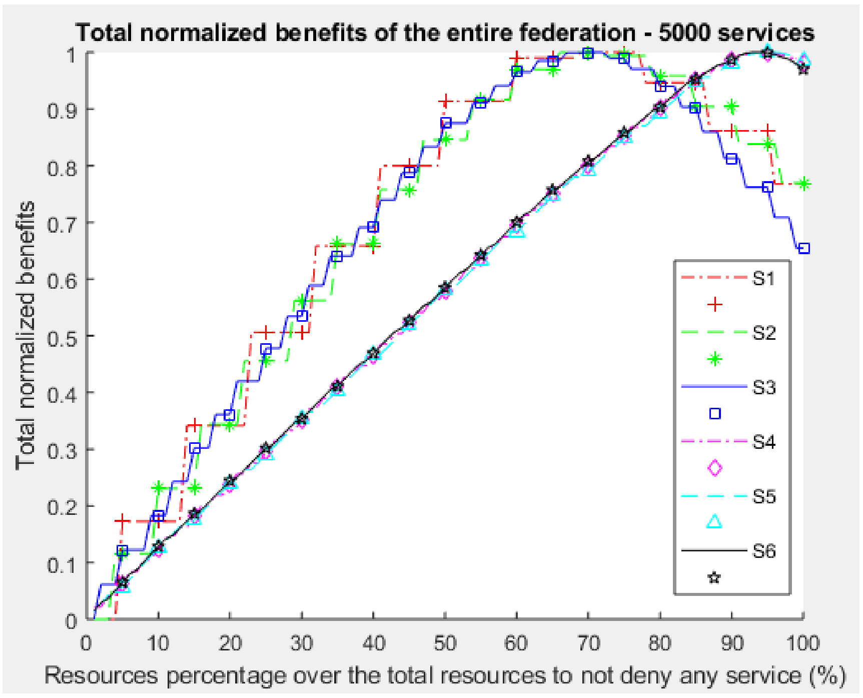


Figure 14. Total normalized benefits of the NECOS system - 5000 services

EUB-01-2017

- 10000 services

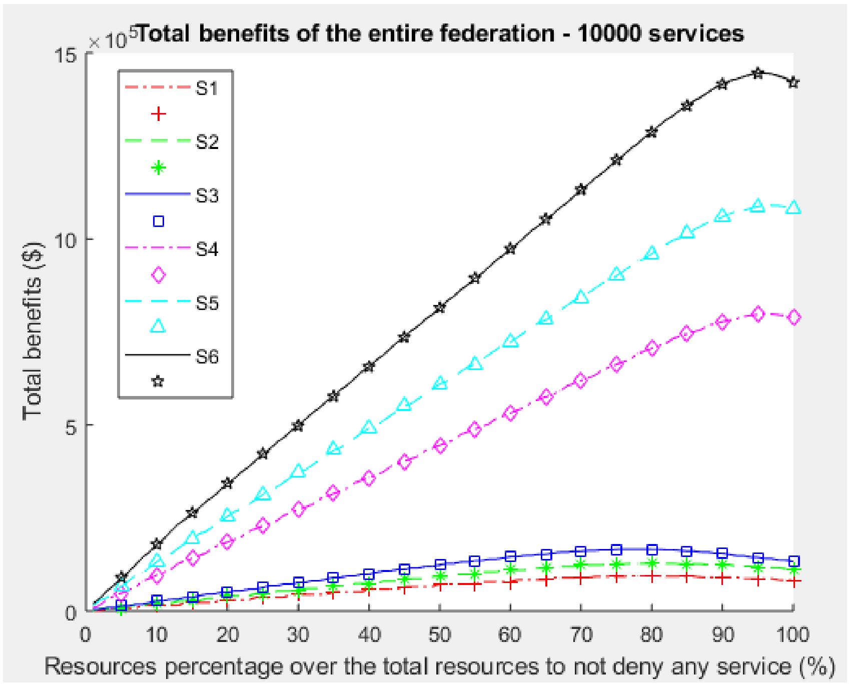


Figure 15. Total benefits of the NECOS system – 10000 services

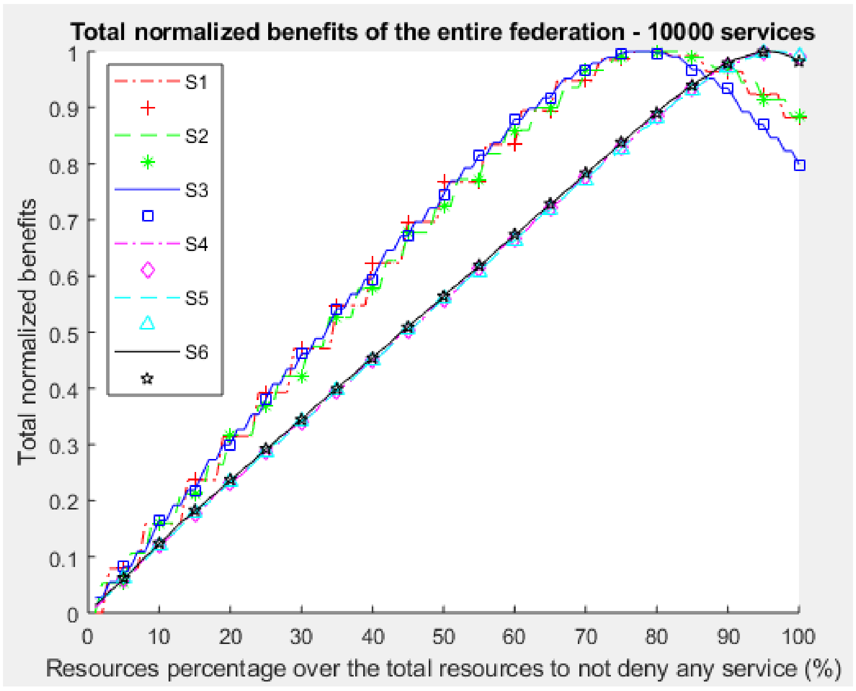


Figure 16. Total normalized benefits of the NECOS system – 10000 services

- 15000 services

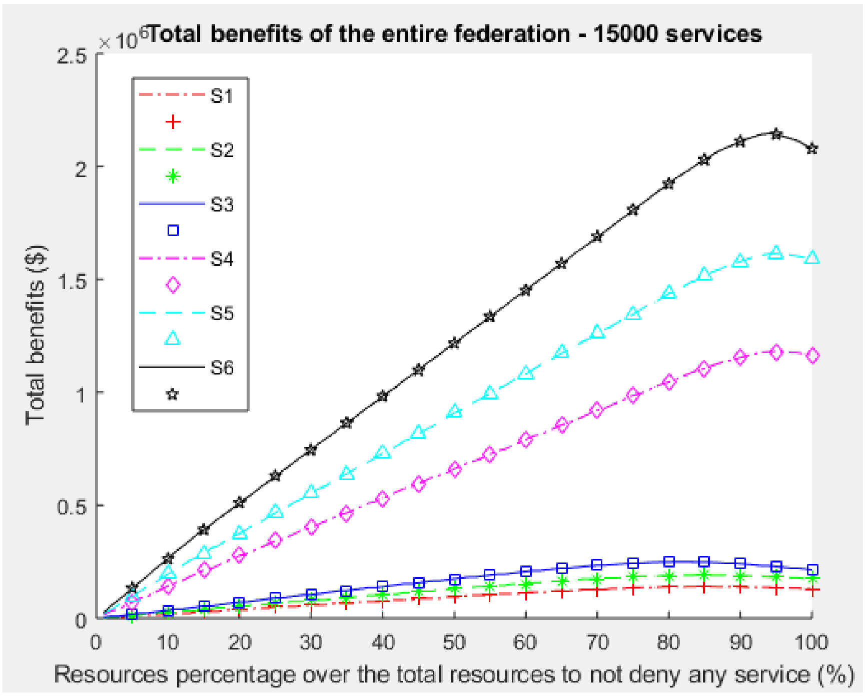


Figure 17. Total benefits of the NECOS system – 15000 services

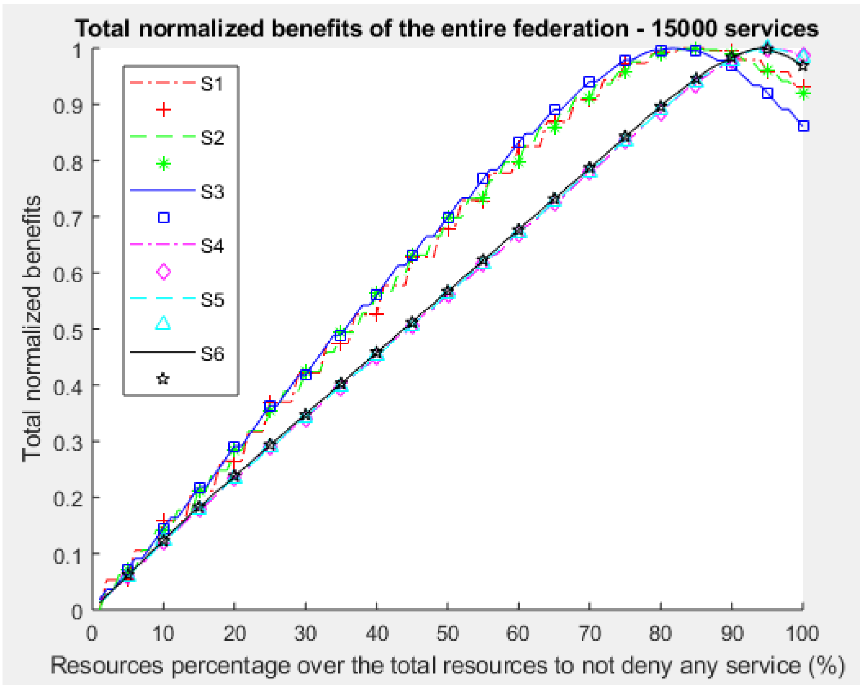


Figure 18. Total normalized benefits of the NECOS system – 15000 services

- 20000 services

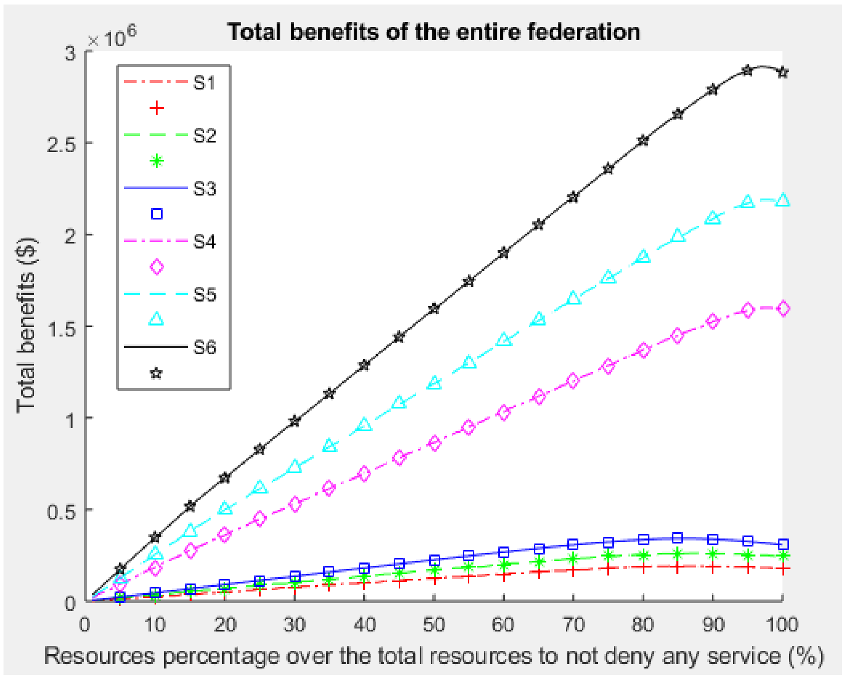


Figure 19. Total benefits of the NECOS system – 20000 services

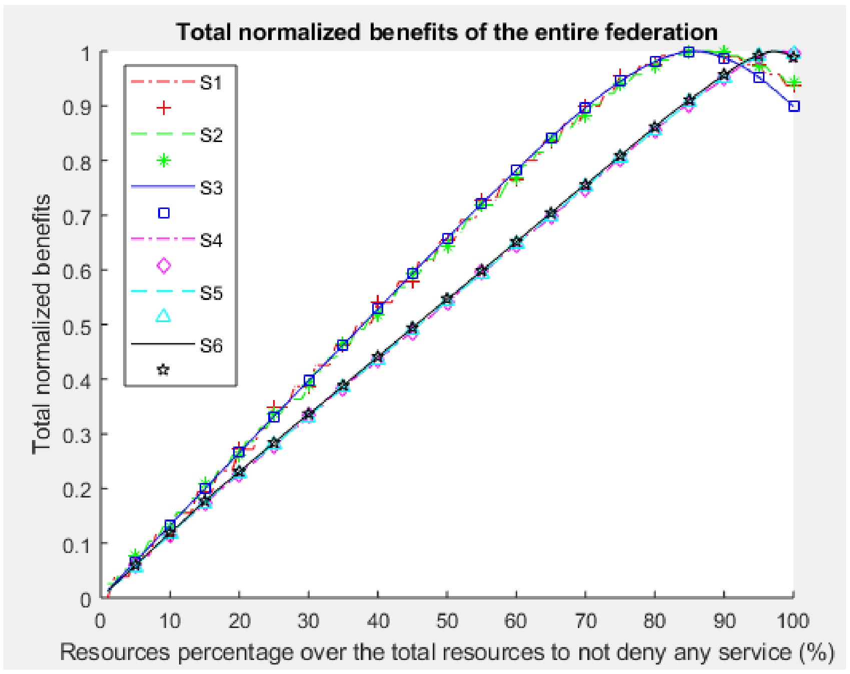


Figure 20. Total normalized benefits of the NECOS system – 20000 services

As a first outcome, it can be inferred from the not normalized graphs that the services generating larger benefits are the long-lived ones (for the same number of services). In the same way, another conclusion from the normalized graphs is that the NECOS federation can achieved higher benefits by denying some services and having less infrastructure than the situation in which all the slices are served (e.g., with 5000 services, short-lived services get the higher benefits with just a 70% of the resources that would be required for not blocking any slice demand). However, this behaviour is minimized as increasing the number of services.

The following results can be highlighted:

- Long-lived services generate more benefits to the federation.
- For services of equivalent duration, heavy-weight services generate more benefits than light-weight ones.
- Light-weight services reach maximum benefits with lower percentage of resources respect to the total ones needed for not denying any demand, than heavy-weight services.
- The percentage of resources needed for reaching the maximum benefit with respect to the total ones needed for not denying any demand increases with the growth of service requests.

5.5. Summary of the business analysis

NECOS federation has been characterized through a number of analytical and experimental analysis with the objective of understanding its behaviour when a number of slices of different kind are requested. The main findings of this section are:

- By fixing the number of slices to be served, the higher benefit is obtained from services of long duration and requiring high number of CPUs (i.e., long and heavy slices), such as type 6 services. This is at the cost of deploying a large infrastructure.
- By fixing the number of servers, the higher benefit is obtained from services of long duration, where the slices requiring less CPU resources perform better (i.e., long and light slices), such as type 4 and 5 services. This is at the cost of passing services over other providers in the federation.
- The larger the number of total services requested, the lower the number of services required by the FCP for being profitable.
- In any case, the FCP requires a large number of services diverted from the ECP for being profitable, if it only counts on federated services for its business model. Then it is required to complement its incomes with own services (slices) directly traded with tenants, that is, it has to become an ECP by itself for guaranteeing sustainability.
- Long-lived services generate more absolute benefits to the federation.
- For services of equivalent duration, heavy-weight services generate more benefits than light-weight ones.
- Light-weight services reach maximum benefits with lower percentage of resources respect to the total ones needed for not denying any demand, than heavy-weight services.
- The percentage of resources needed for reaching the maximum benefit with respect to the total ones needed for not denying any demand increases with the growth of service requests.

With all of this in mind, the following business approaches can be considered:

- FCPs need to complement their business with slice offerings direct to tenants, in order to ensure sustainability. That is, it is recommended to become ECP by themselves, making available to the federation resources that could be vacant during some periods of time.
- For ECPs it seems more convenient to focus on long-lived slices as primary business focus.
- Two different kinds of ECP could emerge, ones focused on light-weight slices and another one in heavy-weight ones.

Taking these conclusions to NECOS, it seems that a federation with execution environments of different capabilities, like Telco Cloud (for heavy-weight services) and MEC (for light-weight ones) seems sustainable, with primary focus on long-lived slices to guarantee a higher level of benefits, and behaving as ECPs where some of the service requests are passed over to the federation with the rationale of minimising the infrastructure necessary for achieving the maximum benefit (for those providers able to optimize their infrastructure) and using some vacant capacity in the federation (for those providers not able of getting such optimization, but using this as complement to their business model as ECPs).

6. Conclusions and outlook

This deliverable collects the final outcomes of WP2 for feeding the rest of the work packages in the project.

First, a refinement on use cases and scenarios of NECOS is provided, reflecting the relevance of the use cases selected (MEC and Telco Cloud), which reinforces the approach of the project. Regarding the scenarios of applicability identified in the consortium, minor refinements are provided, basically presenting the structure of main scenarios and sub-scenarios of interest. The approach towards these scenarios has been to consider service situations of interest from the partners in the consortium, in such a way that realistic solutions could be defined for them. All of these helps later on to identify the requirements to be supported by NECOS.

Starting from the initial scenarios' requirements already identified in D2.1, WP2 exercised an analysis of requirements oriented towards the characterization of the NECOS platform itself. In order to do that the BPMN was followed, analyzing different workflows of essential operations in NECOS. These requirements, both functional and non-functional, have then fed the other technical work packages focused on the architecture and development of the NECOS system.

The stakeholders and actors of the NECOS system have been also updated in line with the work produced by WP3 on architecture. The refined view simplifies the preliminary exercise and provides a more straightforward rationale for the interactions among actors in NECOS.

Finally, this deliverable provides a techno-economic analysis of the NECOS ecosystem from three different perspectives: the one of the cloud provider receiving the slice demands from the tenant; the one from the federated cloud providers; and the one from the overall federation. Simulations and analytical studies have been performed, deriving from them business guidelines related to NECOS-based solutions.

All these inputs will be used by the rest of work packages in the remaining time of the project. Use cases, scenarios, requirements and stakeholders can be considered as stable at this point on time. As consequence of future developments in the other work packages of the project new techno-economic aspects of interest could be explored. If that is the case, such analysis will be documented in the future deliverables of the technical work packages.

References

[1] ITU-T IMT-2020, Focus Group on Network Aspects of IMT-2020 - Deliverables, 2017. available at: https://www.itu.int/en/publications/Documents/tsb/2017-IMT2020-deliverables/mobile/index.html#p=1

[2] ITU - RM.2083-0, IMT Vision - Framework and Overall Objectives of the Future Development of IMT for 2020 and beyond, Sep. 2015.

[3] 5G Vision: The 5G Infrastructure Public Private Partnership: The Next Generation of Communication Networks and Services, Feb. 2015.

[4] A. Osseiran, et al., "Scenarios for 5G mobile and wireless communications: the vision of METIS project", IEEE Comunications Magazine, Vol.52, No.5, pp. 26-35, May 2014.

[5] 5GPPP, 5G Empowering Vertical Industries, 2016.

[6] 3GPP TR 23.799. "Study on Architecture for Next Generation System", Rel. 14, Dec. 2016.

[7] ITU-T Y3011, Framework of Network Virtualization for Future Networks, Next Generation Network - Future Networks, Jan. 2012.

[8] S. Shenker, "Fundamental Design Issues for the Future Internet". IEEE Journal on Selected Areas in Communications, Vol. 13, No. 7, pp. 1176-1188, Sep. 1995.

[9] NGMN Alliance, Description of Network Slicing Concept, NGMN 5G P1 Requirements & Architectural, Work Stream End-to-End Architecture, version 1.0, Jan. 2016.

[10] I. Farris, T. Taleb, H. Flinck, and A, Iera, "Providing Ultra-Short Latency to User-Centric 5G Applications at the Mobile Network Edge", in Trans. on Emerging Telecomm. Technologies (ETT), Mar. 2017.

[11] Lookup Safety Recalls & Service Campaigns by VIN, Toyota.com, 2017.

[12] T. Taleb, A. Ksentini, and A. Kobbane, "Lightweight Mobile Core Network for Machine Type Communications", in IEEE Access Magazine, Vol. 2, pp. 1128-1137, Oct. 2014.

[13] NGMN, "5G white paper", February 2015, available at: https://www.ngmn.org/uploads/media/NGMN_5G_White_Paper_V1_0.pdf

[14] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini and H. Flinck, "Network Slicing and Softwarization: A Survey on Principles, Enabling Technologies, and Solutions," in *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2429-2453, third quarter 2018.

[15] CenturyLink – Online Price Estimator, 2018. Available at URL: https://uk.ctl.io/estimator/

[16] B. Suleiman, S. Sakr, R. Jeffrey, A. Liu, "On understanding the economics and elasticity challenges of deploying buisness applications on public cloud infrastructure", J. Internet Serv. Appl., No. 3, pp. 173-193, 2012.

[17] N.C. Luong, P. Wang, D. Niyato, W. Yonggang, Z. Han, "Resource management in cloud networking using economic analysis and pricing models: a survey", IEEE Communications Surveys and Tutorials, Vol. 19, Issue 2, pp. 954-1001, 2017.

[18] J.B. Abdo, J. Demerjian, H. Chaouchi, K. Barbar, G. Pujolle, "Cloud federation means cash", in Proc. of the 3rd International Conference on e-Technologies and Networks for Development (ICeND), Beirut (Lebanon), May 2014.

[19] N. Ghrada, M.F. Zhani, Y. Elkhatib, "Price and Performance of Cloud-hosted Virtual Network Functions: Analysis and Future Challenges", in Proc. of the 4th IEEE Conference on Network Softwarization and Workshops (NetSoft), Montreal, Canada, June 2018.

[20] https://www.bechtle.com/es/shop/hardware/infraestructura-de-ti/servidores--10006009--c

[21] Calculating Depreciation of Server Hardware Cost, https://docs.vmware.com/en/vRealize-Business/7.0.1/com.vmware.vRBforCloud.install.doc/GUID-26B04DFB-1A47-4424-B6FA-18A51FCAC7E0.html

[22] A.K. Mishra, J.L. Hellerstein, W. Cirne, C.R. Das, "Towards characterizing cloud backend workloads: insights from Google compute clusters", ACM SIGMETRICS Performance Evaluation Review, Volume 37 Issue 4, pp. 34-41, March 2010.

Version History

Version	Date	Author	Change record
0.1	09.11.2018	S. Vivas, L.M. Contreras	Document creation
0.6	23.11.2018	All	Final contributions
2.0	30.11.2018	L.M. Contreras, S. Vivas	Final edition

APPENDIX A

This appendix is the consolidation of the refactoring of the previous four initial scenarios from D2.1 in 7 scenarios presents here.

1. virtual RAN

The virtual RAN (vRAN) scenario takes advantage of the virtualization trends, whose consideration for mobile networks is gradually increasing, specifically in the radio access components.

1.1 Description

Mobile Network Operators (MNOs), including the ones leveraging on other operator's infrastructure, known as Mobile Virtual Network Operators (MVNOs), are adopting virtualization as technological paradigm for the deployment of network and services in a general way. While the virtualized approach is already quite evolved in the core network, this trend is also being extended to the Radio Access Network (RAN) for a number of reasons:

- To open the industrial ecosystem by decoupling H/W and S/W for RAN nodes.
- To reduce costs, by means of sharing infrastructure resources, more interestingly at the remote locations where the scarcity of resources can be larger.
- To improve network performance in general terms.
- To provide flexibility for adapting to standard evolutions and traffic demands.

The deployed infrastructure for providing network functions associated to the RAN can also be extended to support different services providers. In this context, vRAN presents potential benefits for multiple stakeholders. For example, Network Connectivity Providers can make available near to the edge resources to attend applications that demand for ultra-Reliable and Low Latency Communications (uRLLC); and tenants have an infrastructure to provide localized or contextualized services. However, this scenario involves sharing of a common infrastructure among tenants with different needs. Additionally, the owner of the infrastructure is interested on optimizing the usage of the resources, what must happen transparently to the tenants.

In Figure A1, we present an example to illustrate the main problem faced in this scenario. The figure shows where two tenants have users and so where they demand for coverage. Tenant 1 offers an enhanced Mobile Broadband (eMBB) service for its users, while Tenant 2 users rely on an uRLLC application. Additionally, the MNO also has its own traditional mobile users. To cover the area, the MNO employs a combination of vRAN and MEC infrastructure with two Cloud/Central Unit (CU), i.e., CU1 and CU2. The MNO needs to assure the performance required by each tenant but also wants to optimize the resource usage and attend its own users.

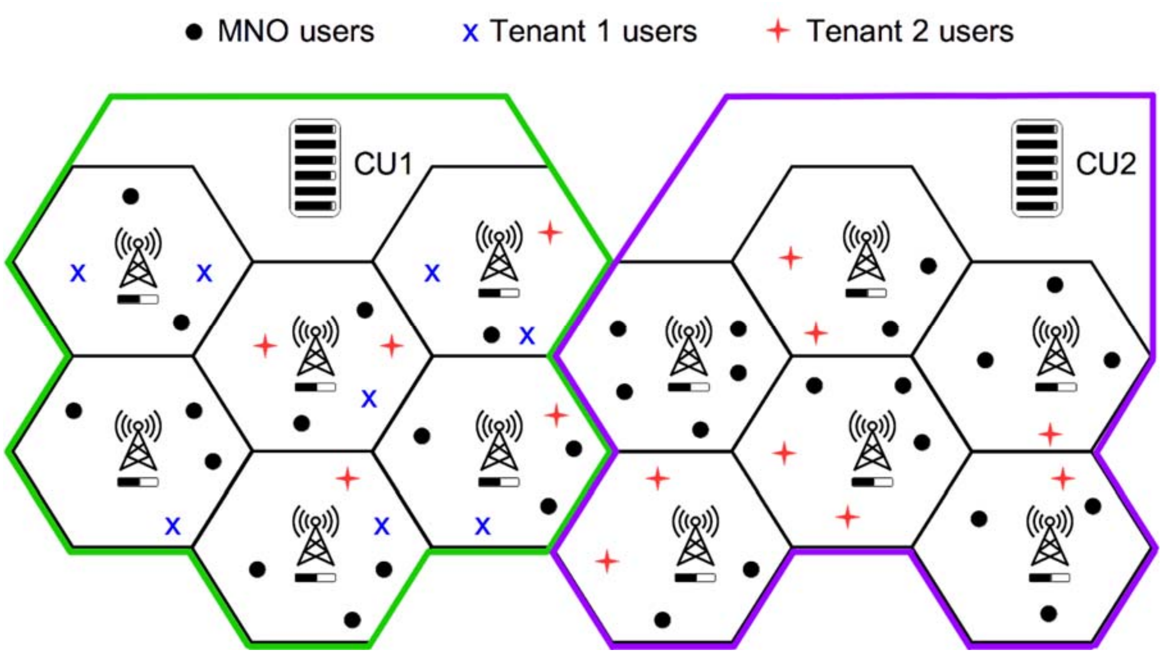


Figure A 1. vRAN scenario with two distinct tenants

1.2 Technical enablers

In order to attend this vRAN scenario, NECOS introduces cloud slicing which encompass several technical enablers. Figure A2 illustrates the NECOS approach in which three slices are created to address the issues raised previously. There is one slice for each tenant and an additional slice for the MNO.

The slicing system must be able to create an isolated set of resources (i.e., a slice) for each tenant. These resources include network, computing and storage. All resources in the same slice must be connected and must be identified by the tenant as a unique infrastructure. A tenant must be able to monitor the performance metrics concerted in its SLA. A tenant must be allowed to request management and orchestration capabilities for its slice. Naturally, due to the isolation, the operation performed by a tenant must not affect any other tenant. The slicing system must also facilitate the service deployment despite the software technology employed by a tenant.

The MNO must be able to monitor, manage, orchestrate and optimize the whole infrastructure, but these actions must not disturb the SLA negotiated with any tenant. As illustrated by this scenario, a MNO may need to create a slice in order to support its own users. However, an MNO may need a slice due to other reasons, e.g., to offer communication services for tenants in a transparent manner.

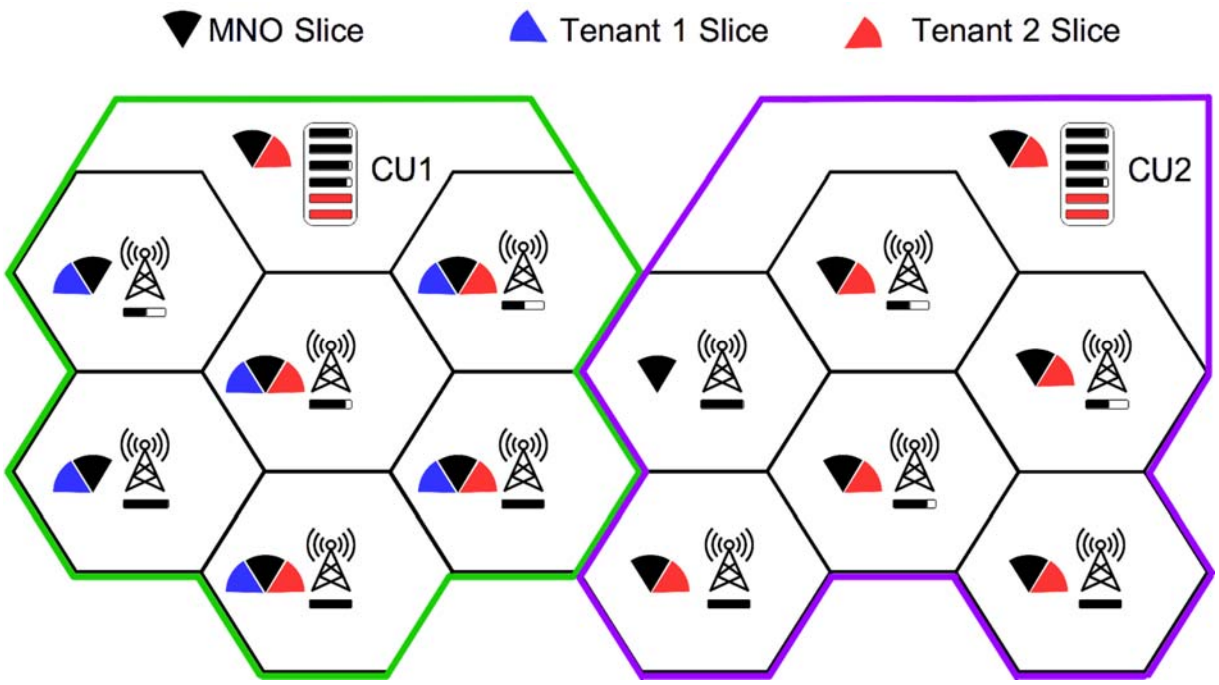


Figure A 2. Slices to support the vRAN scenario

1.3 *Critical success factors and KPIs*

The following factors are identified as critical for success of this scenario:

- F1 - High degree of scalability (variable and dynamic number of users), with a short-time response for the deployment on the field;
- F2 - Fast service deployment;
- F3 - Real-time monitoring;
- F4 - Support to ultra-reliable and low latency communications;
- F5 - Support to enhanced mobile broadband.

These critical success factors are mapped to the following KPIs:

- K1 - Accomplishment of operations such as creation, enlargement, shrink of slice. Metrics - Provisioning time, and decommission time;
- K2 - Provision of the slice to the tenants considering a service that defines the time for this provisioning. Metrics - Service provisioning time;
- K3 - Provision of monitored data to the operator and the tenant. Metrics - Monitoring-data availability;
- K4 - Accomplishment of operations such as creating, enlarging, reducing and deactivating a slice within acceptable time periods. Metrics - end-to-end delay;
- K5 - Provision of data transfer rate in order to meet the tenants. Metrics - throughput.

1.4 *Mapping to NECOS key characteristics*

The following NECOS characteristics are identified as key enablers of this scenario:

- C1: Slice as a Service model dynamically allocates, modify, or deallocates slices on-demand;
- C2: Adaptations and reconfigurations are done at a per slice level, keeping the proper isolation;
- C3: Each aspect that comprises the cloud environment - from the networking between virtual machines, to the SLAs of the hosted applications - is managed via software;
- C4: Lightweight management and virtualization systems deployable on large number of small servers at the network edge.

1.5 Requirements

The following functional requirements were identified:

- RF.vRAN.1: *Service Level Agreement*. The system must assure all the performance metrics (bandwidth, latency, CPU) negotiated for each slice.
- RF.vRAN.2: *Accountability*. The system must offer monitoring and accounting in a per slice basis.
- RF.vRAN.3: *On-demand slice provisioning*. The system must allow the operator to create or adapt the slices on demand.

The following non-functional requirements were identified:

- RN.vRAN.1: *Isolation of slice resources*. Tenants must be protected from each other, i.e., it must be avoided any information leakage (e.g., monitoring measurements) among the slices.
- RN.vRAN.2: *Fairness*. Operator must be able to optimize the resource usage without negative impact on any tenant.
- RN.vRAN.3: *Fault detection*. Any service should operate continuously for a long time, so failures in the slice level should be automatically handled without impacting the service.

2. 5G services

Network operators are facing now the need of adapting their existing networks in order to be able of providing forthcoming 5G services. Three main types of services have been identified so far: enhanced Mobile Broadband (eMBB), massive Machine-Type Communications (mMTC) and ultra-Reliable and Low Latency Communications (uRLLC). The eMBB service type encompasses the challenge of providing an unprecedented volume of data delivery, associated with e.g., high-definition video sharing. The mMTC focuses on applications where a large number of IoT devices, such as sensors, collectively creates a significant data volume passing through the network. Moreover, these data are highly localized and are often associated with requirements like privacy, data ownership, etc. Finally, the uRLLC type refers to services in the need for extremely low end-to-end latency, like Tactile Internet, Interactive Gaming, Virtual Reality, Automotive, Industry and Automation.

2.1 Description

Figure A3 presents the scenario considered here. The purpose is to segregate the eMBB, uRLLC and mMTC services in different slices on top of the same infrastructure, then creating network partitions tailored and adapted to them.

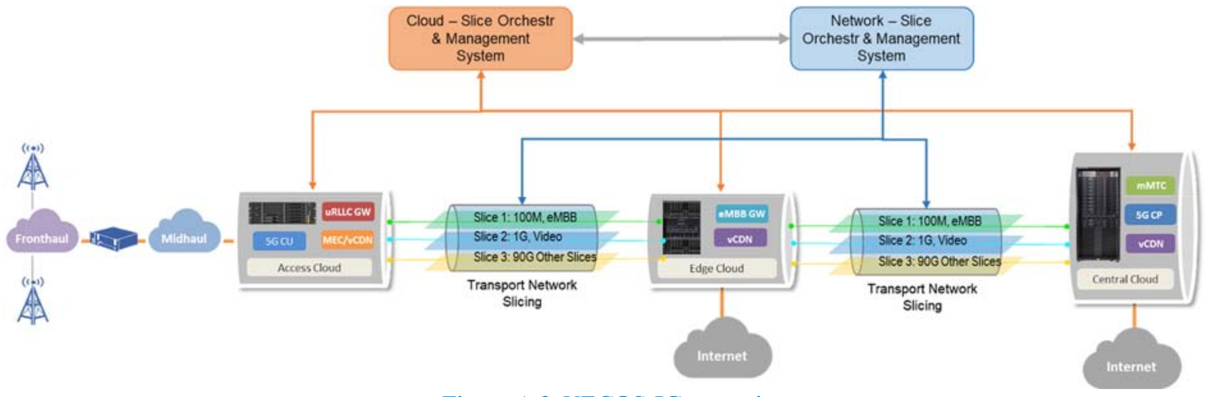

Cloud – Slice Orchestr & Management System
Network – Slice Orchestr & Management System
Fronthaul
Midhaul
uRLLC GW
5G CU
MEC/vCDN
Access Cloud
Slice 1: 100M, eMBB
Slice 2: 1G, Video
Slice 3: 90G Other Slices
Transport Network Slicing
eMBB GW
vCDN
Edge Cloud
Internet
Slice 1: 100M, eMBB
Slice 2: 1G, Video
Slice 3: 90G Other Slices
Transport Network Slicing
mMTC
5G CP
vCDN
Central Cloud
Internet

Figure A 3. NECOS 5G scenario

It is worthy to note that in 5G networks the slicing capabilities should extend also to the transport network. Then some mechanisms should be in place to jointly provide the aforementioned network partitions, including both IT and network resources. Consequently, there is a need for interactions with other systems, in this case transport-related orchestrator and management systems.

Finally, sufficient mechanisms for control and management have to be facilitated for external tenants to manage their own services running on the slice, including the allocated resources.

2.2 Description

When applies to 5G, the slicing system must be able to create slices tailored to the specific requirements dictated by the kind of scenario to be supported. Parameters like latency, bandwidth or number of sessions should be interpreted for allocating necessary resources and for implementing smart decisions with respect to the kind of resources and the location of them in the network.

Since the usage of distributed resources can be the norm, the slicing system should interact with control and management systems devoted to configure the transport network in such a way that the service continuity is ensured. The possibility of enforcing the chaining of service functions, soliciting paths with specific characteristics, etc., should be considered in the design.

Finally, since the tenants in this scenario can be assumed to be vertical industries, compliance to SLAs is a mandatory task. This implies the need of reporting sufficient performance indicators in such a way that SLAs can be checked periodically, reacting in case of any violation of the agreed terms.

2.3 Critical success factors and Key Performance Indicators

The following factors are identified as critical to success of this scenario:

- F1 - High degree of scalability (variable and dynamic number of users), with a short-time response for the deployment on the field;
- F2 - Fast service deployment;
- F3 - Real-time monitoring;
- F4 - Support to ultra-reliable and low latency communications;
- F5 - Support to enhanced mobile broadband;
- F6 - Support of massive communications (high density of sessions);
- F7 - Flexible mechanisms for supporting service requests from different vertical industries;
- F8 - Means of verifying compliance of agreed SLAs.

These critical success factors are mapped to the following KPIs:

- K1 - Accomplishment of operations such as creation, enlargement, shrink of slice. Metrics - Provisioning time, and decommission time;
- K2 - Provision of the slice to the tenants considering a service that defines the time for this provisioning. Metrics - Service provisioning time;
- K3 - Provision of monitored data to the operator and the tenant. Metrics - Monitoring-data availability;
- K4 - Accomplishment of operations such as creating, enlarging, reducing and deactivating a slice within acceptable time periods. Metrics - end-to-end delay;
- K5 - Provision of data transfer rate in order to meet the tenants’ needs in case of eMBB kind of services. Metrics – throughput;
- K6 – Provisioning of delay measurements to meet the tenants’ needs in case of uRLLC kind of services. Metrics – latency;
- K7 – Provisioning of computing power to meet the tenants’ needs in case of mMTC kind of services. Metrics – CPUs.

2.4 Mapping to NECOS key characteristics

This scenario considers the applicability of NECOS as solution for creating the slices necessary to support differentiated services in 5G. The following NECOS characteristics are identified as key enablers of this scenario:

- C1: Slice as a Service model, considering very distinct kind of slices to fit the different nature of the 5G services.
- C2: Smart allocation and placement of resources for ensuring SLAs as requested by the vertical customers (the tenants).
- C3: Lightweight management, especially towards the access, where the cloud resources will be limited.

2.5 Requirements

The following functional requirements were identified:

- RF.5G.1: Service Level Agreement. The system must assure all the performance metrics (bandwidth, latency, CPU) negotiated for each slice.
- RF.5G.2: Accountability. The system must offer monitoring and accounting in a per slice basis.
- RF.5G.3: On-demand slice provisioning. The system must allow the operator to create or adapt the slices on demand.
- RF.5G.4: External control and management of the offered slices. The system must allow the control and management of the resources allocated to the tenant if it is demanded by the tenant in the service request.

The following non-functional requirements were identified:

- RN.5G.1: Isolation of slice resources. Tenants must be protected from each other, i.e., it must be avoided any information leakage (e.g., monitoring measurements) among the slices.
- RN.5G.2: Fairness. Operator must be able to optimize the resource usage without negative impact on any tenant.
- RN.5G.3: Fault detection. Any service should operate continuously for a long time, so failures in the slice level should be automatically handled without impacting the service.

3. vCPE

Fixed telco environments are also target for the usage of cloud capabilities where locating total or partial communications functions, such as a virtual Customer Premises Equipment (vCPE).

3.1 Description

Traditional CPE deployments have network functions installed on customer site to provide local NAT, local DHCP, IGMP proxy-routing, PPP sessions, routing and etc., along with remote site connectivity. This model has the following characteristics:

- Heterogeneous installed park;
- Many legacy devices;
- Unequal services portfolio;
- Expensive operations and support.

Virtual CPE claims to address many issues related to the current model by turning the CPE into a very simple standard device while moving the network functions to a cloud infrastructure, where they run over virtualization technologies. The goal is to simplify the deployment, support and maintenance procedures of CPEs for network service providers, allowing them to deliver dynamic services to their subscribers, as depicted in the Figure A4.

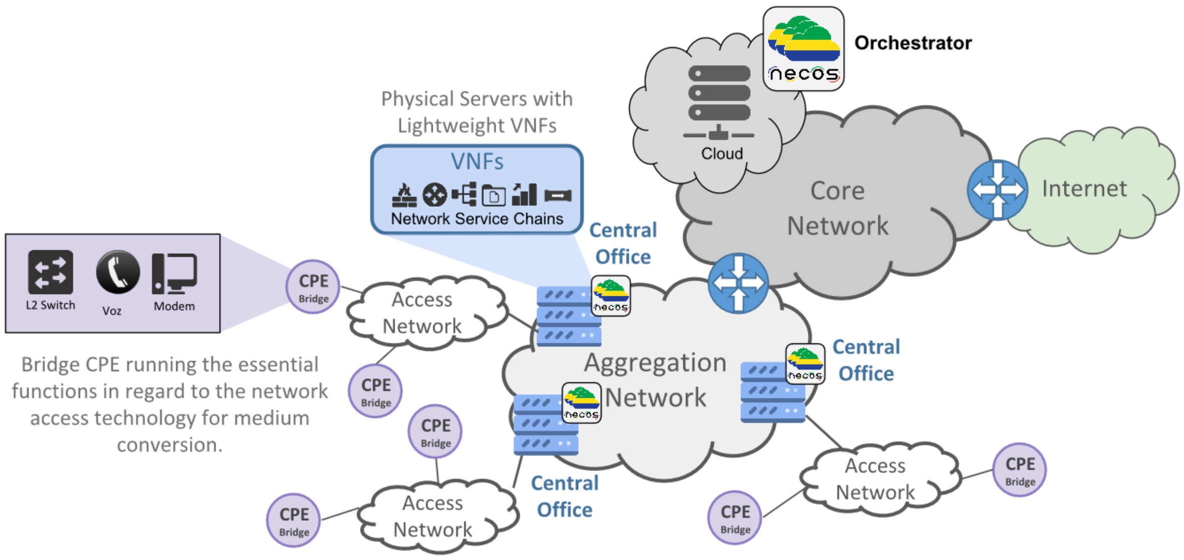

Orchestrator
Cloud
Core Network
Internet
Physical Servers with Lightweight VNFs
VNFs
Network Service Chains
Central Office
L2 Switch
Voz
Modem
CPE Bridge
Access Network
Aggregation Network
Bridge CPE running the essential functions in regard to the network access technology for medium conversion.

Figure A 4. vCPE deployment scenario

The virtual CPE enables the evolution of Central Offices by applying the key concept of Telco Clouds: the integration of both non-virtualized equipment (bridge CPE) and virtualized elements (virtual network functions), enabling new services to be deployed in an automated fashion. The vCPE services can also be deployed at the Edge Cloud, near to the end-user, in order to reduce latency and improve the user experience.

Although vCPE can bring many benefits, the implementations have faced the challenge of providing a high scalable platform without increasing latency and network bottlenecks. In this context, NECOS is an optimized solution that brings this scenario into reality by providing a deep integration of DCs and networking systems.

Decoupling the owner of the physical infrastructure (infrastructure provider) from who runs and manages the service (service provider) can lead to innovation, allowing new business models and a reduction in the complexity of running services. Since the world of computing has a similar experience, a relevant model can be applied to networking by using the cloud infrastructure to provide virtualized network services.

The infrastructure provider, which owns the cloud and network infrastructure, will be responsible for the management of the life cycle of the network slices and to guarantee the individual service-level agreement (quality, availability, responsibilities) for each slice. The service provider, which owns the slice, can use the slice to provide Internet access for its customers through virtual network functions (VNFs). Multiple service providers can share the same cloud infrastructure in order to meet the demand for network services in different niches, such as residential access, smart factory, agribusiness, smart cities and so on. The Figure A5 describes the scenario where multiple Virtual Internet Service Providers (Virtual ISPs) share the same infrastructure through fully isolated slices in order to provide Virtual CPE service.

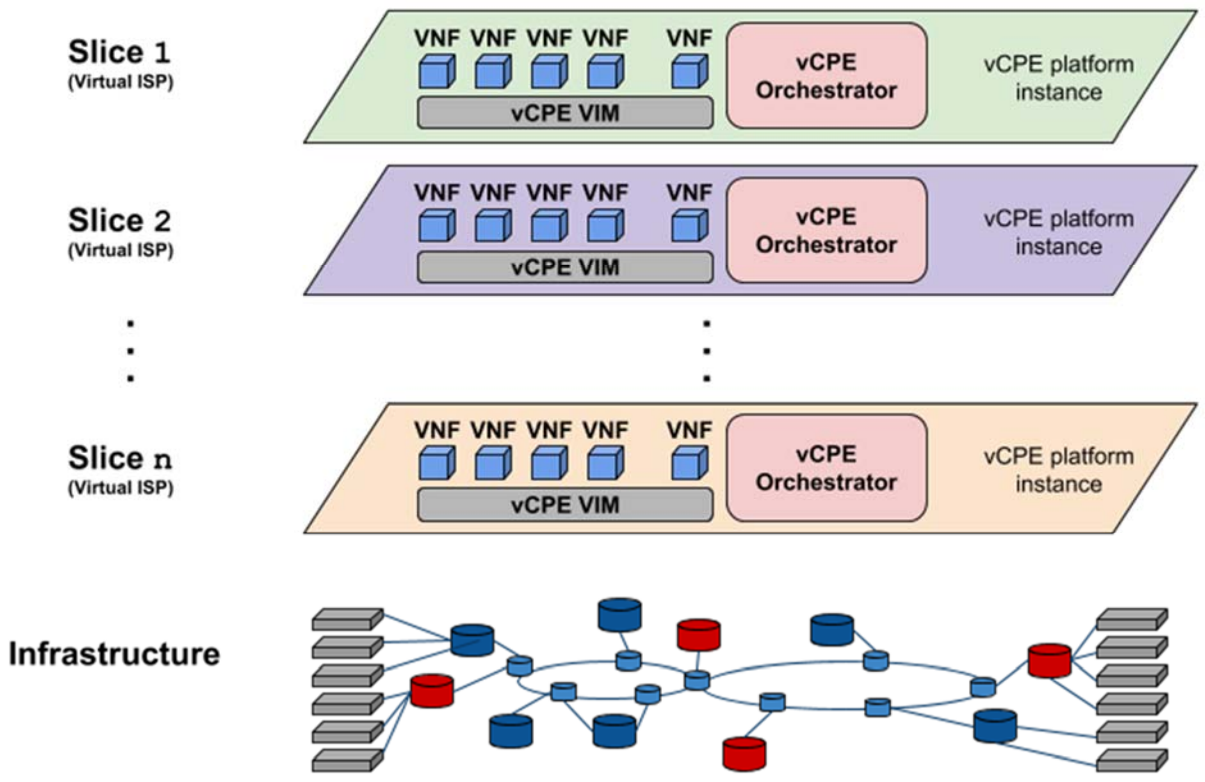


Figure A 5. vCPE deployment scenario with multiple instances

A network operator can play both roles, such as: *(a)* provide network services to the end user through internal slices within its infrastructure (e.g., Virtual CPE); and *(b)* offer unused resources as external slices for third-party companies providing network services, such as Virtual ISPs, e.g., as shown in Figure A5.

Different slicing models can be applied, as represented in Figure A6. For example, the external slice can be managed by the network operator and the latter can delegate the service management to the Virtual ISPs. So, the virtual ISPs can leverage the geographically distributed infrastructure from the network operator, instead of building its own infrastructure. Besides, the Virtual ISP does not have to deal with the complexity of managing the infrastructure and the slice.

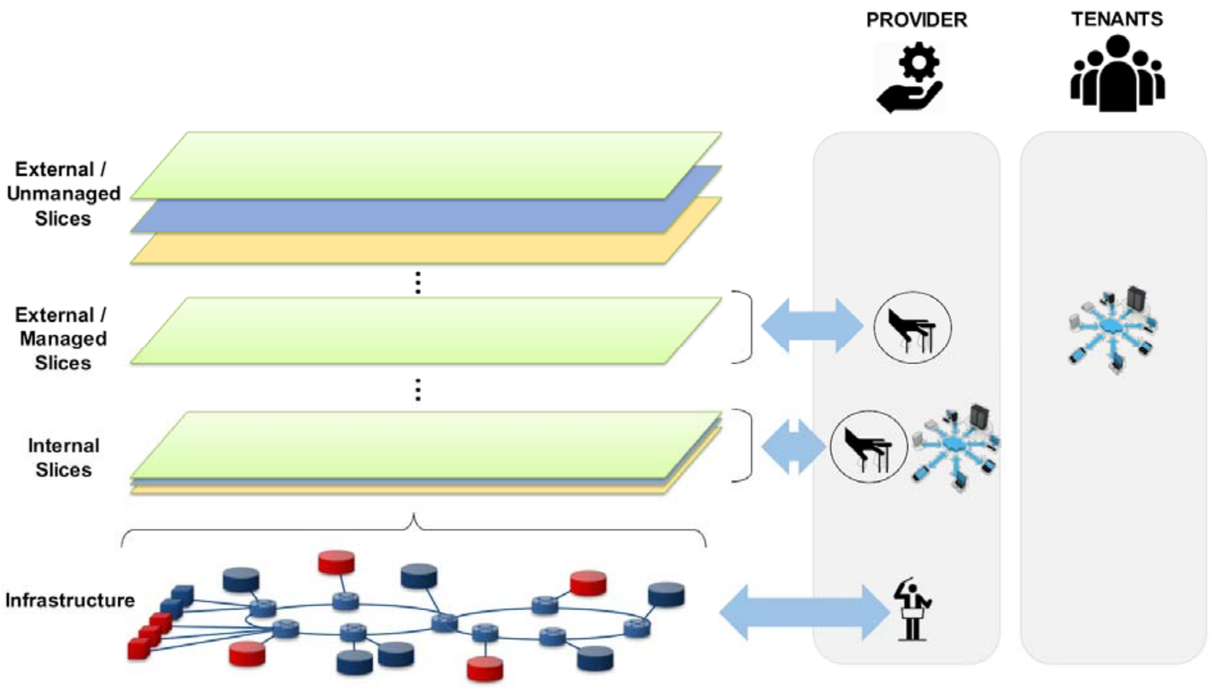


Figure A 6. Slice management in vCPE scenario

NECOS will be deployed by federated network operators, which can offer communication services in two ways: (a) provide Virtual CPE to their customers through internal slices; and (b) provide Virtual CPE as a service to ISPs via external slices. The Virtual CPE service can also come with basic VNFs, so the ISPs can focus on value-added VNFs regarding its niche market.

3.2 Technical enablers

The following technical enablers can be identified.

- Lightweight virtualization based on containers helps the system to achieve more efficient resource allocation, allowing the Virtual CPE platform to scale.
- Dynamic deployment of new network functions in a distributed and virtualized environment, without the intervention of the network operator.
- Network data plane programmability creates a much more agile, flexible and automated network, allowing the platform to build network services chains dynamically.
- Network I/O optimization technologies can accelerate overall packet processing performance in software, enabling more efficient VNFs, with a higher throughput.
- Automatic monitoring of physical and network resources to enable workload changes in an automatic manner.

3.3 Critical success factors and KPIs

The following factors are identified as critical to success of this scenario:

- F1 - High degree of scalability (variable and dynamic number of users), with a short-time response for the deployment on the field;
- F2 - High performance standards (low jitter, low packet error rate, high availability, redundancy mechanisms, and priority schemes implementation);
- F3 - Slice management (cross-layered approach for controlling, monitoring, analysis and long-time operations); and
- F4 - Easy network elements configuration.

To accomplish this goal, several KPIs must be addressed, like:

- K1 - Average end-to-end delay (in milliseconds, measured as half average RTT); Metric – average delay.
- K2 - End-to-end slice availability (% of time); Metric – availability.
- K3 - Average slice provisioning time (in seconds); Metrics – provisioning time, decommission time.
- K4 - Performance isolation index to identify performance impact between slices in a multi-tenant environment; Metric – isolation degree.
- K5 - Average elasticity response time (in seconds) taken to provision of new resources based on real-time demand; Metric – response time.
- K6 - Average throughput (in Mbps); Metric – throughput.

3.4 Mapping to NECOS key characteristics

The following NECOS characteristics are identified as key enablers of this scenario:

- C1: Slice-as-Service model dynamically allocates and deallocates slices on-demand.
- C2: Bare-metal VIM-independent slicing makes the slice fully manageable to deploy any kind of service.
- C3: Lightweight management and virtualization systems deployable on large number of small servers and clouds at the network core and edges.
- C4: End-to-end slice provisioning enables the deployment of services across physical resources from federated cloud networking infrastructures.

3.5 Requirements

The following functional requirements were identified for the virtual CPE scenario.

- RF.vCPE.1: *On-demand slice provisioning*. Each Virtual ISP will operate using a slice, thus the slice needs to be dynamically created and removed when requested by the customer.
- RF.vCPE.2: *Manageable slice*. Virtual ISPs need to manage the slice to deploy their own VNFs and to build the network service chains.
- RF.vCPE.3: *VIM-independence*. A Virtual CPE platform has its own orchestrator and VIM to deploy the VNFs and to build the network service chains. The service platform should be able to run its own VIM and orchestrator.
- RF.vCPE.4: *Bare-metal slice*. The VNFs will be deployed within the slice. If the slice is composed of virtual resources, there will be two levels of virtualization, which brings a huge impact on performance. In this context, bare-metals should be allocated to the slice.
- RF.vCPE.5: *Lightweight virtualization*. A Virtual ISP can have hundreds of thousands of subscribers, so lightweight virtualization is important to achieve scalability.
- RF.vCPE.6: *Elasticity*. The slice should be able to adapt to workload changes by provisioning and de-provisioning resources in an autonomic manner.
- RF.vCPE.7: *Zero touch service provisioning*. The deployment of the Virtual CPE service to the slice has to be automated, so the Virtual ISP does not need to deploy it manually.
- RF.vCPE.8: *Fault detection*. NECOS should be able to detect critical failures in the slice resources and notify the services that are running within the slice.

Additionally, the following non-functional requirements were identified for this same scenario.

- RN.vCPE.1: Isolation of slice resources. The Virtual ISPs will be allocated to different slices, so there must be complete isolation among the slices.
- RN.vCPE.2: SLA monitoring (QoS). It is important for the Virtual ISP to have the SLA's guarantee to provide the service to its subscribers.
- RN.vCPE.3: Low latency. All network traffic from the subscribers will travel through the Virtual ISP slice before reaching the Internet. The Virtual CPE service should minimize the impact as much as possible on connection's latency.
- RN.vCPE.4: High throughput. The resources allocated to the slice should provide a high throughput when processing network packets, considering that a Virtual ISP can have a large number of subscribers with high bandwidth.

EUB-01-2017

- RN.vCPE.5: High availability. The Virtual CPE service should operate continuously for a long time, so failures in the slice level should be automatically handled without impacting the service.

4. Network Slicing for Touristic Content Distribution

High profile Metropolitan areas attract a significant number of visitors each year. The scenario involves a Metropolitan Tourist Centre (MTC), that aims to offer state-of-the-art location-aware cultural content information services in order to enhance their visiting experience. These services will be offered: (i) within public transport vehicles (buses, subway, etc.), as visitors move throughout the city; and (ii) at various city locations (e.g., squares, museums) possibly taking advantage of public Wi-Fi infrastructure.

4.1 Description

The ubiquity of visitor mobile devices leads to a significant demand for state-of-the-art location-aware cultural content delivery (CD). The Metropolitan Tourist Centre (MTC) needs to provide high-quality network services (e.g., location-dependent content) to tourists with adaptable behavior to the dynamic network conditions and particular resource constraints and QoS requirements, focusing also to user mobility.

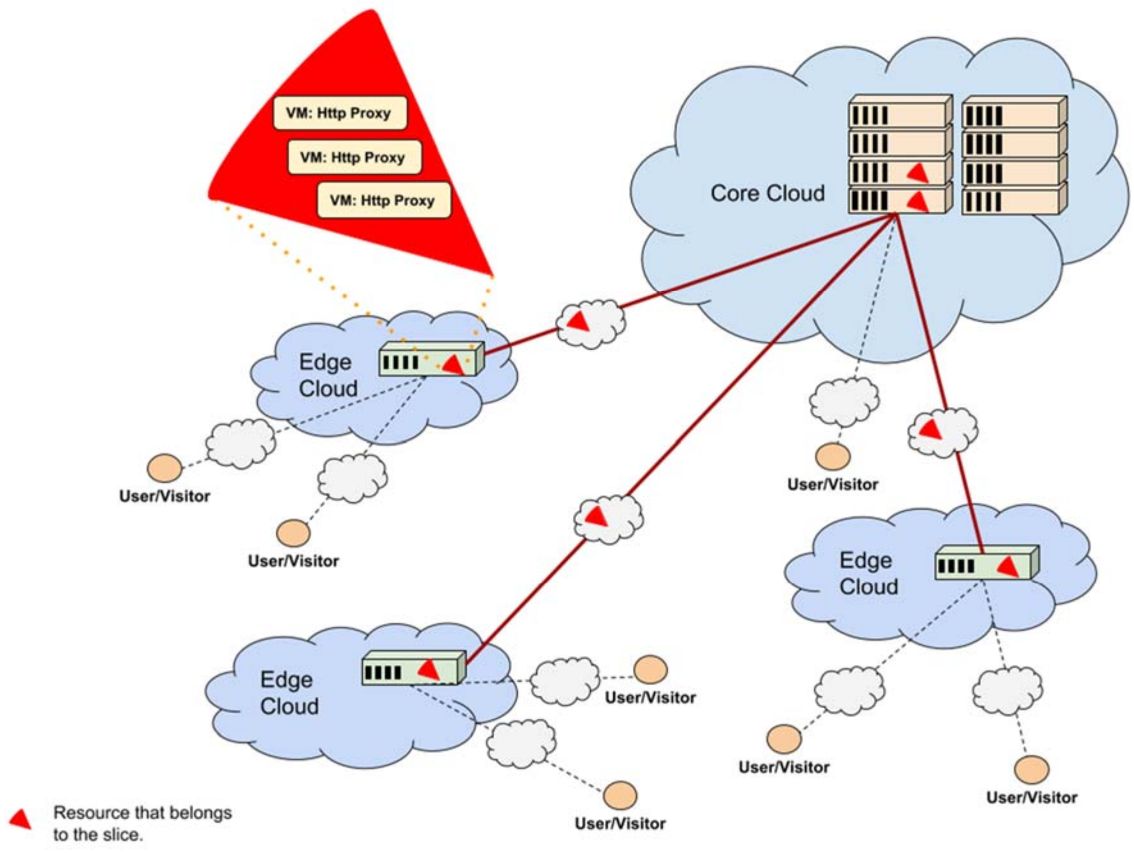


Figure A 7. Touristic Content Distribution as a NECOS slice

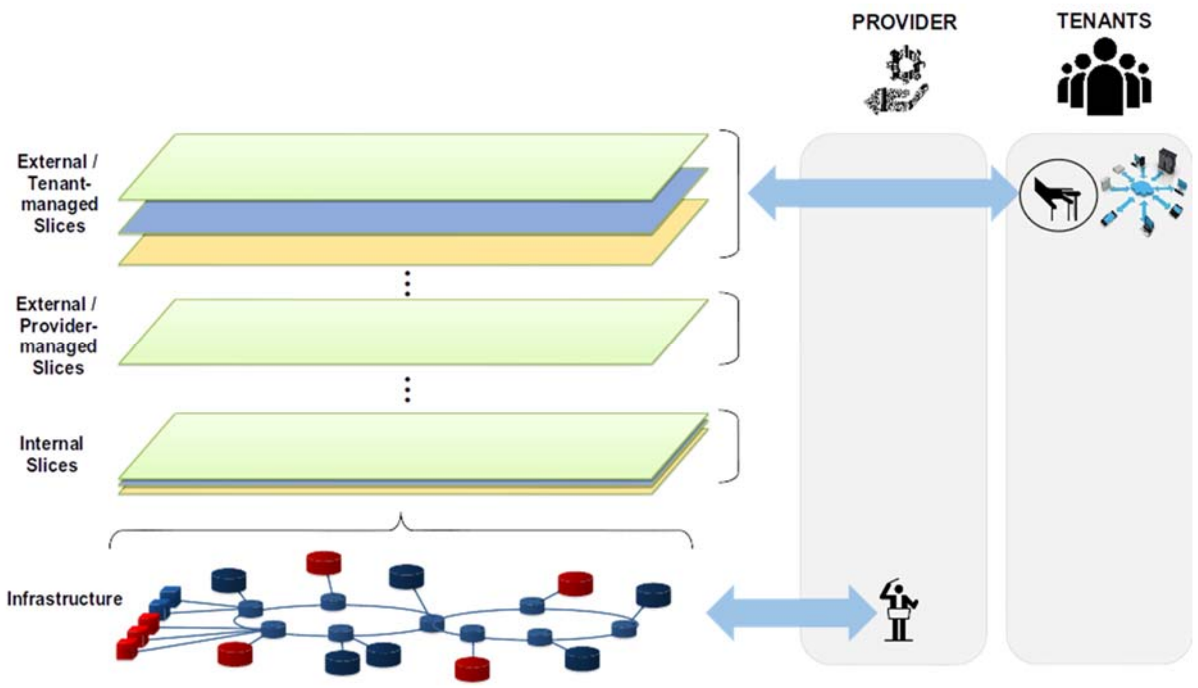
In the following, we describe the different perspectives of both end-users and MTC.

End-user perspective: User Victor decides to take either a public bus or a classic hop-on-hop-off bus to have a tour of the city. Moving throughout the city, Victor downloads content related to cities monuments near his current location, e.g. text, images (Service 1) and video (Service 2). Victor, being an active social networks user, posts comments regarding his status, uploads photos, and accesses information for places of interest nearby (coffee shops, restaurants) in the Touristic Social Network (Service 3).

MTC perspective (Slice Tenant): To address the above scenario, the MTC has deployed a number of lightweight VMs hosting web servers, which offer area specific information (text, images, videos), on edge cloud servers near the Piazza. Updated content is pushed to these VMs from a central server accommodated in the same slice (Figure A7). Since the edge cloud has limited resources, content pushed is based on requests originating from visitors, thus adapting to dynamic patterns of demands. These traffic data originate from the slice's monitoring facilities and used to perform load balancing in the CD service. The main aim from the MTC point of view is to reduce the resource consumption and in turn the cost, while satisfying the end-users' requirements.

Slice Provider Perspective: The slice-as-a-service provider offers the slice by combining edge and core cloud resources and the network infrastructure to interconnect them. The former (i.e., the edge cloud) hosts content proxies in the form of VMs on-demand and the latter (i.e., the core cloud) hosts the content. A key aspect here is the resource-efficiency of the physical servers, while providing the resources for the deployed slices. The infrastructure providers are responsible to offer a pool of resources on demand and support the slice elasticity aspects.

For this scenario, we consider one slice/per tenant, i.e., the MTC will have some control over the slice through a set of operations/configuration actions and the slice provider will segregate the necessary infrastructure (i.e. see Figure A8).


PROVIDER
TENANTS
External / Tenant-managed Slices
External / Provider-managed Slices
Internal Slices
Infrastructure
Figure A 8. Slice management in Content Distribution

User mobility is an important aspect in this scenario, as shown in Figure A8. For example, user A switches between different edge clouds, as he/she moves around the city, and utilizes sliced resources allocated on demand, i.e., the user downloads Internet content with ultra-low delays due to the locally cached content proxies that follows him/her. The content proxies are hosted in lightweight Virtual Machines (VMs) that can be deployed rapidly. As a bottom line, this scenario instantiates the Multi-access Edge Computing (MEC) use-case in a mobility-aware elastic content distribution context, i.e., implementing the NECOS slice as a service features for touristic content distribution.

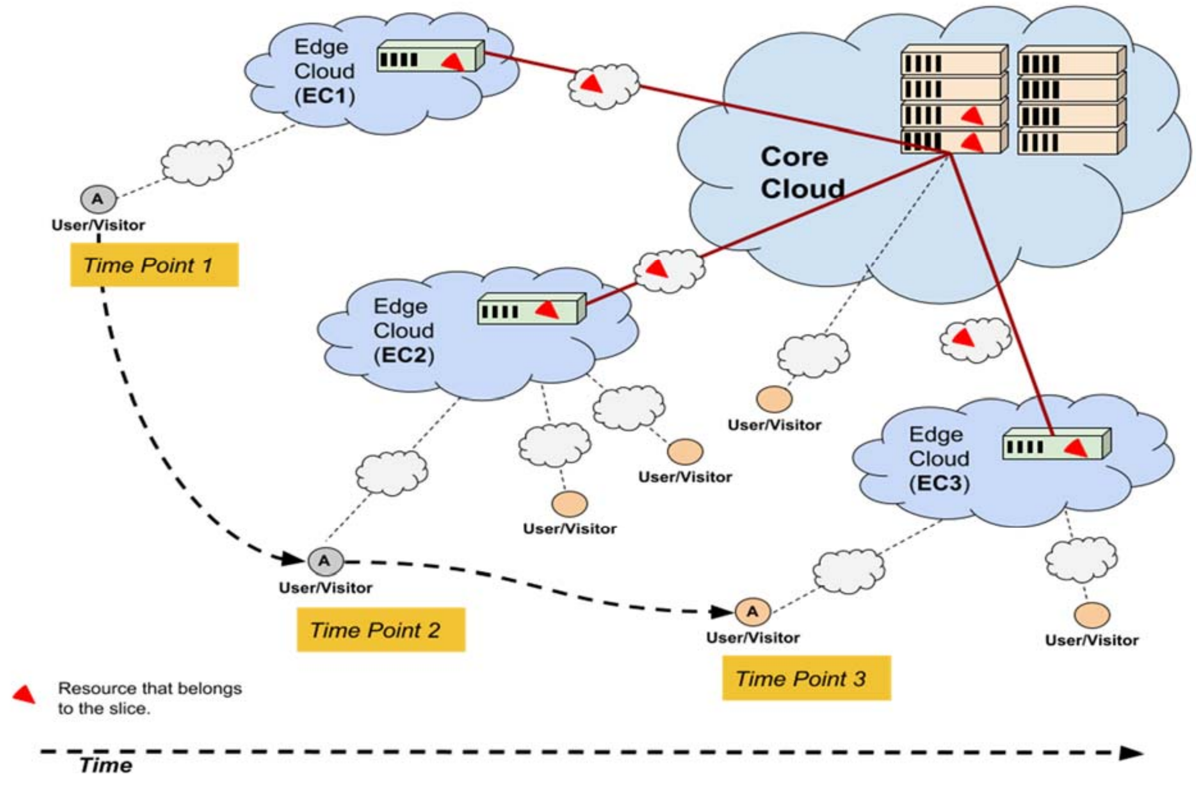


Figure A 9. User Mobility and Association with Multiple Edge Clouds

4.2 Technical enablers

In order to realize this scenario, the platform should provide:

- Flexible network technologies that allow traffic load balancing, for associating the end-users (visitors) with the appropriate http proxies, deployed at the edge clouds.
- Network and physical server monitoring facilities that enable intelligent decisions for http proxy deployment / management and efficient slice operation.
- Lightweight virtualization technologies for hosting Content Distribution (CD) services on the edge cloud that can be easily deployed and managed.
- Efficient and simple slice manipulation facilities allowing the implementation of alternative elastic CD approaches.
- Mobility handling features to allocate content proxies to end-users as they move around a wider geographical area, while avoiding the service disruptions due to the involved handovers between different edge clouds, as depicted in Figure A9.

4.3 Critical success factors and KPIs

The Network Slicing for the Touristic Content Distribution scenario is assessed from the point of view of the involved stakeholders. Relevant critical success factors are:

- F1 - Quality of content provisioning, i.e., the delay the visitor experiences when retrieving a webpage, or uploading photos, should be under application-specific limits. This allows an enhanced viewing experience to the user (Slice Efficiency);
- F2 - Heterogeneous resource management over multiple VIMs deployed at both edge and core clouds. Especially in the case of edge clouds, the resource utilization should be as efficient as possible, allowing the slice provider and the MTC to offer quality services with a minimum cost (Slice Efficiency / Cost Reduction);
- F3 - Elastic slice operation targeting a large number of supported end-users without compromising the level of provided service (Elasticity); and

- F4 - User mobility handling aspects that enable new touristic applications with locality considerations (Flexibility).

The above success factors can be quantified using a number of KPIs, such as the following:

- K1 - Content delivery performance to highlight the end-user satisfaction, e.g., for video streaming content, such as video start-up time, buffering percentage, switching between bit rates, etc.; Metrics – end-user QoE.
- K2 - Overall consistency to SLA index, expressing the Quality of Service to demonstrate the efficient network performance over heterogeneous resources with respect to the SLA; Metric – SLA fulfilment index.
- K3 - Physical server utilization to quantify the resource-efficiency of the NECOS Slice as a Service capability in the touristic content distribution scenario, while considering the case of multiple coexisting VIMs (i.e., CPU utilization, memory allocation, link utilization); Metrics – physical server utilization.
- K4 - Service disruption index to evaluate the efficient mobility handling, i.e., expressing end-to-end availability (% of time in which a ping gets a response over total time); Metrics – service disruption index.

4.4 Mapping to NECOS key characteristics

The aim of the scenario is to utilize the NECOS platform (NECOS objective 1) to demonstrate a novel Service Provisioning (NECOS Objective 2), i.e., elastic content distribution in a Multi-access Edge Computing context, through Uniform and Efficient management of infrastructure resources (NECOS Objective 3), with emphasis on automatic reallocation of resources and services across geographically distributed and computing/storage/networking infrastructures. The scenario validates the full impact of the NECOS approach to the network slicing (NECOS Objective 4).

We consider the following key NECOS characteristics for the scenario:

- C1: The MTCs can create, based on the **slice-as-service model** (Characteristic 1), a novel elastic content delivery platform for geo-specific content that is adaptable to the number of expected visitors. For example, the deployed slices can scale up when large scheduled events occur (e.g., New Year's Eve) or in an "on-demand" fashion.
- C2: Since the MTC services require both edge and core cloud resources, thus the ability to **configure slices across heterogeneous physical resources and VIM technologies** (Characteristic 2) is critical.
- C3: The MTC **manages everything via software** (Characteristic 3), including network resource allocation, monitoring slice/service performance and the slices' (re-)configuration. For instance, the MTC can dynamically adapt to an unexpected large number of requests for popular content or to support user mobility.
- C4: Content delivery http proxies are based in **lightweight virtualization technologies**, deployed on small edge cloud servers, closely operating with core cloud content delivery services. Such a deployment requires a **unified view and management of core cloud, edge cloud and network components** (Characteristic 4), a key NECOS platform characteristic.

4.5 Functional and Non-functional Requirements

The following are considered to be functional requirements for the specific scenario:

- *RF.Touristic(CD).1: Slice and slice-resource management*. Definition and lifecycle management of a slice as a service over a certain geographical area that includes both core and edge-cloud resources. The MTC sets up the desired slices around areas of high interest in its city.

- *RF.Touristic(CD).2: Automated Virtual Machine deployment.* Deployment of lightweight and regular VMs on particular slices, based on real-time and historical monitored data. This involves the creation/deletion of VMs on strategically selected physical nodes.
- *RF.Touristic(CD).3: Traffic load-balancing for content delivery*. Automated assignment of lightweight VMs providing cached content to Users, i.e., load-balancing of new bursts of user requests to newly created content-hosting VMs or to follow end-users as they move.
- *RF.Touristic(CD).4: Slice resource and service monitoring*. Monitoring of resource specific traffic patterns/service usage to detect changes in service requests demands.
- *RF.Touristic(CD).5: Service planning.* The MTC should be able to request additional resources to be included in its slice, to cope with occasional circumstances in which high traffic is anticipated. For instance, scheduled festivities that attract an increased number of visitors (e.g., the New Year's Eve) require more infrastructure resources for a relatively short period of time.

Respect to non-functional requirements, these are the ones identified so far:

- *RN.Touristic(CD).1: Transparent end-user performance*. End-user performance transparency, with respect to service usage, i.e. visitors should be unaware of changes due to load-balancing actions (introduction of new VMs, changes in routing, etc.), and experience minimal service delays and interruptions.
- *RN.Touristic(CD).2: Heterogeneity handling*. The service provisioning should be transparent with respect to details of the physical infrastructure, VIM used, load balancing and slice management, i.e. the MTC must "see" and control slice resources in a uniform manner, regardless of the underlying (different) technologies used.
- *RN.Touristic(CD).3: Elasticity*. The slice as a service should adapt to changes in the network (e.g., delays due to congestion) or user context (e.g., new end-users appear, content becomes viral etc.).
- *RN.Touristic(CD).4: Resource-efficiency*. The slice resources should be efficiently utilized, i.e., no over-provisioning of resources.
- *RN.Touristic(CD).5: Scalability*. Touristic content distribution should be scalable, to offer quality services to a large-number of users.

5. Multi-Domain Network Slicing for Next Generation Touristic Applications

Next generation touristic services involve augmented reality (AR) applications that offer a technologically advanced, enhanced travelers' experience. These applications aim to increase the value of the touristic product of high-profile metropolitan areas, by adding a layer of content to locations captured by the traveler's mobile device (e.g., smartphone or PDA). It is expected that municipal authorities are going to adopt the provisioning of such services in the near future. However, augmented reality applications demand significant CPU power for image processing and content to annotate camera video, that is either not available in the (visitor's) mobile device or will overtax the device's battery.

In the NECOS project, all CPU-intensive processing tasks should be delegated to nearby edge or core clouds, offering AR applications as a service.

5.1 Description

This scenario assumes a Metropolitan Tourist Centre (MTC) acting as a main provider of high-quality network services (e.g., location-dependent content) to tourists with particular resource constraints and QoS requirements, but now focusing on next-generation touristic services, such as those utilizing augmented reality.

Victor is the *end-user* of the services residing in the slice, while the MTC is the *Public Service Customer* (i.e., slice as a service tenant), that has setup all the above Services in appropriate slices through the *Slice Provider*. In the following, we describe the different perspectives of both end-users (visitors) and MTC.

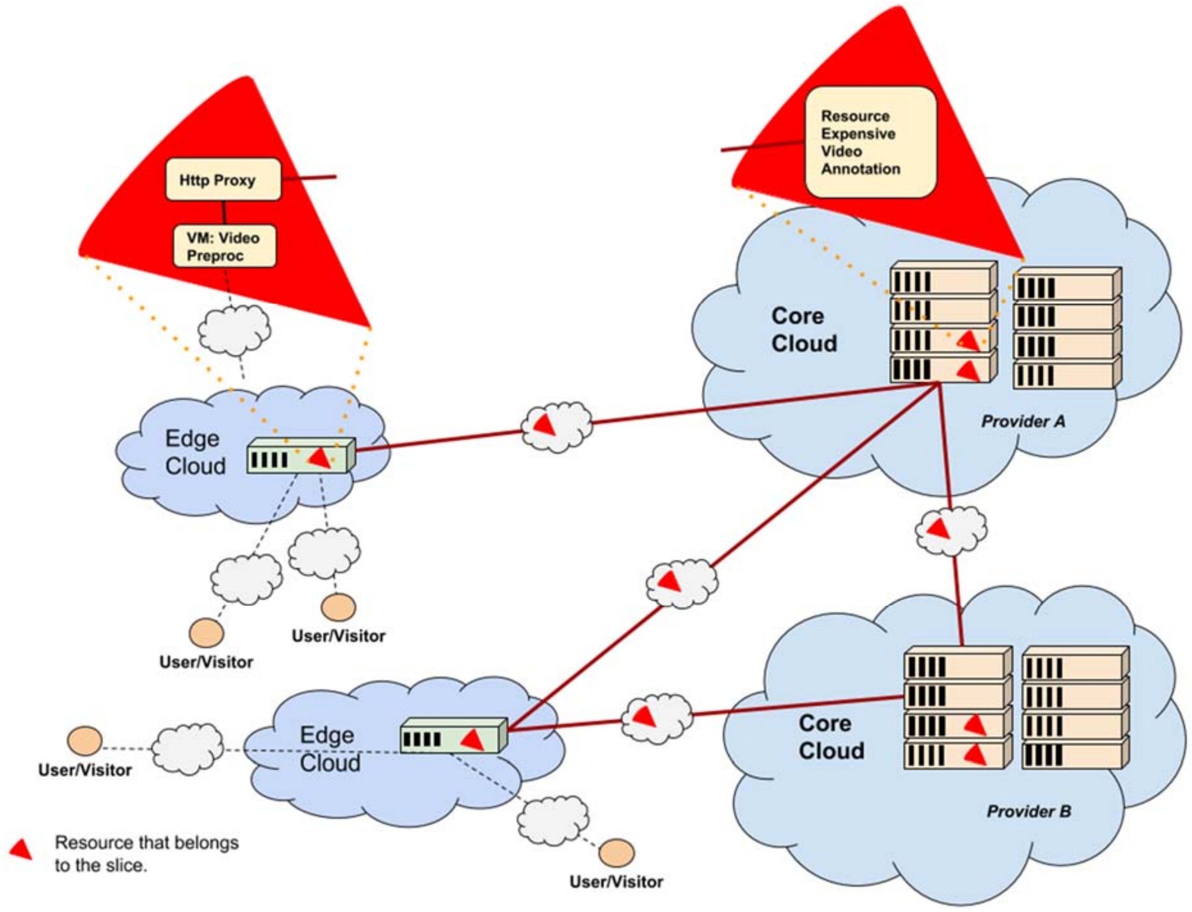


Figure A 10. Augmented Reality Touristic Services in the Multi-Domain Cloud

End-User Perspective: User/visitor Victor arrives at touristic location Piazza, where monument(s) Cathedral is located. Upon arrival, Victor downloads from the MTC web server a map to navigate in the Piazza, information (e.g., historical and architectural data) regarding the Cathedral (Service 1), related videos and sound recordings (Service 2). Victor points his cell phone to the facade of the Cathedral and the Virtual Guide app displays on his screen (augmented reality) information regarding particular architectural details (Service 2). Being fascinated by the facade's beauty posts his selfie, status and comments on the metro Social platform (Service 3). Once his visit is completed, he checks traffic conditions (Service 4) to decide whether to continue his tour to another place of interest or rest at a Piazza cafe (Service 3). If he decides to follow the first option, then upon his arrival to the new location, he will be offered a similar set of services, connecting to its new location edge cloud (possibly to a different one than that of the Piazza).

MTC Perspective: The MTC (tenant) has deployed a slice hosting a number of VMs that realize a touristic augmented reality application. Augmented reality content delivery is a multistep process that can be implemented with service function chaining, i.e., pre-processing of content input data is performed on a dedicated VM (located on the edge cloud), then the data is transmitted to a cloud server for computationally intensive processing and finally the enriched content is sent back through the edge cloud VM to the end-user. This communication involves a number of VMs deployed to both edge and core clouds, owned possibly by different cloud providers, so multi-domain orchestration is a main requirement.

Slice Broker Perspective: The slice broker sets up and manages the slice involving resources from multiple providers. Since the connectivity between different edge clouds and core cloud servers requires high link capacity and ultra-low delay, a single provider may not be present in all city areas. The involved data center infrastructure and network connectivity providers are motivated to participate in the resource federation through the reduced cost due to the efficient resource allocation and new novel touristic services that bring profit. A market-place type of resource offering and reservation balances the requirements and directions of different cloud providers.

In this case, we still consider one slice/per tenant, i.e., the MTC will be the single tenant in the slice, but due to the federation the slice control is on the slice (broker) provider side.

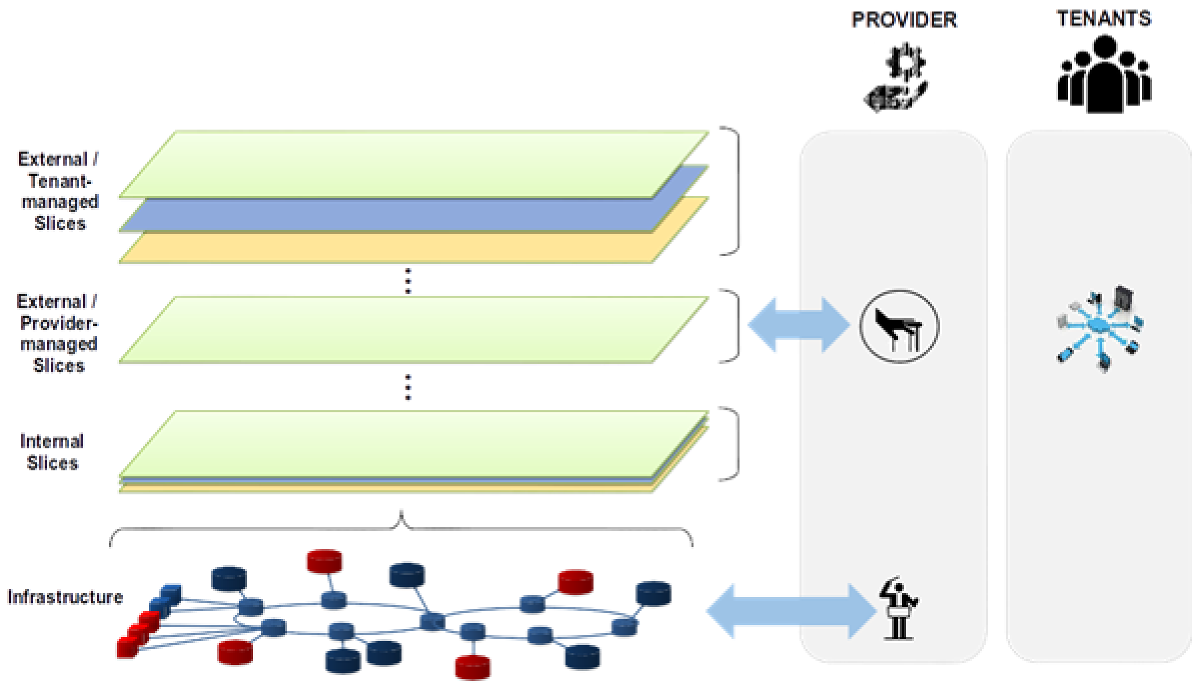

PROVIDER
TENANTS
External / Tenant-managed Slices
External / Provider-managed Slices
Internal Slices
Infrastructure

Figure A 11. Slice Management in AR Service

This is another instantiation of the Multi-access Edge Computing use-case that pushes the boundaries of the "Network Slicing for Touristic Content Delivery" scenario towards multi-domain orchestration and heterogeneous resources' federation at edge and core clouds (i.e., for both processing and communication). Its main focus is to provide novel next-generation touristic services.

5.2 Technical enablers

We see the following technical enablers in this scenario:

- Discovery and allocation of resources to form a slice over multiple domains, respecting a number of constraints, such as geographical location, bandwidth and delay connectivity demands, CPU requirements, etc.
- Flexible network softwarization technologies that allow efficient traffic load balancing that improves resource utilization of both network and cloud servers.
- Network-Function Virtualization technologies to support the service chaining necessary for the operation of the augmented reality service (i.e., VNFs in both edge and core clouds).
- Multi-domain orchestration facilities for the efficient slice manipulation over federated resources that allow a unified view of the underlying resources in the created slice.
- Intelligent orchestration techniques and prediction mechanisms on expected usage regarding both infrastructure resources and end-user demands. The latter aim to support advanced techniques in multi-domain orchestration, by detecting patterns of service usage, that indicate changes in demands in order to intelligently infer resource orchestration decisions.
- Network and physical server monitoring facilities that enable prediction mechanisms and decision-making regarding elasticity and efficient multi-domain slice operation.

5.3 Critical success factors and KPIs

The critical success factors of the "multi-domain network slicing for next generation touristic applications" scenario follow:

- F1 - Quality of service in next-generation touristic applications, i.e., the slice should sustain the *large traffic volumes* and offer the *processing* required for video streaming and augmented reality, in order for the end-users to enjoy real-time, low-latency and low jitter service delivery (Slice Efficiency);
- F2 - Resource efficiency and balanced resource allocation between the different infrastructure providers, motivating different providers to realize multi-domain slices (Slice Efficiency);
- F3 - Scalability and reduced cost in slice setup and management that will allow infrastructure providers to extend their client-base and the MTCs to offer quality services to more citizens (Cost Reduction/Scalability);
- F4 - Efficient resource offloading between the different clouds to support a large number of users, i.e., the service should be able to scale to satisfy a significant number of visitors within a limited geographical area (Lifecycle Efficiency/Flexibility); and
- F5 - Intelligent multi-domain orchestration that supports prediction of service demands and resource utilization, to cope with time-varying service demands (Elasticity).

The above success factors can be quantified using KPIs, such as the following:

- K1 - AR application performance in terms of response time of receiving annotated video, to highlight the end-user's satisfaction, e.g., for video streaming AR applications such as: video start-up time, buffering percentage and switching number between bit rates, etc.; Metric – end-user QoE.
- K2 - Number of application users serviced within an area to demonstrate the scalability aspect; Metric – number of application users.
- K3 - Time to adapt network service provisioning through multi-domain orchestration, in cases of a sudden rapid increase in the service requests number; Metric – service adaptation time.
- K4 - Accuracy of prediction of service demands and resource utilization for the intelligence aspects of multi-domain orchestration. Metric - service demands prediction accuracy.
- K5 - Physical server utilization per cloud to quantify the resource-efficiency and the balanced resource offloading between the different cloud providers (i.e., CPU utilization, memory allocation, link utilization); Metric – physical server utilization.

5.4 Mapping to NECOS key characteristics

The scenario addresses all NECOS objectives and key characteristics. For example, the NECOS platform (Objective 1) enables novel next-generation touristic applications (Objective 2) through realizing multi-domain orchestration over multiple providers and heterogeneous clouds (Objective 3). Furthermore, it demonstrates all the main NECOS characteristics (Objective 4), such as:

- **C1: Slice as a Service provisioning** (Characteristic 1). Since the MTC has to specify and deploy its services in the cloud, through integrating Cloud and Edge cloud resources from multiple providers.
- **C2: Configuration of slices across the physical resources in the cloud** networking infrastructure to better accommodate various service demands (Characteristic 2). The slice is dynamically configured to support augmented reality applications that pose stringent requirements in terms of network and cloud server resources.
- **C3: Everything is managed via software** (Characteristic 3), from the network configuration and adaptability to the service function chain orchestration that enables novel next-generation touristic content delivery.
- C4: It supports **uniform management over lightweight virtualized resources** (Characteristic 4), through realizing service function chains over edge clouds with limited physical server

resource availability. In case of more demanding processing, a nearby core cloud, residing in the same or different infrastructure provider, is involved.

5.5 Functional and Non-functional Requirements

The functional requirements of this particular scenario follow:

- RF.Touristic(APP).1: Service function chain orchestration, i.e., to deploy and configure the service components realizing the augmented reality applications. For example, NECOS can support automated deployment of lightweight VM on slice resources based on real-time and historical monitored data.
- RF.Touristic(APP).2: Resource and user-demand prediction capabilities, i.e., driving the NECOS slice as a service capability and offering resource-efficient next-generation touristic services.
- RF.Touristic(APP).3: Resource offloading between edge, core clouds and cloud providers. The synergy between cloud providers and hierarchical clouds can make a range of new touristic services possible, e.g., ultra-low delay augmented reality applications.

We consider for the scenario the following non-functional requirements:

- RN.Touristic(APP).1: Resource federation and intelligent multi-domain orchestration, to enable novel touristic services that were not possible before.
- RN.Touristic(APP).2: Scalability. Touristic augmented reality services should be offered at a scalable manner to support a large-number of users.
- RN.Touristic(APP).3: Efficient next-generation touristic application performance. Next-generation touristic applications, such as augmented reality, are very resource-demanding.
- RN.Touristic(APP).4: Elasticity. Such novel resource-demanding applications can operate only over slices that adapt to changes in the network or user contexts.

6. Network Slicing for Metropolitan Integrated Monitoring

This scenario addresses a Command, Control and Communications (C3) Public Safety Centre (PSC), responsible for enhancing incident management and resolution to a significant number of first responders (e.g., police department, fire & rescue services, emergency medical services, and public works), as well as their interaction with civilian population in a metropolitan area. The C3 PSC aims at leveraging the information flow level between citizens, responders, and agencies, by quickly offering the ability to receive, correlate, and share information.

6.1 Description

The Network Slicing for Metropolitan Integrated Monitoring (MIM) aims at leveraging the information flow level between distinct public agency teams – namely, first responders deployed on the field and major staff authorities headquartered at C3 PSC buildings, by quickly offering the ability to receive, correlate, and share timely information in an ecosystem of lightweight edge clouds to be in support of the mobile user applications, paving the way of the Lightweight Slice Defined Cloud paradigm to demonstrate NECOS suitability to the aforementioned MEC use case.

To accomplish its goal, MIM scenario may utilize either (mobile) edge clouds deployed near crowded areas, police stations, fire department stations, town hall, and other public buildings, on an opportunistic manner or based on agency-owned infrastructure. Additionally, it may use larger cloud telco-operated infrastructure, maintaining all content (e.g., C3 state-owned applications, which gather data from sensors deployed on the field, as well as inputs from weather channels, traffic updates, and emergency alerts) required by the services offered to public authorities. Some of these features are described in Figure A12 depicted below:

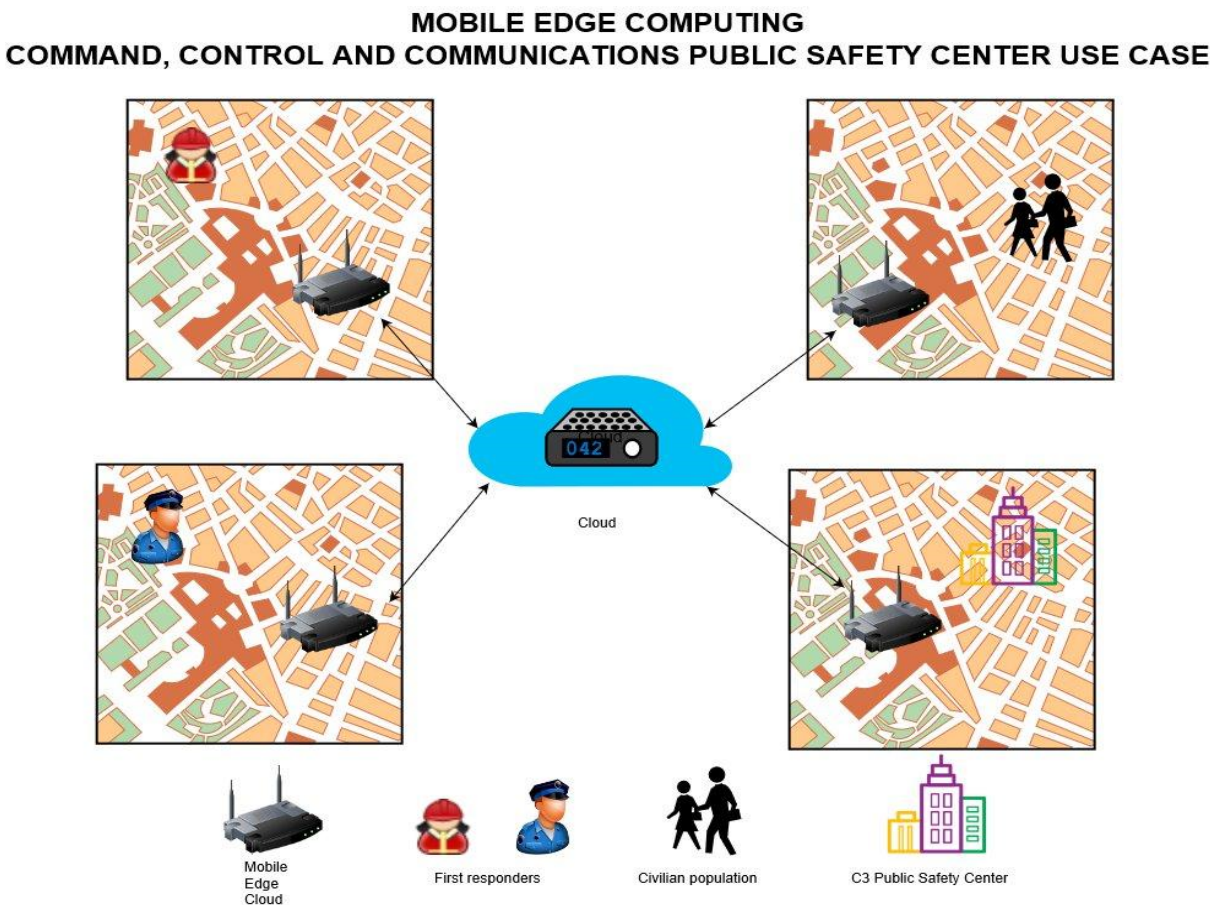


Figure A 12. Command, control and communications scenario

On the other hand, edge clouds caching will reduce latency to access data for the first responder's teams. NECOS will provide the necessary solutions for the C3 PSC to deploy MIM services over slices with adaptable behavior to the dynamic network conditions and application requirements, focusing on user mobility and quality of experience.

In order to cope with the services described above, one can envision the creation of internal and external (managed and/or unmanaged) slices. Moreover, several service and infrastructure providers shall be listed to provide the aforementioned features, responsible for content provision and data processing infrastructure, which can be from 4 different types:

In brief, the C3 PSC may offer the following services:

1. Service 1 - Basic community information, in the form of text and images sent to mobile devices using HTTP. Targeted audience: civilian population.
2. Service 2 - Advanced GIS mapping and location services, in the form of downloadable location specific information, including narration (sound) and video of safety incidents, targeted to first responder's mobile equipment, as well as to civilian mobile device (smartphone, tablet, etc.). Targeted audience: civilian population and first response officers.
3. Service 3 - C3 Network Services, that involves users sharing photos, videos and comments, related to their experience on different city spots, including reviews of criminal incidents, comments regarding suspect personnel, traffic updates, weather updates, etc. Targeted audience: civilian population and first response officers.

4. Service 4 - Advanced Analytics & Incident Management Services (i.e., IoT), that includes information sharing to/from the C3 PSC out to/from responders at the scene, gathering audio, text, image, video, and reports produced by responding agencies, citizens and officers into the command center, as well as by sensors/devices deployed on the field with the public safety staff. Targeted audience: first response officers and C3 PSC command staff.

Based on the preconditions stated above, there are several advantages on the adoption of network slicing mechanisms, since each user/group of users may behave as a real and independent network. Several and distinct perspectives can be foreseen at MIM scenario for different stakeholders, according to NECOS ecosystem categories:

Slice consumer perspective: User Julian, a policeman serving at an urban garrison, wants to get and share updated crime reports downtown with his hierarchical superiors based at C3 PSC. Moving throughout the city, Julian shares data to/from the C3 PSC out to/from responders at the scene, gathering audio, text, image, video, and reports produced by responding agencies, citizens and officers into the command center, as well as by sensors/devices deployed on the field with the public safety staff (C3 PSC Service 4). The main focus is to improve the Quality of Experience (QoE) of users like Julian.

Infrastructure Resource Provider Perspective: the infrastructure resource provider offers edge and core clouds, data center (e.g., compute, storage, network) and/or wired and wireless connectivity resources to build slices. We may list specific C3 requirements, ranging from advanced analytics with Augmented Reality support provided by C3 PSC headquarters to a plethora of field data provided by IoT sensors (e.g., IR camera, video, speed limit sensors) deployed on the field to provide the resources for the deployed NECOS slices.

Service provider perspective: Several public agencies shall be motivated to participate in the resource federation due to the novel analytics services upspring, enhancing their C3 capabilities through service/resource cooperation and data sharing, providing cost reduction due to service/resource/infrastructure sharing. The main goal from the C3 PSC point of view is to provide timely and consistent information to public agencies in general. Moreover, it shall accomplish resource consumption reduction with a low-cost approach.

6.2 Technical enablers

We see some boundary conditions in this scenario:

- Widespread usage of Software-Defined Networking technologies by operators and providers to support data security policy enforcement (e.g., between agency-owned core data centers and edge clouds) and traffic load balancing (e.g., between the end-users and the http proxies deployed at the edge clouds).
- Network-Function Virtualization technologies to support the service chaining providing C3 PSC analytics and incident management services (i.e., VNFs in both edge and core clouds).
- Automatic network and physical server monitoring by operators and providers to enable elasticity and efficient slice operation.
- Lightweight virtualization technologies (e.g., VLSP, unikernels or containers) for the edge clouds' virtual resources.
- Prediction mechanisms for the resource allocation and end-user requirements, to support the intelligence of the multi-domain orchestration.
- Intelligent multi-domain orchestration for the efficient slice manipulation over federated resources.

- Mobility handling aspects by operators and providers to allocate content proxies to end-users as they move around the area, while avoiding the service disruptions due to the involved handovers between different edge clouds.

Given the aforementioned support, we envision the following technical enablers:

- Dynamic slice creation and tear-down in any given MIM network slice.
- Services and capabilities definition and on-demand update supporting interest data retrieval.
- Support to slice merging, isolation, and modification, w/ on-demand capacity redefinition and function insertion/removal supporting slice consumer mobility and disruption.
- Slice monitoring and maintenance to accomplish service provision and resource management scalability supporting typical fluctuations on slice consumer population in a timely manner.
- Operation onto a multi-domain environment, to allow missions gathering several service providers (namely, public safety agencies), with previously agreed interfaces;
- Provision of an orchestration-enabled environment to deal with resource requests from different slice consumers.
- Infrastructure services provider support to typical device density in metropolitan areas to attend low speed vehicles and pedestrians
- High-level service and slice reliability, survivability, and availability, for different types of traffic to accomplish service customer requirements.
- Provision of a secure data exchange environment, enhanced by extreme levels of confidentiality, message authentication, operation registry and auditing, and detection of violation of authorized policies to deal with service provider security policy.
- Exposition of features like resource discovery, selection and allocation to support slice provision on a timely basis.

6.3 Critical success factors and KPIs

The Network Slicing for Metropolitan Intelligent Monitoring (MIM) scenario is assessed from the point of view of the involved stakeholders. In brief, the most critical success factor is situational awareness as well, which implies in relevant critical success factors as:

- F1 - Elastic slice operation providing a high degree of scalability (variable and dynamic number of users);
- F2 - Elastic slice operation providing short-time response for the deployment on the field;
- F3 - High QoE (low jitter, low packet error rate, high availability, redundancy mechanisms, and priority schemes implementation) for end-users;
- F4 - Network management (cross-layered approach for monitoring, analysis, and update) to endure long-time operations under different mobility patterns;
- F5 - Easy and low-cost network elements configuration;
- F6 - Security management based on proper data segregation and authentication patterns; and
- F7 - Intelligent multi-domain orchestration that supports prediction of service demands and resource utilization.

The above success factors may be derived in a set of KPIs, such as the following:

- K1 - Physical server utilization to quantify the resource-efficiency of the NECOS Slice as a Service capability in MIM scenario; Metric - physical server utilization.
- K2 - Content delivery performance to highlight the end-user satisfaction; Metric – end-user QoE.
- K3 - Service disruption to evaluate the efficient mobility handling; Metric – service disruption index;
- K4 – Timely and cost-effective slice deployment, conjugated with end-user QoE to demonstrate the efficient network performance; Metric – slice time deployment.

- K5 – Slice isolation to highlight data segregation and authenticity over a multi-domain environment; Metric – slice isolation index.
- K6 - Number of application users to demonstrate the scalability aspect; Metric – number of application users.
- K7 - Accuracy of prediction of service demands and resource utilization for the multi-domain orchestration; Metric – service demand prediction accuracy.

6.4 Mapping to NECOS key characteristics

The particular service needs for the scenario are the following:

- Secure content availability;
- Support for alleviating bulk data transfers of public safety data to first responders;
- Low latency content-delivery to first responder's teams by low-cost dynamic resource allocation according to end-user demands;
- E2E service chaining in the multi-domain cloud environment;
- Bandwidth guarantees to support different connectivity levels required by teams deployed on the field.

Thus, NECOS service provisioning approach (Objective 2) is suitable for MIM scenario, since it foresees the integration of resources within the collection of independent slices. A goal of the LSDC approach is to reduce the complexity and timescale for service provisioning and deployment in federated DCs, thus reducing the OPEX for the infrastructure owner.

Moreover, NECOS also envisions service and resource orchestration and management methods for the LSDC infrastructure resources that are located within and at the edge of the MIM network. This service orchestration and management approach, which comprises NECOS Objective 3, includes the automatic re-allocation of resources and services across distributed and geographically separated computing, data storage, and network infrastructures within separate slices to support requests for MIM application data.

Finally, NECOS objectives 1 (develop and build a Lightweight Slice Defined Cloud (LSDC) platform enabling computing, network, and storage elements in the cloud through the Slice as a Service across federated clouds) and 4 (a pilot-based impact demonstration) will be achieved as well to demonstrate the provision of location-aware public safety content delivery to distinct public agencies teams.

In order to accomplish the aforementioned objectives, we consider the following key NECOS characteristics for the scenario:

- C1: Slice as a Service, mapping enhancing component (re) configuration to support E2E MIM applications;
- C2: Lightweight Edge Cloud, with small footprint components, deployable on both core and edge small servers and cloud systems, to support a user environment characterized by mobility and disruption
- C3: Software Management under a per-slice approach, to support features like user demand elasticity (i.e., Load Balancing through multiple VMs) and monitoring.

6.5 Functional and Non-Functional Requirements

This section presents the main requirements for metropolitan intelligent monitoring domains. These requirements are listed under functional and non-functional scopes:

The functional requirements are:

- RF.MIM.1 – Dynamic slice management: support to dynamic slice creation and tear-down in any given PSC network slice.
- RF.MIM.2 – Dynamic service definition: support to services and capabilities definition and on-demand update.
- RF.MIM.3 – Timely slice management: support to slice monitoring and maintenance to accomplish service provision and resource management scalability.
- RF.MIM.4 – Orchestration: provision of orchestration capabilities to deal with resource requests from different users, under a multi-domain environment, to allow missions gathering several public safety agencies, with previously agreed interfaces.

On the other hand, the non-functional requirements considered are:

- RN.MIM.1 – High Reliability: provision of high-level service reliability (> xx %)
- RN.MIM.2 – High Availability: provision of high-level service availability (> xx %)
- RN.MIM.3 – High Survivability: provision of high-level service survivability (> xx %) for the scenario described in requirements RN.emergency.1 and RN.emergency.2

7. Network Slicing for Smart Cities Data Content Distribution

This scenario addresses the need to integrate different infrastructure provider in a city in order to provide data content delivery in a Smart City through the use of different infrastructure providers.

7.1 Description

According to United Nations 2014 World Urbanization Prospects, 54 % of the world's population lives in urban areas, a proportion that is expected to increase to 66 percent by 2050. Public services are one of the most relevant issues for this large population, whether in more developed centers or in developing countries. Thus, the widespread use of mobile devices by the population in urban areas leads to an increase in demand for novel techniques related to location-aware public content delivery. However, traditional solutions, ranging from SMS alerts to over-the-top smartphone applications, experience side-effects like useless data sharing, high delay response, and low availability rate. Similarly, topics like traffic, major sport and artistic events, floods, fires, manifestations and strikes present great relevance and interest. The Figure A13 presented below describes this concept, gathering several urban data sources onto a common ground named "smart city":

Figure A 13. SmartCity Data concept

In this context, smart services rely on low latency to offer essential real-time, such as autonomous drones that need incredibly fast response times but won't necessarily require high data rates. Contrary, cloud-based services rely on high data rates, but not necessarily need low latency. Additionally, the natural heterogeneity of the smart city demands that different infrastructure providers operate together to offer services for the citizens.

Given the scenario stated above, we introduce the Network Slicing for Smart Cities Data (SCD) Content Distribution scenario, which aims at leveraging the information flow level between citizens and services providers, by quickly offering the ability of receiving, correlating, and sharing timely information in an ecosystem of lightweight edge clouds to be in support of the mobile user applications, paving the way of the Lightweight Slice Defined Cloud paradigm to demonstrate NECOS suitability to the aforementioned MEC use case.

We claim that providers must "slice up" their 5G networks to meet the different requirements of smart city implementations as devices involved in the smart city could directly or indirectly have an impact on individuals' lives based on several services which are driven by conflicting metrics as capacity, latency, security, duration, reliability, and geographic coverage.

To accomplish its goal, the SCD scenario may utilize either (mobile) edge clouds deployed near crowded areas, museums, and other public buildings, on an opportunistic manner. Additionally, it may use larger cloud telco-operated infrastructure, maintaining all content (e.g., weather channel, traffic updates, and emergency alerts) required by the services offered to the population. On the other hand, edge clouds caching will reduce latency to access data for the urban population. NECOS will provide the necessary solutions to deploy SCD services over slices with adaptable behavior to the dynamic network conditions and application requirements, focusing on user mobility and quality of experience. Figure A14 presents a functional view of the envisioned system architecture, where NECOS comes up as a solution for multidomain end-to-end network slicing provision.

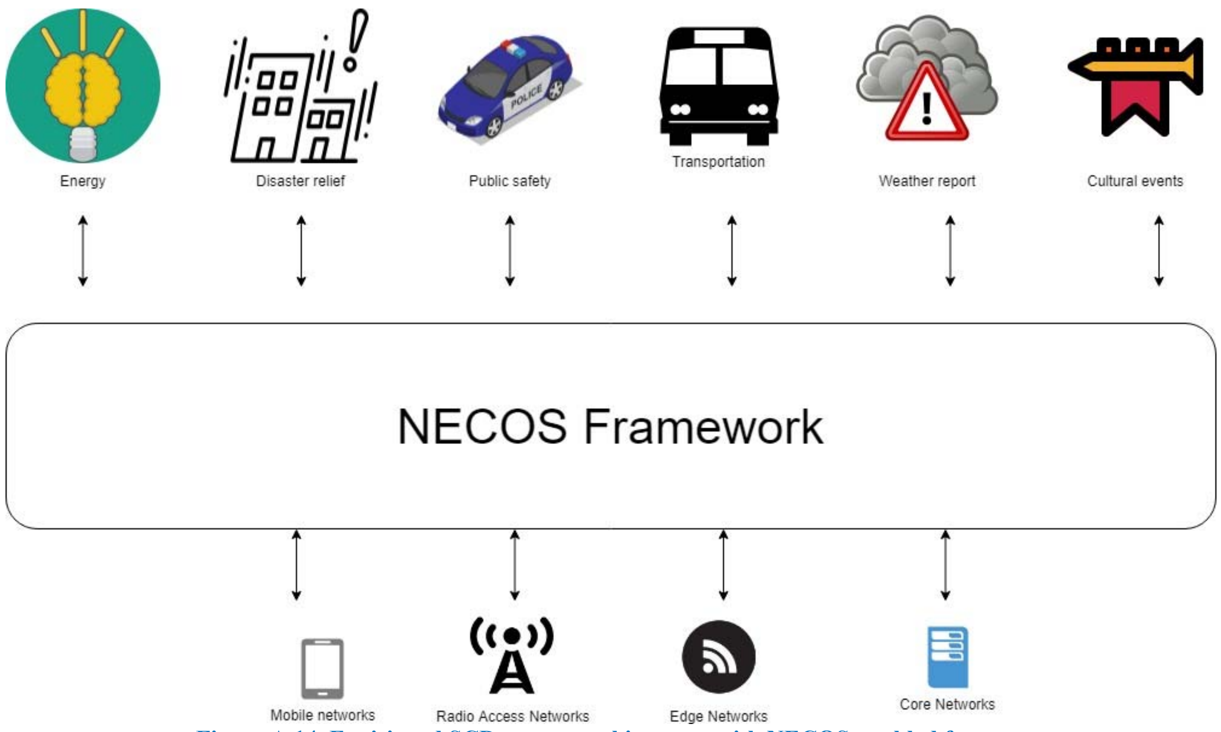


Figure A 14. Envisioned SCD system architecture, with NECOS-enabled features

Several and distinct perspectives can be foreseen at SCD scenario for different stakeholders, according to NECOS ecosystem categories:

- **Slice consumer perspective:** User Alice, living in a large urban center, wants to get updated traffic and incident reports to guide her daily home-to-work commuting. Moving throughout the city, Alice downloads content related to main points of interest near her current location, e.g., text, images and video. The main focus is to improve the Quality of Experience (QoE) of users like Alice.
- **Infrastructure Resource Provider Perspective:** The infrastructure resource provider offers edge and core clouds, data center (e.g., compute, storage, network) and/or wired and wireless connectivity resources to build slices. Several key aspects may be listed, like physical and virtual servers resource-efficiency, on-demand and timely resource offers to support NECOS elasticity aspects, stringent requirements for data isolation and integrity, and adoption of low-cost data center solutions to provide the resources for the deployed NECOS slices.
- **Service provider perspective:** To address the above scenario, the service providers deploy a number of lightweight VMs hosting web servers, which offer widespread area specific information (text, images, videos), on edge cloud servers scattered throughout the city. Updated content is pushed to these VMs from a central server accommodated in the same slice. The main goal is to provide timely and consistent information to population in general. Moreover, it shall accomplish resource consumption reduction with a low-cost approach.

7.2 Technical enablers

We see the following technical enablers in this scenario:

- Lightweight virtualization based on containers helps the system to achieve more efficient resource allocation, allowing the use of lightweight data center equipment.
- Dynamic deployment of new services instances in a distributed and virtualized environment, without the intervention of the network operator.
- Lightweight virtualization technologies for hosting Content Distribution (CD) services on the edge cloud that can be easily deployed and managed.
- Automatic monitoring of resources to enable workload changes in an automatic manner.

7.3 Critical success factors and KPIs:

The Network Slicing for Smart Cities Data Content Distribution (SCD) scenario is assessed from the point of view of the involved stakeholders. In brief, the most critical success factor is situational awareness - "all knowledge that is accessible and can be integrated into a coherent picture, when required, to assess and cope with a situation"[CITATION Sarter \l 3082], which imply in relevant critical success factors as:

The following factors are identified as critical to the success of this scenario:

- F1 - Scalability and reduced cost in slice setup and management that will allow infrastructure providers to extend their client-base and the MTCs to offer quality services to more citizens (Cost Reduction/Scalability).
- F2 - Elastic slice operation providing short-time response for the deployment on the field.
- F3 - Easy and low-cost network elements configuration.
  F4 - Fast service deployment will be reached through the use of Services templates, facilitating service deployment on the allocated slice in an automatic manner.

These critical success factors are mapped to the following KPIs:

- K1 - Accomplishment of operations such as creation, enlargement, shrink of slice. Metrics - Provisioning and decommission times.
- K2 - Fast provision of the slice to the tenants considering the service's requirements. Metrics - Service provisioning time.

- K3 - Provision of monitored data to the operator and the tenant. Metrics - Monitoring-data availability.
- K4 - Content delivery performance to highlight the end-user satisfaction. Metrics – end-user QoE.
- K5 - Network disruption to evaluate the efficient offline slice execution. Metrics – service disruption index.

7.4 Mapping to NECOS key characteristics

The particular service needs for the scenario are the following:

- Secure content availability
- Support for alleviating bulk data transfers of public safety data to end-users
- Low latency content-delivery to end-users (civilian population) by low-cost dynamic resource allocation according to end-user demands.

Thus, NECOS service provisioning approach (Objective 2) is suitable for SCD scenario, since it foresees the integration of resources within the collection of independent slices. A goal of the LSDC approach is to reduce the complexity and timescale for service provisioning and deployment in federated DCs, thus reducing the OPEX for the infrastructure owner.

Moreover, NECOS also envisions service and resource orchestration and management methods for the LSDC infrastructure resources that are located within and at the edge of the SCD network. This service orchestration and management approach, which comprises NECOS Objective 3, includes the automatic re-allocation of resources and services across distributed and geographically separated computing, data storage, and network infrastructures within separate slices to support population requests for SCD application data.

Finally, NECOS objectives 1 (develop and build a Lightweight Slice Defined Cloud (LSDC) platform enabling computing, network, and storage elements in the cloud through the Slice as a Service across federated clouds) and 4 (a pilot-based impact demonstration) will be achieved as well to demonstrate the provision of location-aware public safety content delivery to population.

In order to accomplish the aforementioned objectives, we consider the following key NECOS characteristics for the scenario:

- C1: Slice as a Service, mapping enhancing component (re) configuration to support E2E SCD applications;
- C2: Lightweight Edge Cloud, with small footprint components, deployable on both core and edge small servers and cloud systems, to support a user environment characterized by mobility and disruption
- C3: Software Management under a per-slice approach, to support features like user demand elasticity (i.e., Load Balancing through multiple VMs) and monitoring.

7.5 Functional and Non-Functional Requirements

The following functional requirements were identified:

- RF.SmartCity.1: On-demand slice provisioning. The system must allow the operator to create or adapt the slices on demand.
- RF.SmartCity.2: Slice modification, w/ capacity redefinition and function insertion/removal.
- RF.SmartCity.3: VM migration and isolation.
- RF.SmartCity.4: Offline slice execution when it loses the connection with the core network.
  RF.SmartCity.5: Accountability. The system must offer monitoring and accounting in a per slice basis.

The following non-functional requirements were identified:

- RN.SmartCity.1: Isolation of slice resources. Tenants must be protected from each other, i.e., it must be avoided any information leakage (e.g., monitoring measurements) among the slices.
- RN.SmartCity.2: Fault detection. Any service should operate continuously for a long time, so failures in the slice level should be automatically handled without impacting the service.
- RN.SmartCity.3: Location-aware resources allocation. To allow working with a specific data center distributed over the city
- RF.SmartCity.4: Service Level Agreement. The system must assure all the performance metrics (bandwidth, latency, CPU) negotiated for each slice.