5GCITY

Grant Agreement No.761508 5GCity/H2020-ICT-2016-2017/H2020-ICT-2016-2

D2.4: 5GCity Final Architecture and Interfaces

Dissemination Level		
☒	PU:	Public
☐	PP:	Restricted to other programme participants (including the Commission Services)
☐	RE:	Restricted to a group specified by the consortium (including the Commission
☐	CO:	Confidential, only for members of the consortium (including the Commission Services)

Grant Agreement no: **761508**	Project Acronym: **5GCity**	Project title: **5GCity**

Lead Beneficiary: **I2CAT**	Document version: **V0.1**

WP2 - 5GCity Architecture, Requirements and Use Cases

D2.4: 5GCity Final Architecture and Interfaces

Start date of the project: 01/06/2017 (duration 30 months)	Contractual delivery date: 30.06.2019	Actual delivery date: 24.07.2019

Editor name: Shuaib Siddiqui (i2CAT)

List of Contributors

Participant	Short Name	Contributor
Fundacio I2CAT	I2CAT	Shuaib Siddiqui, August Betzler, Apostolos Papageorgiou, Sergi Figuerola
Italtel	ITL	Antonino Albanese, Viscardo Costa, Valerio Vecchio Verderame
INCITES	INC	Ioannis Neokosmidis, Theodoros Rokkas
Retevision (Cellnex Telecom)	RTV	David Pujal
Nextworks s.r.l.	NXW	Gino Carrozzo, Leonardo Agueci, Elian Kraja
WindTre	WindTre	Maria Rita Spada
Ubiwhere	UBI	Ricardo Preto
Virtual Open Systems	VOSYS	Michele Paolino, Teodora Sechkova
ADLINK	ADLINK	Gabriele Baldoni
ACCELLERAN	XLRN	Antonio Garcia

List of Reviewers

Participant	Short Name	Contributor
Italtel	ITL	Antonino Albanese, Viscardo Costa
Nextworks s.r.l.	NXW	Gino Carrozzo
WindTre	WindTre	Maria Rita Spada

Change History

Version	Date	Partners	Description/Comments
v0.1	18-04-2019	I2CAT	Table of Content
V0.2	09-05-2019	VOSYS	Contributions by VOSYS
V0.3	10-05-2019	ADLINK	Contributions by ADLINK
V0.4	30-05-2019	ITL	Contributions by ITL
V0.5	14-05-2019	INC, W3, XLRN, NXW, UBI, i2CAT	Contributions by various partners
V0.6	30-06-2019	I2CAT	First integrated version
V0.7	10-07-2019	I2CAT	Completion of workflows
V0.8	16-07-2019	I2CAT	Consolidation of section 4 and internal review
V1.0	24-07-2019	I2CAT	Final version for submission

DISCLAIMER OF WARRANTIES

This document has been prepared by 5GCity project partners as an account of work carried out within the framework of the contract no 761508.

Neither Project Coordinator, nor any signatory party of 5GCity Project Consortium Agreement, nor any person acting on behalf of any of them:

- makes any warranty or representation whatsoever, express or implied,
  - with respect to the use of any information, apparatus, method, process, or similar item disclosed in this document, including merchantability and fitness for a particular purpose, or
  - that such use does not infringe on or interfere with privately owned rights, including any party's intellectual property, or
- that this document is suitable to any particular user's circumstance; or
- assumes responsibility for any damages or other liability whatsoever (including any consequential damages, even if Project Coordinator or any representative of a signatory party of the 5GCity Project Consortium Agreement, has been advised of the possibility of such damages) resulting from your selection or use of this document or any information, apparatus, method, process, or similar item disclosed in this document.

5GCity has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 761508. The content of this deliverable does not reflect the official opinion of the European Union. Responsibility for the information and views expressed in the deliverable lies entirely with the author(s).

Table of Contents

Figures **7**

Tables **8**

Executive Summary **9**

1. Overview **10**

1.1. 5GCity Architectural principles *10*

1.2. 5GCity ecosystem *12*

1.2.1. Incentives to adopt 5GCity solution 13

2. Service Layer **16**

2.1. SDK *16*

2.1.1. Architecture Design 17

2.1.2. SDK Service updated info model 19

2.2. 5G Service and Application Catalogue *23*

2.3. OSS/BSS *24*

2.3.1. Product/Service Catalogue Management 24

2.3.2. Customer Contact Management, Retention & Loyalty 25

2.3.3. Corporate Sales Management 25

2.3.4. Contract Management Functionality 26

2.3.5. Customer Order Management 26

2.3.6. Service Order Management 26

3. Orchestration & Control Layer **27**

3.1. Dashboard *27*

3.2. 5GCity Orchestrator *28*

3.2.2. Resource Placement 29

3.2.3. SLA Manager 29

3.2.4. Slice Manager 30

3.2.5. Infrastructure Abstraction 30

3.2.6. WAN Resource Manager 31

3.3. Virtualized Infrastructure Manager (VIM) *31*

3.3.1. Core 32

3.3.2. Edge 33

3.3.3. Extended Edge 34

3.4. SDN controller and agents *34*

3.5. Monitoring Framework *36*

3.5.1. Monitoring Module 36

3.5.2. Monitoring system 37

4. Infrastructure Layer **40**

4.1. Core NFVI (data center) *40*

4.2. Backhaul Network *40*

4.3. Edge NFVI - Radio *42*

4.3.1. 3GPP RAN Network Slicing and Virtualized Edge Architecture 42

4.3.2. Mobility Management at the Edge 43

4.3.3. Management and Orchestration of RAN VNFs 44

4.4. Front-haul Network *45*
4.5. Extended Edge NFVI *46*
4.5.1. MEC in the Extended Edge NFVI 46
4.6. Radio Element (for Small Cells and/or Wi-Fi) *46*
4.6.1. Small Cell Radio Element 46
4.6.2. Wi-Fi Radio Element 47
5. Interfaces and Workflow 49
5.1. Interfaces *49*
5.1.1. SDK Interfaces 49
5.1.2. 5GCity Platform interface updates 54
5.2. Workflows *60*
5.2.1. Slice User Registration 60
5.2.1. Service creation 61
5.2.2. Slice creation 62
5.2.3. Service Provisioning 63
5.2.4. SLA-based Monitoring 64
6. Conclusion 66
7. References 67
Abbreviations and Definitions 68
7.1. Abbreviations *68*

Figures

Figure 1 - 5GCity Three-Tier Topological Architecture 10
Figure 2 - 5GCity high-level architecture functional design 11
Figure 3 - Three-Tier 5GCity 'topological' architecture along with SW mapping/deployment 11
Figure 4 - 5GCity reference value network 14
Figure 5 - 5GCity SDK Toolkit high-level architecture 17
Figure 6 – Monitoring Parameter information model 19
Figure 7 – Service Action information model 20
Figure 8 – Service Action Rule information model 20
Figure 9 - General schema NFV architecture (source ETSI GS NFV 002 V1.2.1, 2014-12) 24
Figure 10 – 5GCity Dashboard reference component design 27
Figure 11 - 5GCity orchestrator high-level architecture 28
Figure 12 - OpenStack architecture 32
Figure 13 - EdgeVIM architecture 33
Figure 14 - High-level architecture of Eclipse fog05 34
Figure 15 - Internal structure of the RAN SDN controller components 35
Figure 17 - 5GCity Platform design and interface to Monitoring framework 37
Figure 18 - Deployment scenario selected in 5GCity 38
Figure 19 - Backhaul network 41
Figure 20 - 5GCity RAN Network Architecture Overview 42
Figure 21 - Intra vEPC Mobility 43
Figure 22 - Inter vEPC Mobility 44
Figure 23 - RAN SDN Control Interface Model 45
Figure 24 - Accelleran E1000 Series Local Area Small Cell 47
Figure 25 - Accelleran E1000 Series inside 5GCity Showroom Lamppost 47
Figure 26 – SWAGGER documentation for service descriptor resource 50
Figure 27 - SWAGGER documentation for service resource 51
Figure 28 - SWAGGER documentation for function resource 52
Figure 29 - Snapshot of Slice Manager REST documentation (some high-level resources) 59
Figure 30 - Snapshot of Slice Manager REST documentation (managing a compute resource) 60
Figure 31 -Slice User Registration at Neutral Host (5GCity platform operator) 61
Figure 32 -Service Design workflow 62
Figure 33 - Slice creation workflow with the 5GCity neutral host platform 63
Figure 34 - Service instantiation upon a previously commissioned slice 64
Figure 35 - Handling MEC applications as part of the service provisioning workflow 64
Figure 36. SLA-based monitoring workflow 65

Tables

Table 1 - Players of 5GCITY ecosystem 12
Table 2 - Players of 5GCity ecosystem w.r.t. incentives 13
Table 3 - Roles and permissions of the 5GCity Service & Application layer actors 16
Table 4 - SDK Business logic 21
Table 5 - Information model of the SDK service 21
Table 6 - Information model of the SDK service function 22
Table 7 - Interface description for the service descriptor resource 49
Table 8 - Interface description for the service resource 50
Table 9 - Interface description for the function resource 51
Table 10 - Interface to the Public Catalogue for Network Service Descriptors 53
Table 11 - Interface to the Public Catalogue for VNF Packages 53
Table 12 - Interface for managing compute resources 54
Table 13 - Interface for managing physical network resources 54
Table 14 - Interface for managing RAN infrastructure resources 55
Table 15 - Interface for managing compute chunk resources 55
Table 16 - Interface for managing physical network chunk resources 56
Table 17 - Interface for managing access network chunk resources 56
Table 18 - Interface for managing slices 57
Table 19 - Interface for managing users 57
Table 20 - Interface for managing network services 57
Table 21 - Interface for managing network services 58

Executive Summary

This document details the updated and final version of the 5GCity Architecture and Interfaces after the 3rd iteration. The initial 5GCity architecture and interfaces were specified in deliverable D2.2 and D2.3.

The main contribution of D2.4 includes final description of:

- 5GCity Architecture
- Service Layer functions
- Orchestration & Control Layer functions
- Infrastructure Layer functions

This deliverable provides details of the design, at system-level, of 5GCity virtualization, service and orchestration & control platform implemented in WP3 and WP4, respectively. Furthermore, this report illustrates the final interfaces and workflows of 5GCity platforms.

The document is organized in a way to be consistent with the previous two versions and, more importantly, also self-contained.

1. Overview

1.1. 5GCity Architectural principles

5G domain advanced technologies allow to unlock high quality services exploiting dynamic efficient resource allocation mechanisms. It is important to implement solutions able to take advantage of these technologies to realize services and applications with ultra-low latency, very-high bandwidth and flexible dynamic configuration offered by the new 5G networks.

In this context, the main goal of 5GCity project is to design solution that can turn a city into a "digital fabric" capable to exploit distributed and multi-tenant edge infrastructure, integrate 5G services administered by a neutral host who use orchestration and service programming tools to manage the underline resources. 5GCity can ultimately result into an enabler of a Smart City leveraging on 5G technologies to offer its digital services to citizens and vertical industries.

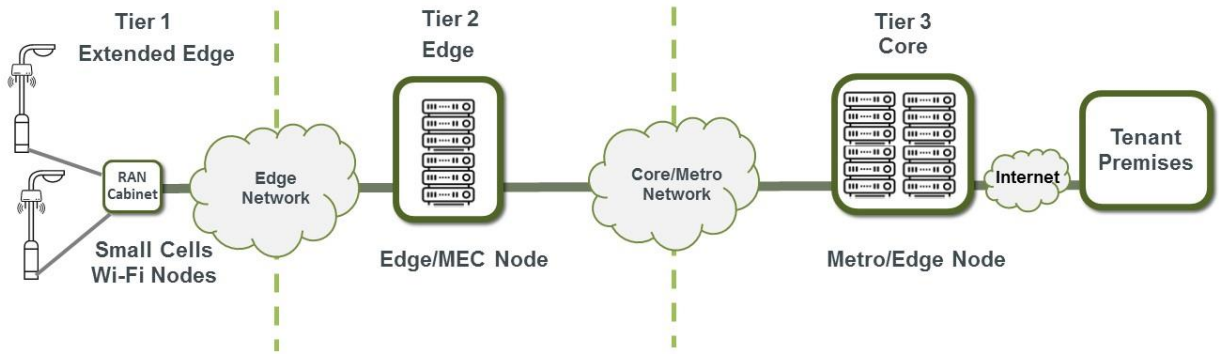

Figure 1 - 5GCity Three-Tier Topological Architecture

The approach followed to develop the 5GCity architecture, able to get the described goal, is to integrate Cloud and Edge computing paradigms in a single end-to-end infrastructure.

The 5GCity physical resources result to be deployed and identified according to the architectural schema depicted in Figure 1. Therefore, from a physical perspective, it has been logical to design a reference three-tier architecture blueprint, which identifies three different geographical areas (or tiers) of the actual infrastructures for digital services in the cities:

- *Tier 1*: edge area where wireless devices (Small Cells or Wi-Fi powered) are deployed;
- *Tier 2*: edge area corresponding to street cabinets (where limited computing resources are available due to physical constraints);
- *Tier 3*: centralized area (which typically corresponds to a Data Center, where massive computing resource are deployed);

From a functional point of view and to better fit with the 5G requirements and with the proposed physical concept, the 5GCity architecture has been designed to split vertically across three layers (see Figure 2):

- Service/Application Layer;
- Orchestration & Control layer;
- Infrastructure layer.

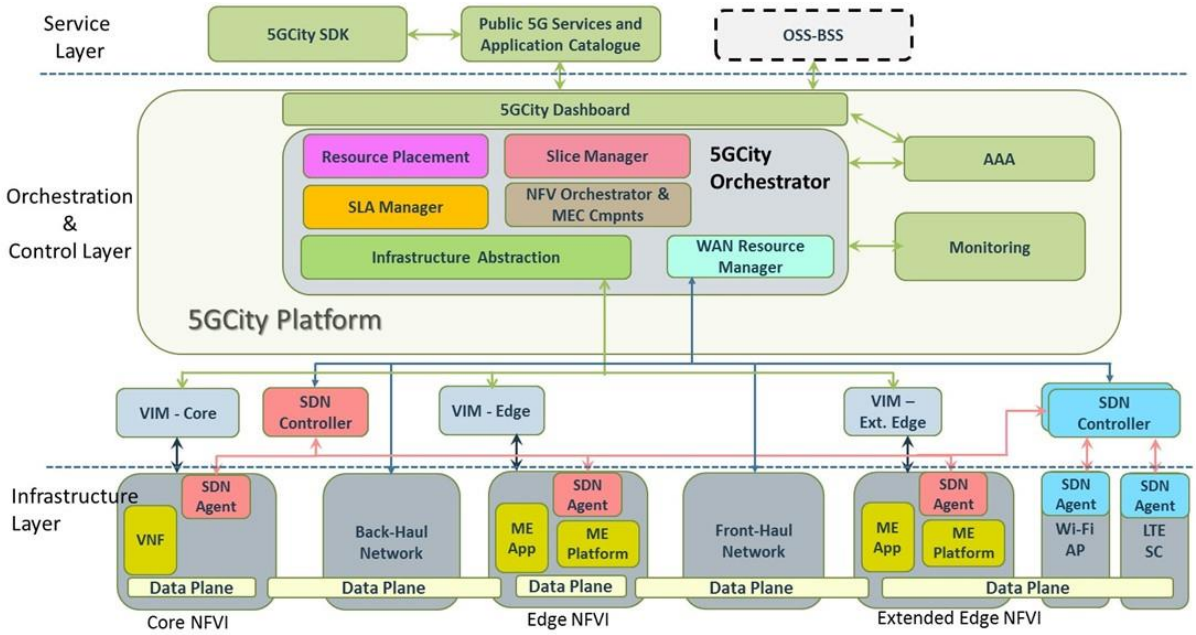

Figure 2 - 5GCity high-level architecture functional design

It contains functional blocks to combine distributed cloud technologies and edge network virtualization making full use of the new city edge infrastructure deployments such as street cabinets, city-owned edge servers, wireless access points and small cells.

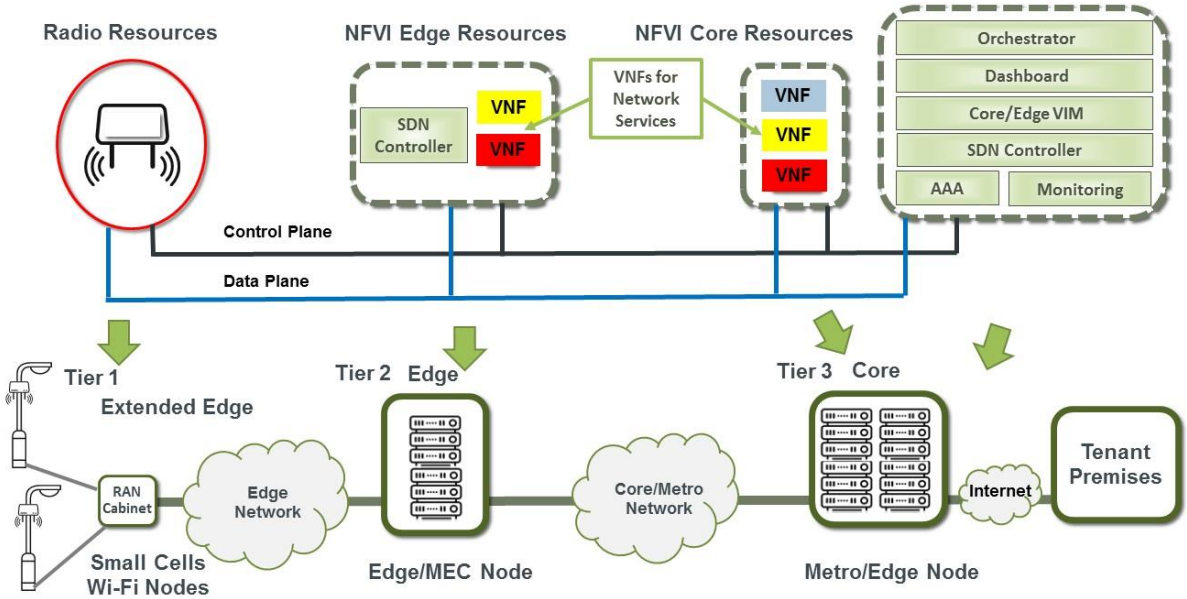

Figure 3 - Three-Tier 5GCity 'topological' architecture along with SW mapping/deployment

To give a more clear view of the idea behind the 5GCity architecture, Figure 3 shows the software mapping/deployment with respect to the city infrastructure. Typically, radio resources are deployed at the far-edge/lamppost, Virtual Network Functions (VNFs) can be deployed at edge/MEC node or in Metro/Edge node depending on the specific use case deployment scenarios and requirements (e.g. local processing or reduced latency) to be fulfilled; orchestration platform components and service design and management portals are deployed at core.

In the following paragraphs of this document we provide a detailed description of the 5GCity architecture, its functional components and interfaces with reference to the core service workflows, starting from a description of the 5GCity ecosystem.

1.2. 5GCity ecosystem

The stakeholders involved in the 5GCity ecosystem are key to motivate the need for designed a new innovative orchestration platform as described in this deliverable. Taking into account the initial 5GCity architecture and Use Cases described in D2.1 [1], the Neutral Host represents the main player in 5GCity ecosystem because it is positioned as the mainly adopter of the 5GCity solutions and tools.

The description of the players along with some example entities is presented in Table 1 and illustrated in Figure 4 - 5GCity reference value networkFigure 4.

Table 1 - Players of 5GCITY ecosystem

Player	Description – Product/service details	Example entities
Facility Manager	An entity that provides space (malls, stadiums, streetlights, etc.) to operators, verticals and/or NH for equipment installation	Public (e.g. Municipality), or private (venue owner, transportation players, etc.)
Neutral Host (NH)	NH can provide wholesale access to **passive** network infrastructure: energy, safety, access, air conditioning, etc.	Facility manager, or dedicated company (e.g. Cellnex)
	NH can provide wholesale access to **active** network infrastructure: IT and network equipment	Cellnex
Connectivity provider	Connectivity (fiber, transport, wireless backhaul) from the sites where the equipment is installed to the external network (typically to the MNO networks) and among them	Cellnex
Mobile Network Operator (MNO)	MNO provides wireless access to end users in wide areas.	WindTre
Spectrum owner	Possess and rent spectrum licenses to interested parties (MNOs, verticals or NH). Although MNOs must currently possess spectrum licenses, this may not be the case in the future.	WindTre
End-user	Residential end-user: the ultimate consumer of a service or the user of a product. The end-user can be a home user or the owners or users of IoT devices. End-users can also act as content providers.	Home network user or mobile end user
	Vertical industries (business or enterprise user): Industry or group of enterprises in which similar products or services are developed and marketed. These include: Automotive companies, e-Health companies, Manufacturing companies, Energy companies, Media & Entertainment companies	BMW, SES, RAI, BETEVE, BBC, Hutchison Whampoa Europe, UnitedHealth Group
Service Provider	Third parties that provide products to end users (EUs). These can be a network function or a bundle of network functions from one or from multiple Function Providers (FPs) or just an end-to-end network service, e-services (storage, processing e.t.c.) and value-added services (VAS)	Amazon Web Services, Oracle Cloud, Mahindra Comviva Facebook, Google

HW manufacturer and equipment vendor	An entity manufacturing and providing IT and network equipment such as servers, switches, routers, EPC, etc.	Huawei, Cisco, Ericsson, HP, Dell
SW / Functions Developers	They supply virtual network appliances, gateways, proxies, firewalls, transcoders, etc., eliminating the need for the customer to acquire install and maintain specialized hardware. They are also developing several types of software programs	Accelleran, NEC, Nextworks, Ubiwhere, ADLink, VOSYS, i2CAT, ITL
Content / License provider	An entity that buys the content (e.g., text, graphics, news, audio, video, etc.) developed by content producers and supplies (sells) content for use on a website or other platforms. It manages access to a structured set of data, encapsulates the data, and provides mechanisms for further use. It is the standard interface that connects data in one process with code running in another process.	The New York Times, Thomson Reuters Corporation, Netflix, Dailymotion Broadcasters

1.2.1. Incentives to adopt 5GCity solution

Even though the value generation will be specific to each player in the business model, the adoption of the 5GCity solution, offers the following strong incentives:

Table 2 - Players of 5GCity ecosystem w.r.t. incentives

Player category	Incentives

Operators	• Cost reduction
	  ○ CAPEX: no need for investments;
	  ○ OPEX: several functions like maintenance, operation etc. are on NH side.
	• Complexity reduction (no need for maintenance etc.);
	• Dynamic billing: Different costs models to be applied like pay as you grow, as you use;
	• Facilitate the rights of way;
	• Ability to focus more on the customers' needs and services;
	• Less risk and shorter process in introducing new services.

Vertical Industries	• Guaranteed performance according to the SLA; • Network characteristics according to their needs; • No need to install their own networks; • Ease of private networks deployment; • Advanced functionalities; • Ability to provide advanced services with strict requirements due to the guaranteed performance.
Cities / Councils	• Exploit their infrastructure; • Provide specialized services to their citizens; • New revenue sources;

	• Environmental benefits: reduction of small cells to be installed / Reduce the antennas impact; • Facilitate the deployment of 5G across the whole city.
Service Providers	• Ease of service creation due to the ability to combine pre-developed services and functions provided by the services dashboard.
Functions and SW developers	• Ease of functions and services development due to the SDK tool provided by the services dashboard.
Infrastructure Providers and Facility Managers	• Growth opportunities due to the increased needs for space, IT, fiber and other resources. • Increased participation in the value chain of the telecommunication services (from passive to active assets)

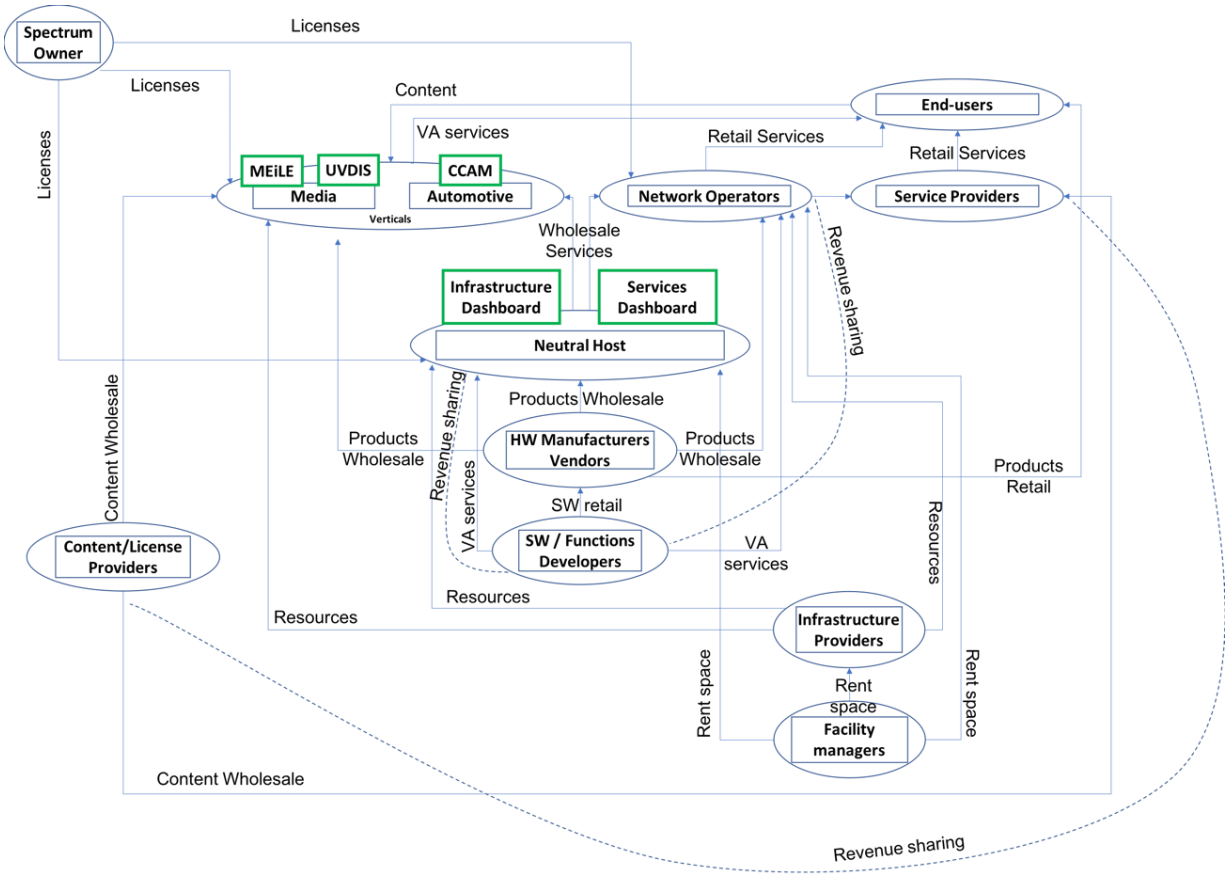

Figure 4 - 5GCity reference value network

The next step is to define the relationships between the players of 5GCity ecosystem based initially on our Use Cases and to identify new roles and actors/players.

The related business model involves many different players and a complex value network (reference network). In Figure 4, the 5GCity reference value network is described with all the participating players, relationship interfaces and revenue streams.

Definitions:

- The direction of the arrows in the model represents the direction of service flow;

- Revenue flow is considered to be in the opposite direction. In some cases, revenue sharing exists between two players, which is bidirectional.

Based on the general reference model presented in Figure 4, different focused Business models have been designed (and crafted) per Use Case to identify the specific values and strengths of implementing e.g.,

- Neutral Host
- UHD Video Distribution Immersive Services
- Cooperative, Connected and Automated Mobility (CCAM)
- Video Acquisition and Production – Community media engagement in live events

Insights on these business models and techno-economic analyses for some of them are planned to be delivered in deliverable D2.5 due by the end of the project.

2. Service Layer

In this section we describe the final design of the components of the 5GCity Service Layer. As described in Deliverable D2.2 [2], the 5GCity Service & Application layer consists of functionalities needed by infrastructure operators, their customers, subcontractors and any third-party actor to design, manage and request deployment of network slices and services in the 5GCity infrastructure, depending on the various roles covered.

The 5GCity Service layer comprises the SDK, an application catalogue and OSS/BSS systems.

2.1. SDK

An initial description of the architecture of the 5GCity SDK functional component has been provided in Deliverable D2.3 [3]. As initially designed, the 5GCity SDK toolkit is a self-contained, stand-alone software platform, which provides the service designers (application developers and verticals) with a set of tools to define and compose service functions and services in a simplified and guided mode, thus abstracting the underlying actual NFV MANO infrastructure complexity.

The SDK toolkit has been designed having in mind two main objectives:

- Providing application developers with tools for designing Service templates and Service Functions in abstract mode related to atomic functions realized by their services. Atomic functions constitute the building blocks for more complex, end-to-end, composite services. In 5GCity, atomic functions correspond to VNF packages or Network Services as defined in raw NFV contexts, but they result simpler to define and package with respect to the NFV counterparts; services correspond to [nested] Network Services.
- Providing vertical users (belonging to vertical sectors like Media, Industry, Automotive) with a design platform through which to compose complex Services templates with customization of the service functions defined by the developers. The modelling is based on an abstract and simplified view of the Network Services and VNF packages, without requiring any detailed knowledge of infrastructure level requirements.

The 5GCity SDK toolkit can facilitate the service planning and design phase for the two main actors in the 5GCity Service & Application layer, i.e. the Designer and the Composer. The two actors' roles, together with the Admin role, and their permissions are summarised in Table 3:

Table 3 - Roles and permissions of the 5GCity Service & Application layer actors

User/Actor	Role	SDK Service	SDK Function
Admin	Admin	R/W	R/W
Vertical	Composer	R/W	R
Developer	Designer	R/W	R/W

Given that the admin user and role exist with full read/write access to SDK operations (as in 5GCity platform), a Vertical user is assigned the role of Composer and he can re-use service functions edited by Developers by wiring them together to realize end-to-end network services tailored to the needs of the specific application

business logic. Contrarily, a developer is assigned the role of Designer and he can build SDK Functions pertaining to the different areas (media, automotive, smart city services) from scratch.

2.1.1. Architecture Design

The 5GCity SDK architecture initially produced for deliverable D2.3 [3] has been updated based on refinements and optimization deriving from software implementation. The new and final high-level functional design of 5GCity SDK is depicted in Figure 5.

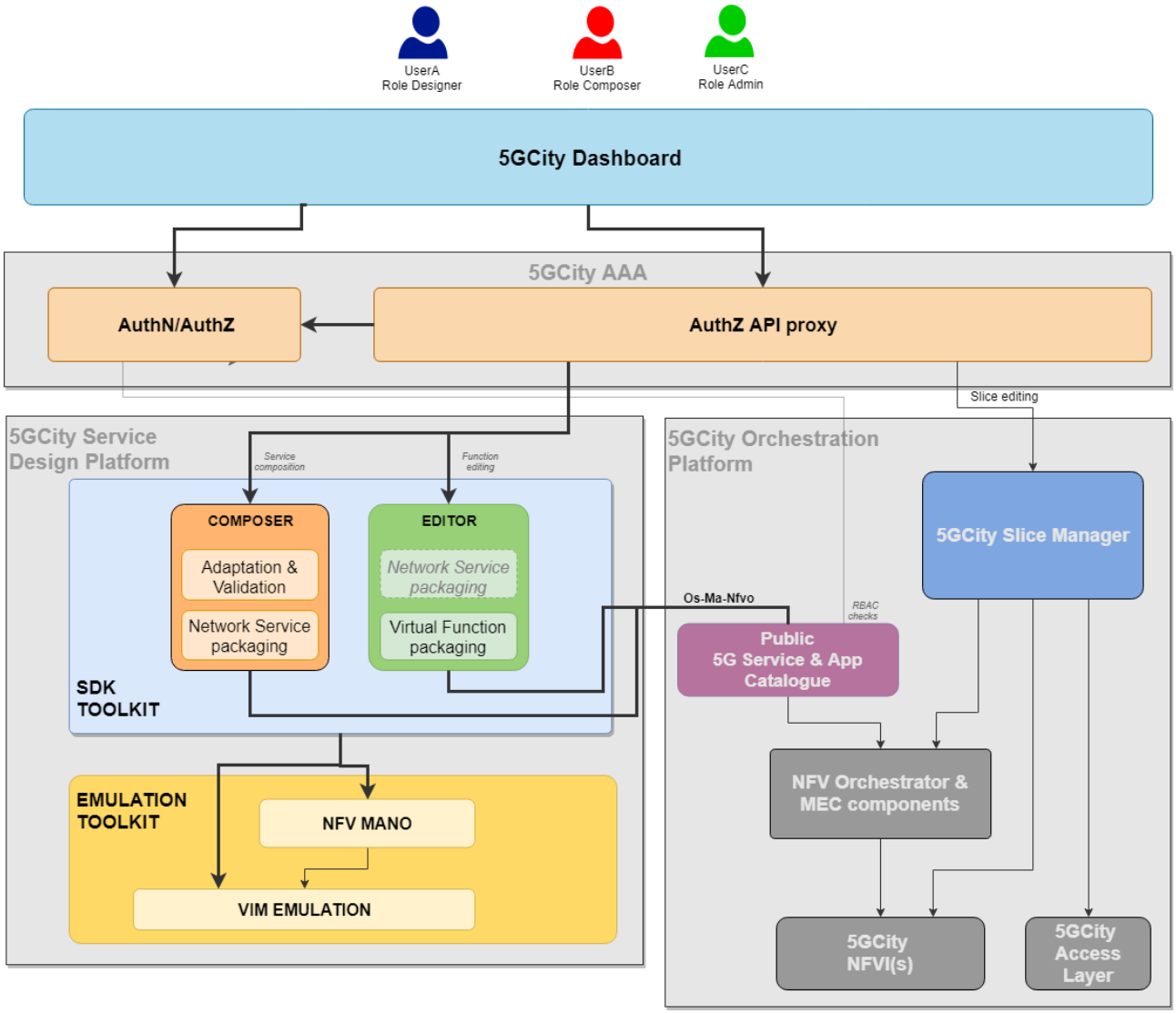

Figure 5 - 5GCity SDK Toolkit high-level architecture

The components of the 5GCity SDK and their main functionalities remain unchanged:

- **5GCity Dashboard** represents the entry point to the SDK platform, allowing different categories of users to define new 5G services based on different levels of abstraction.
- **SDK Composer** allows the vertical user to compose services and perform CRUD operations on them. The vertical user will have the possibility to publish the created service into a public 5G Service and Application Catalogue, after an internal process of validation and translation into the ETSI TOSCA model. The 5G Service and Application Catalogue are typically deployed as part of the MANO infrastructure, and will be used to onboard ETSI compliant Network Services and VNF package

descriptors into the underlying NFV Orchestrator. During the process of service design, the vertical user is guided by the composer GUI interface to select the building blocks of the service, interconnect them based on the required network connectivity and define a list of high-level parameters associated to each function and service link. The resulting service description will be then translated by the logic of the internal Adaptor module into a Network Service Descriptor compliant with the ETSI NFV standards, specifying low-level details like the size of the required Virtual Machines, the bandwidth capabilities of the Virtual Links, etc.

- **SDK Editor** allows the DevOps/admin user to define new functions that are used by the vertical user to create new services or update already created ones. A SDK function is an abstraction of NFV parameters better known from a DevOps user when he/she intends to create a vertical service. For each function, a list of translation rules and configuration parameters are declared to map the high-level function parameters into the low-level parameters for Network Services and VNFs, compliant with the ETSI NFV model. These high-level service parameters are defined by the vertical user during the service creation phase through the Composer GUI.

Compared to the previous versions of SDK design reported in deliverables D2.2 [2] and D2.3 [3], we have made less strict the architectural split between Composer and Editor functionalities: in fact, the SDK toolkit exposes a unified Northbound programming Interface through which it is possible to handle different resources (services, service descriptors and functions), and the actual functional differentiation between Composer and Editor reduces to the type of actor/roles authorized to operate on specific resource endpoints.

Also, the Policy Engine initially presented as part of the SDK architecture, has been removed in this final design since it resulted in a duplicated functionality with respect to what is just implemented by the cross-layer 5GCity AAA component. In fact, depending on the authenticated user, the system provides access to Composer and/or Editor functionalities and exposes the list of functions and services for which access is authorised.

The main logic to translate the simplified information model in SDK, which is based on vertical-driven concepts like functions and services, into a full ETSI NFV model made of VNF packages and Network Services descriptors is embedded in a translation function used by the Composer and the Editor for the various types of resources to be handled (i.e. Services and Service Descriptors for NSD in Composer; Functions for VNF packages in Editor).

The full workflow that describes their service design and composition within the 5GCity SDK can be described as follows:

1. A developer user, via Editor endpoints, creates a VNF package from scratch, and after saving it into the internal 5GCity SDK repository, then associates it to an elementary “service function” with a set of configurable, high-level parameters (which will be set by the user as part of the service customization). Moreover, the developer also specifies a list of translation rules, in a predefined format, which states how the high-level service parameters are mapped into specific information elements of the VNF Package and Network Service Descriptor information models (e.g. deployment flavours, instantiation levels, etc.).
2. Service functions created at the previous step are exposed with their high-level configurable parameters to the vertical user when he/she composes the desired end-to-end service.
3. During the service design process, a vertical user selects the target service functions and provides specific values for the defined configuration parameters, according to the business logic required to implement a specific service (e.g. combination of deployment flavour and instantiation level depending on number of users to be served by the function/service).
4. Based on the specified parameters and the translation rules associated to each function, a service template is translated into a Network Service Descriptor defined according to the standard ETSI NFV information model (ETSI NFV SOL 001 [7], SOL 004 [6], SOL 005 [8], specifications). The resulting descriptor can thus be processed by NFV catalogues and orchestrator platforms.

The 5GCity SDK interacts with other components of the 5GCity platform. In particular, it interacts with a 5G Services and Apps Catalogue to onboard VNF Packages and Network Service Descriptors during the service design phase. The interface with the 5G Services and Apps Catalogue is based on a REST API compliant with the ETSI GS NFV-SOL 005 specification [8], where the SDK acts as client and the 5G Services and Apps catalogue acts as server. The 5G Services and Apps Catalogue is a third-party open source software developed by Nextworks and publicly available in [4]. The component documentation is available in [5]. It implements the functionalities of an NFV catalogue able to on-board VNF Packages and Network Service Descriptors in TOSCA CSAR format, as specified in ETSI GS NFV-SOL 004 [6] (package format) and ETSI GS NFV-SOL 001 [7] (NSD and VNFD format). Internally, the catalogue translates the standard descriptors received in input into NFVO-specific descriptors (e.g. based on the information model adopted for descriptors in Open Source MANO) and publishes them into the target NFVO(s).
The 5GCity SDK toolkit interacts also with the 5GCity AAA (which is the Policy Engine component introduced earlier) to authenticate and authorize the users. This component also acts as a proxy providing, according to specific policies, only the functions and services that are visible for the given user.
As optional, the 5GCity SDK toolkit may also interact with an Emulation to run in emulated MANO environment exemplary Network Services designed through the 5GCity SDK Toolkit. This interaction could be used to functionally validate the services before their actual deployment in the operational NFV infrastructure.

2.1.2. SDK Service updated info model

The information model of the SDK toolkit has been designed as an abstract and vertical-driven version of the full ETSI NFV standard information model, in order to provide the vertical user with a simplified procedure for designing a network service template.

To clarify the concept, a service is composed by a list of components (a component may be a function or another service), connection points, links and of course several others parameters. Moreover, a service allows the user to specify a list of monitoring parameters, actions, actions rules and L3 connectivity rules useful to manage the service in a dynamic manner (e.g. scale in or scale out a specific function) depending on the specified aspects (number of connections, usage of the CPU etc.).

Although most of these parameters have been described in previous deliverables from WP2 and WP4, new additional ones have been introduced in this final release. In particular, we added:

Monitoring Parameters to model monitoring of network services and resources. Their information model is depicted in Figure 6.

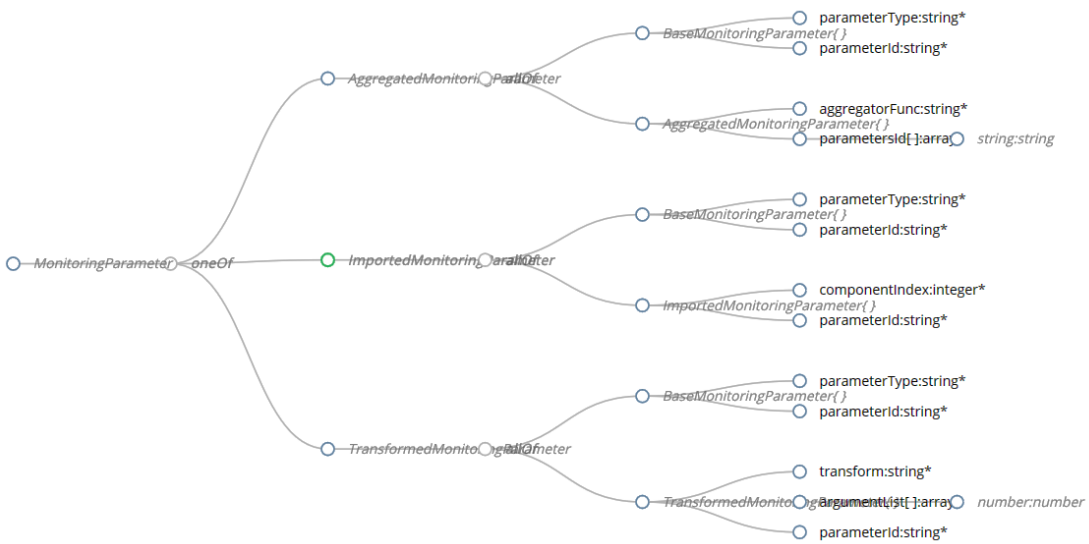

Figure 6 – Monitoring Parameter information model

- **Actions and Action Rules,** which define the actions to be performed when a certain event occurs. Their information model is depicted in Figure 7 and Figure 8:

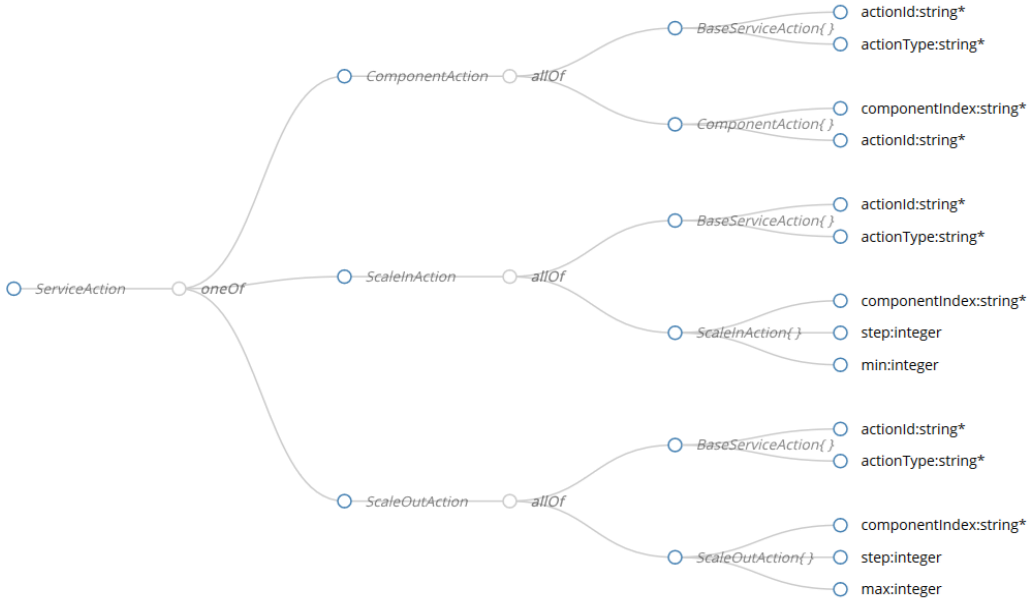

Figure 7 – Service Action information model

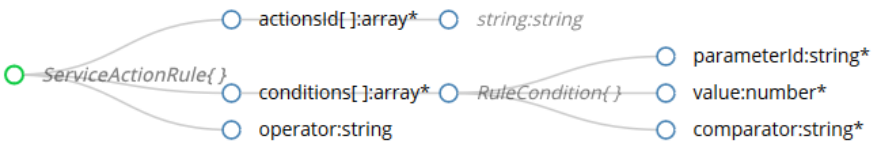

Figure 8 – Service Action Rule information model

- **OwnerId, GroupId, Visibility** to identify respectively the owner of the resource, one of the membership group of the owner and the scope the resource (it may be PUBLIC or PRIVATE). The access to the resources is managed using these parameters.
- **AccessLevel** to implement the SDK business logic. It allows creation of new SDK Services based on visibility level through which it is possible to filter access to specific functions and services based on the visibility level granted to a user/group.

Table 4 - SDK Business logic

Access level	Symbolic Name	Description
0	Platinum	These users can see all the functions and services tagged with visibility level ≥ 0
1	Gold	These users can see all the functions and services tagged with visibility level ≥ 1
2	Silver	These users can see all the functions and services tagged with visibility level ≥ 2
3	Bronze	These users can see all the functions and services tagged with visibility level ≥ 3

The complete information model of the service is described in Table 5.

Table 5 - Information model of the SDK service

Parameter Name	Cardinality	Description
Id	1	SDK-internal unique identifier of the service.
Status	1	Identifies whether the service is published to 5G Apps and Services Catalogue or not.
Name	1	Human readable identifier.
Version	1	Version number.
Designer	1	Designer name of the service.
Parameters	0 .. N	List of parameters available for service customization.
Licence	1	Licence defined for the service.
Link	1 .. N	List of virtual links contained into the service.
Components	1 .. N	List of the functions (see information model in **Table 6**) or more elementary services that compose the given service. This element also includes translation rules to map the service parameter values into the corresponding parameter values of the specific component.
Metadata	0 .. N	Key-value pairs to model service-specific data.
L3Connectivity	0 .. N	L3 firewalling rules to be applied on the defined connection points.
MonitoringParameter	0 .. N	List of available monitoring parameters.
Action	0 .. N	Defines the action to be performed based on monitoring parameters.
ActionRule	0 .. N	Defines the rule to activate a specific action
ConnectionPoint	0 .. N	Defines the internal and external connection points. The external ones allow the interconnection of the given service to other services or external networks (e.g. used to interconnect the

		users). The internal connection points are used to connect the components of the service.
OwnerId	1	Identifies the owner of the service.
GroupId	1	Identifies the group owner of the service.
Visibility	1	Defines the scope of the service (PUBLIC or PRIVATE).
AccessLevel	1	Business logic parameter through which it is possible to filter access to specific functions and services based on the access level granted to a user/group.

The complete information model of the function is described in Table 6.

Table 6 - Information model of the SDK service function

Parameter Name	Cardinality	Description
Id	1	SDK-internal unique identifier of the function.
Status	1	Identifies whether the function is published to 5G Apps and Services Catalogue or not.
Epoch	1	Timestamp related to the last update.
Name	1	Human-readable identifier.
Description	1	Description of the function, its purpose and capabilities.
Vendor	1	Function provider.
Version	1	Version of the particular function.
Parameter	0 .. N	Configurable parameters for function customization.
VnfdId	1	Id of the VNFD implementing this function (information not exposed to the vertical user).
VnfdInfoId	1	Id of the VNF Package published to the Public Catalogue.
SoftwareImageData	1	Information related to the image (image name, vCPU, vRAM etc.).
MinInstanceCount	1	Minimum number of instances to be instantiated.
MaxInstanceCount	1	Maximum number of instances to be instantiated.
FlavourExpression	1	Expression determining the deployment flavour of the VNF implementing this function based on the values of the parameters (information not exposed to the vertical user).
InstantiationLevelExpression	1	Expression determining the instantiation level of the VNF implementing this function based on the values of the parameters (information not exposed to the vertical user).

Metadata	0 .. 1	A map of key/value, provider-defined parameters for additional configuration of the function.
ConnectionPoint	0 .. N	A description of the connection point(s) exposed by the function.
MonitoringParameter	0 .. N	A description of the runtime parameter(s) which should be monitored and notification threshold associated to them.
RequiredPort	0 .. N	Port forwarding rules to be applied on the defined connection points.
OwnerId	1	Identifies the owner of the function.
GroupId	1	Identifies the group owner of the function.
Visibility	1	Defines the scope of the function (PUBLIC or PRIVATE).
AccessLevel	1	Business logic parameter through which it is possible to filter access to specific functions and services based on the access level granted to a user/group.

2.2. 5G Service and Application Catalogue

As described in deliverable D2.2 [2], the 5G Service and Applications catalogue is a 3rd party software component developed by Nextworks [4], [5] outside the 5GCity project and integrated as-is in the 5GCity Service Layer to operate with SDK and the underlying 5GCity platform.

The 5G Service & App Catalogue allows:

- service customers to bring their own VNFs or vApps into the Service Provider‘s MANO
- composing services from multiple service providers, each with its own specialized offer of VNF(s) and NS(s)
- using a generalized standard format and contents for descriptors and translate it into the various descriptions needed by the different targets (e.g. different NFVO)
- adopting core standard descriptions for the VNF, NS, MEC App that can be instantiated based on IFA-11, SOL-004, etc.
- adding NFVI capabilities (e.g. placing of MEC platforms, HW accelerations, GPU availability and where, etc.)
- adding description of additional VNF/vApp characteristics not related to Lifecycle Management (e.g. App monitoring/configuration after instantiation)

The 5G Service & App Catalogue’s key features in use for 5GCity are:

- Unified and extendable format for descriptors
    - VNF packages (ETSI GS NFV IFA 011, ETSI GS SOL 004)
    - NSs (ETSI GS NFV IFA 014, ETSI GS SOL OO1 draft)
- MANO domain-specific translation from common to specific descriptors
    - Support to OSM rel. 3, rel. 4, rel. 5

- Extended Interfaces to MANO (NFVO, VIM)
    - Base Life Cycle Management (LCM) behaviours based on basic NFV interfaces and descriptors
    - Additional behaviours (e.g. load images, configure HW acceleration, other tuning on NFVI, etc.) based on additional interfaces and descriptors

2.3. OSS/BSS

2.3.1. Product/Service Catalogue Management

As initially specified in D2.2 [2] , the OSS/BSS service provides Operating and Business supports functions in order to manage 5G network infrastructure (OSS) as well products, customers and customer life cycle (Orders, Upgrades, Complains, Usage & Invoicing, Leave) of the services provided by 5GCity framework (BSS), see Figure 9.

The need for highly automated and standardized processes is particularly stressed in 5G environments due to frequency/network/IT resource sharing in Product/services deployment with the target of making customer experience, services and network resources part of one orchestration.

Therefore, the integration should be based on "Open API" provided by the 5GCity framework (via service orchestration, usage if needed and management alerts).

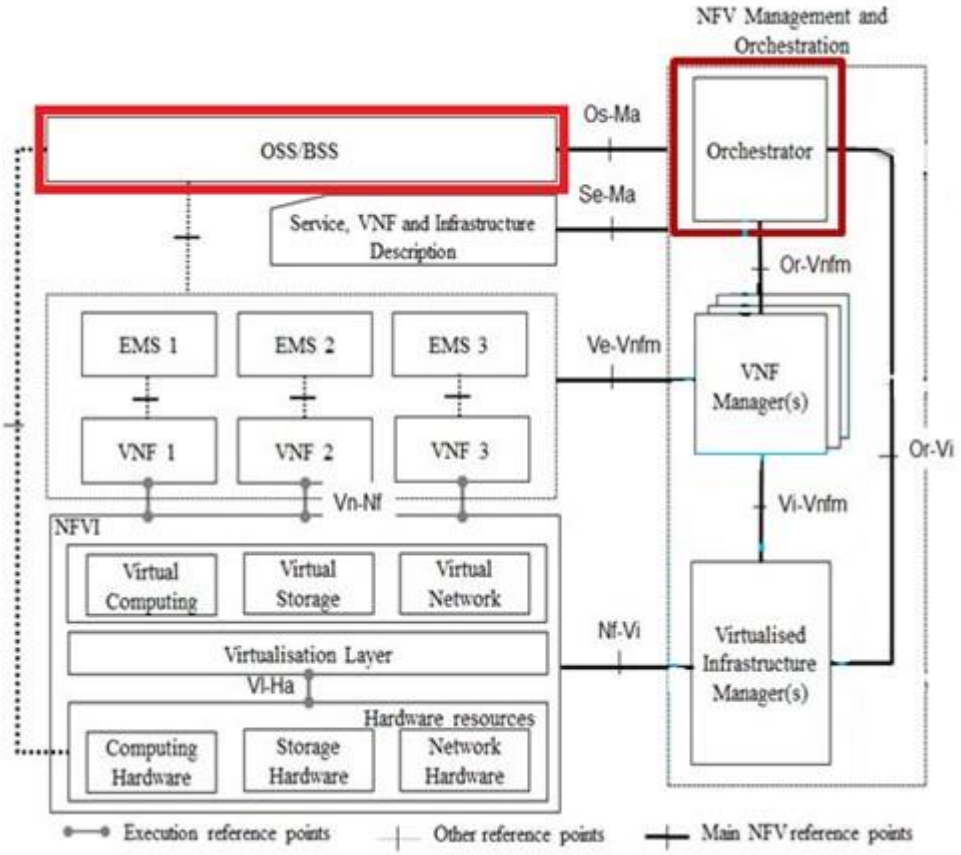

Figure 9 - General schema NFV architecture (source ETSI GS NFV 002 V1.2.1, 2014-12)

Additionally, as the Network Infrastructure moves towards virtualized infrastructures on COTS hardware, OSS and BSS systems evolve to a virtualization path based on IaaS, PaaS, or SaaS, by using a common commercial hardware and storage, and by providing separate virtualized infrastructure, virtualized platform, and

virtualized software to a set of tenants using the shared physical infrastructure. Specific application environments can be than made available for vertical implementations, including Artificial Intelligence and Machine Learning functionalities in order to provide enhanced analytics needed for a quick service time to market.

In the following, the principal features expected from these modules are introduced and discussed at high level because the 5GCity architecture includes concepts of OSS and BSS that it is worth to properly frame in the context of this document. However, 5GCity pilot infrastructure and validation activities do not plan to use OSS/BSS in the platform operations.

2.3.2. Customer Contact Management, Retention & Loyalty

As initially specified in D2.2 [2], the definition of a product is an item that satisfies a market's want or need. Product/Service Catalogue Management is the ability to create and maintain products that can be sold to customers in the target market. More specifically, it is the ability to explicitly model the structure of a product, then create and centrally manage the instances (or "catalogue") of products based upon that structure. Products are not always discreet, single items. A product can be a number of components associated together and sold as a single purchasable entity. Therefore, the product may be comprised of multiple components, tangible or intangible, such as services, features, devices, etc., that are "assembled" together to form a single sellable entity. Some of the components within a product will be enabled by shared/common/reusable services (e.g., location finder). Some of the components within a product will be enabled by shared/common/reusable resources (e.g., network exchange). These underlying services and resources may be managed by different parts of the organization.

2.3.3. Corporate Sales Management

As initially specified in D2.2 [2], the Product Management organization is typically responsible for managing the Product/Service Catalogue through the assembly and update of products utilizing available components of the Public 5G services and Application Catalogue. Customer contact management, retention and loyalty applications are a varied group of functions that are generally sold as part of a Customer Relationship Management (CRM) suite of applications. In general, these applications allow an operator to create, update and view the customer's information (names, addresses, phone numbers, organizational hierarchy), record and view all customer interactions across different communication channels and department. This way whoever is speaking to a customer can see the history of issues that have concerned that customer, be they order issues, billing enquiries or service problems. More sophisticated systems allow capabilities to highlight customers as risk of switching to an alternative carrier (churn indicator) and provide comparisons with other operator's service packages to allow customer care agents to try to persuade a customer that their current operator can provide the best value for money. These indicators can be provided via integration to business intelligence platforms.

2.3.3.1. Overview

The Corporate Sales Management application provides the necessary functionality to manage the sale to a medium to large business customer. This is fundamental in the case of Neutral Host model. Besides, given convergence occurring in the industry in the area of products and customer classification, along with the need to manage relationships across corporate and mass market customer segments, it is expected that at some point significant aspects of the Corporate and Mass Market Sales Management applications will come together into a single set of applications.

2.3.3.2. Functionality.

With regard to Telecom experience, Corporate Sales Management includes the following:

- Management of sales accounts, including the handling of leads and funnels.
- The design, price, propose lifecycle.
- This could include responding to RFIs, RFBs, etc., as well as a number of iterations on the Design/Price/Propose lifecycle (aspects fundamental in the Neutral Host model).
- Negotiation and closure on a formal contract with the customer.

2.3.4. Contract Management **Functionality**

As initially specified in D2.2 [2], the Contract Management handles the creation of the customer's contract and any associated service level agreements, including approval of custom language, customer contract sign-off, appropriate counter signature and contract expiration. Elements of the contract will flow through to ordering, assurance, and billing processes.

2.3.5. **Customer Order Management**

As initially specified in D2.2 [2], the Customer Order Management applications manage the end-to-end lifecycle of a customer request for products. This includes order establishment (step guiding, data collection and validation), order publication as well as order orchestration and overall lifecycle management. A customer request may also pertain to already purchased product(s). Thus, the Customer Order Management application handles order requests to suspend, resume, change ownership, amend, add, change and discontinue existing ordered products. Customer Order Management application should support repackaging of the purchased offers into alternate product offering (may require sales/contract negotiation). Customer Order Management applications typically serve all the customer touch points/channels, including call center, retail, self-service, dealers, affiliates, etc. The order may be initiated by any channel and visible to the other channels if needed. The general schema has to be tailored according to the Neutral Host model.

2.3.6. **Service Order Management**

As initially specified in D2.2 [2], the Service Order Management applications manage the end-to-end lifecycle of a service request. This includes validating service availability as well as the service order request. Other functionality includes service order issuance, service and or product order decomposition, and service order tracking along with orchestrating the activation and the test and turn up processes. Notifications will be issued to the Customer Order Management during the service order orchestration process (especially upon completion). Such notification can trigger other steps in the Customer Order Management (e.g. service order completion concludes these steps with Customer Order Management).

3. Orchestration & Control Layer

The orchestration and control layer of 5GCity is structured starting from an entry point for infrastructure and service management (Dashboard), and develops down across the orchestration layer introduced in previous sections where the core orchestration components are placed: i.e. the 5GCity orchestrator, the controllers that reside between the central orchestration platform and the infrastructure, namely WAN managers, VIMs, and SDN controllers.

The scope, high-level functionalities, and relationships of these elements are described in this section.

3.1. Dashboard

The 5GCity Dashboard is the entry point for all 5GCity's envisioned actors/stakeholders aiming to interact with 5GCity platform. This component provides a web-based browser interface that allows users to use 5GCity's platform functionalities through an intuitive and appealing interface. The specification provided in 5GCity's deliverable D2.2 [2] was respected and therefore accurately represents the current implementation of Dashboard subcomponents. Figure **10** provides a more detailed view into the internal architecture of the 5GCity Dashboard including the technology base used in each subcomponent.

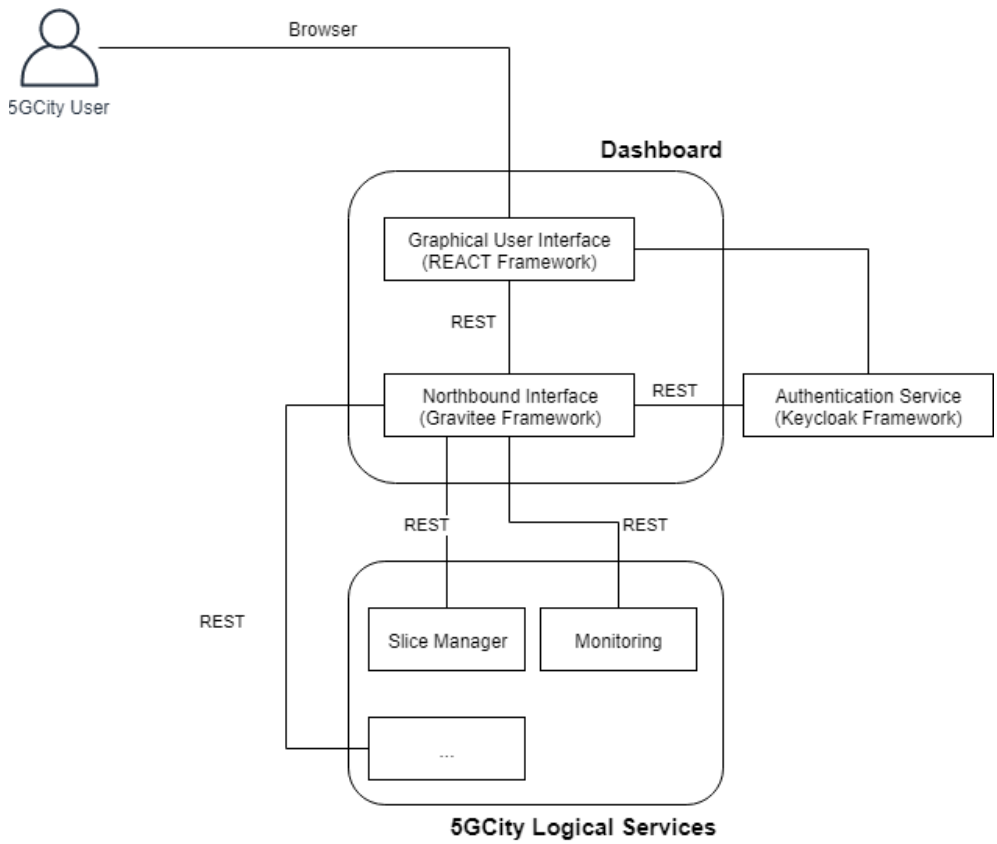

Figure 10 – 5GCity Dashboard reference component design

As illustrated in the Figure 10, the Dashboard is currently composed of two sub-components:

- **Graphical User Interface (GUI)**

The GUI will be the interface provided to 5GCity's platform end users exposing its features through an appealing and intuitive interface. The graphical user interface adapts both its content and available features based on the user's permission role and tenant.

- **Northbound Interface (NBI)**

  This subcomponent provides a REST interface to 5GCity platform third parties (including the GUI) enabling the access to 5GCity data and features. To achieve this, the component interacts with internal 5GCity services retrieving the requested information from one or multiple sources of data. All endpoints are protected with authentication and authorization features. For authorization, NBI relies on an external subcomponent (Authentication Service). Based on the authentication information of a specific user, NBI either accepts or rejects a given request as well as filters the information returned in a given request, based on both the user's role as well as the user's tenant.

3.2. 5GCity Orchestrator

The 5GCity orchestration platform introduces the capability to offload capacity to the Neutral Host (by allowing it to manage network slices that are provisioned to network operators and service providers) and to support a large number of devices in the network. The 5GCity orchestrator is a core management component of the functional 5GCity architecture, which is responsible for lifecycle management and orchestration of all 5G-based edge services and for the control and management of the 5G and edge infrastructure available in the city.

The design of 5GCity orchestrator relies on the core NFV orchestration functionalities offered by the ETSI Open Source MANO platform, complemented with the additional functionalities needed to implement the Neutral Host concept and support combined management of edge nodes and 5G access sharing, as specifically envisioned by the 5GCity project. These extra functionalities are classified into four main components, namely Resource Placement, SLA Manager, Slice Manager, and Infrastructure Abstraction. Figure 11 depicts the high-level architecture of the 5GCity orchestrator.

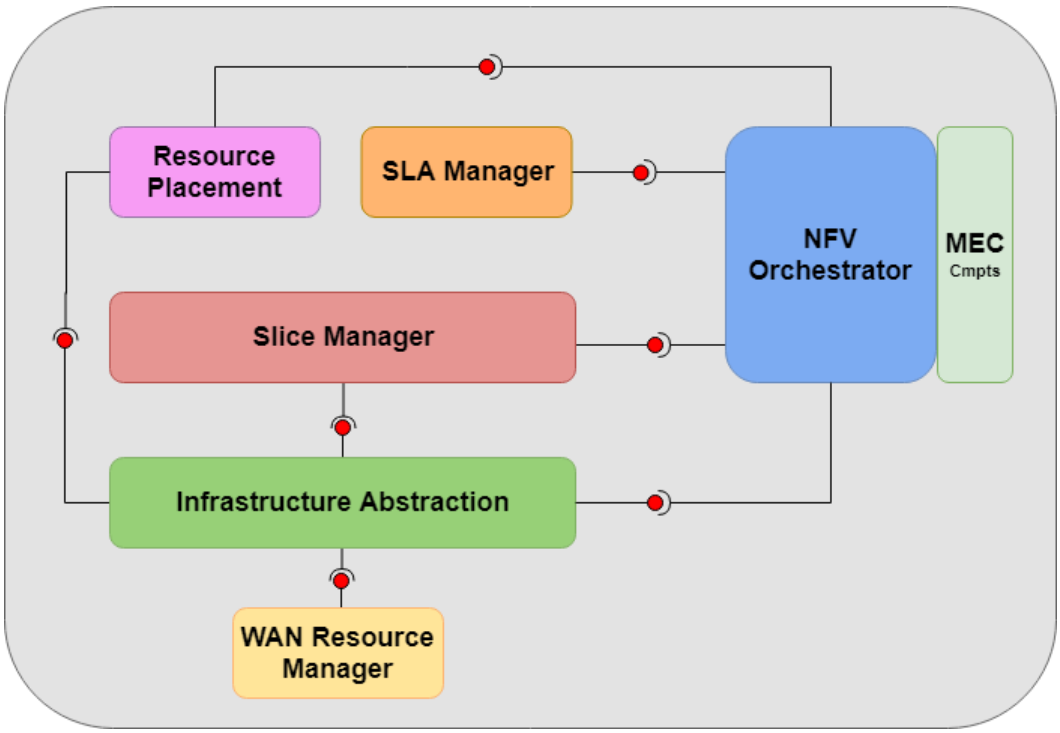

Figure 11 - 5GCity orchestrator high-level architecture

A summary of the main components illustrated in Figure 11 (NFVO/MEC, Resource Placement, SLA Manager, Slice Manager, Infrastructure Abstraction, and WAN Resource Manager) is provided in the following subsections.

3.2.1. NFVO and MEC components

The NFVO shall be the core of the 5GCity Orchestrator with all the functionalities of the ETSI Open Source MANO (OSM) platform, which is responsible to talk to other components of the 5GCity orchestrator for managing and orchestrating the network elements. The NFVO covers the on-boarding of Network Services, for which it receives requests from the Dashboard, the orchestration and lifecycle management of the physical and virtual resources, and the lifecycle management of the Virtual Network Functions (VNFs).

The NFVO shall have the functionalities described in the respective ETSI NFV specification, while including additions that are required to support (and/or incorporate) all the other components of Figure 11, as well as additions that are required for the support of Multi-access Edge Computing (MEC). OSM has been chosen over existing orchestration platform in order to guarantee the performance, high service scalability, and adoption in the 5G research line. Further, the OSM is modular enough for extending its capabilities (internal and external functions/plugins). Additional reasons for selecting OSM as the core of our orchestration platform are highlighted in [4] .

In order to provide the mentioned MEC support, two main options were analysed:

a) Either NFVO or MEAO (MEC Application Orchestrator) acts as a “master” receiving all service on-boarding and deployment requests, appropriately forwarding the requests that are meant for the other orchestrator.

b) A specific “dispatching layer” is developed on top of the two orchestrators, which will receive all requests, perform the appropriate checks by looking into the descriptors and the details of the target slices, and forward to the appropriate orchestrator if such a service deployment is possible.

5GCity opted for option b, introducing a component called “Multi-Tier Orchestrator” (MTO), with the described functionality. This offers a simple interface for accepting “generic” service requests, which is an abstraction (or simple “forwarding”) of the NBIs (Northbound Interfaces) of the underlying orchestrators. Relating this to the architecture of Figure 11, the MTO can be seen as part of the MEC components.

3.2.2.Resource Placement

Resource Placement is responsible for optimizing the allocation of Virtual Network Functions (VNFs) to physical and virtual resources of the infrastructure layer. It calculates optimal placement options by analysing the resource usage along with further information from the Virtual Machines (VM).

5GCity opted for focusing the placement a lot on security restrictions, so that the logic of the 5GCity Resource Placement module makes sure that VNFs run only on VMs with TEEs (Trusted Execution Environments) when Slice Users request a high level of security.

3.2.3.SLA Manager

The 5GCity platform enables the Infrastructure Owner to slice the shared physical and virtualized resources across RAN and core networks so that they can support different industry verticals for an associated SLA. Network slicing will allow 5GCity infrastructure owners to offer differentiated and guaranteed services with varying traffic characteristics on the same infrastructure. In line with the 3GPP-defined 5G slice types, we expect the typical use cases for slicing to be related with one of the following:

- Enhanced mobile broadband (eMBB) that delivers Gigabytes of bandwidth
- Massive machine-type communication (mMTC) that connects billions of sensors and machines (1 million per square kilometre)

- Ultra-reliable, low latency communication (uRLLC) that allows immediate feedback with high reliability and enables real-time remote control.

The 5G typical slicing use cases provide clear evidence that the design and the deployment of a network slice is strictly coupled with a set of QoS parameters, which are translated in a certain SLA.

A specific service and the associated QoS characteristics are assigned to a network slice according to a given set of parameters that ensure the above configurations. It is clear that the same 5G infrastructure which is in charge for the deployment of the networks slice, must also take care to deploy a monitoring framework capable of continuously monitor the infrastructure and check in real time if the QoS/SLA parameters coupled with network slices are respected.

The main functionalities of the SLA Manager are summarized in the following:

- It controls the lifecycle of network slices in real time and proactively manage dynamic demands and failure conditions
- It stores the SLAs related to Network Services in an internal SLA Database
- It interacts with the Monitoring System based on an internal "Monitoring Driver" in order to trigger the activation on the underlying MANO infrastructure of the performance jobs, which collects monitoring data for a specific Network Service.

3.2.4. Slice Manager

Network slicing enables the virtualized and non-virtualized network elements and functions to be easily configured according to network operation requirements and isolated from the other slices in order to meet various network demands. These slices allow the virtualization of the physical network and the assignment of virtual resources (slices) to the different Mobile Virtual Network Operators (MVNOs).

5GCity's network slicing allows network operators to seamlessly control and orchestrate services for different verticals. Each slice has its own specific requirements related to the Quality of Service (QoS) policy, cloud network resource management capability, security functions, and routing functions to name a few. The 5GCity network slicing can perform the following main functions:

- Dynamic provisioning and instantiation of end-to-end network slices that include both computing and networking resources.
- Interaction with different edge VIM technologies for better support of multi-tenancy and multi-tier infrastructures.
- Seamless and dynamic service provisioning at the network level through automated processes.

The listed main functions can be enabled by combining:

- A network slice lifecycle management method that adds dynamicity with regard to the ease and frequency of slice adjustment.
- A policy management method that can enforce dynamic runtime policies over the deployed slices, such as instantiation and expiration dates.
- A slice information repository that is compatible with suitable information models for 5G end-to-end slices.

3.2.5. Infrastructure Abstraction

The Infrastructure Abstraction is a component of the 5GCity orchestrator that enables communication between the orchestrator and the multiple underlying Virtual Infrastructure Manager (VIM) with the support of the WAN Infrastructure Manager. For this, the Infrastructure Abstraction will require a VIM plugin and a

WAN plugin in order to interact in a technology-independent manner with the Virtual infrastructure and WAN infrastructure:

- The VIM plugin covers the Core VIM, the Edge VIM and the Extended Edge VIM interaction requirements. It should allow the 5GCity orchestrator to interact with VIM technologies and implementations such as OpenStack, Edge-VIM (which will be an edge-oriented OpenStack extension), and Fog05[1]. Note that a VIM plugin for Fog05 support has been contributed to by 5GCity to the ETSI Open Source Mano (OSM) software development community.
- The WAN plugin exposes an interface to the WAN Resource Manager element for dealing with the underlying network elements such as SDN controller, fronthaul network, and backhaul network. This keeps the infrastructure abstraction layer relatively simple and poses no restrictions on how the WAN Resource Manager controls the WAN network resources.

3.2.6. WAN Resource Manager

The WAN Resource Manager consists of software elements which allow to talk to the lower-layer network controllers (SDN/RAN controllers, etc.).

During the development of the WAN Resource Manager, it has been identified that this component contains two (architecturally distinguishable) main parts (each of them including various software modules):

a) The part that "talks" to network controllers (esp. SDN controllers like OpenDaylight[2]) only for implementing (or complementing) network service deployment. This part can be implemented as extension to an NFVO (e.g. OSM[3]) and implement the function of a "WIM" (WAN Infrastructure Management).

b) The part that "talks" to (mainly access) network controllers (e.g. the i2CAT RAN controller, a Ruckus[4] Wi-Fi controller, etc.) for configuring the controlled network entities as required for slice creation and setup. This part also talks to SDN controllers (such as OpenDaylight, again) for retrieving all information and doing all configurations that is not directly related to network services deployment/lifecycle (e.g., retrieve configurations of deployed switches/routers).

It has been further identified that while the first part can be potentially implemented as an extension (or a plugin) of an NFVO, while the second part can be implemented as an abstraction layer very similar to the VIM plugin of the "Infrastructure Abstraction" (with the difference that it abstracts access to SDN controllers instead of abstracting access to VIMs).

3.3. Virtualized Infrastructure Manager (VIM)

VIM (Virtualized Infrastructure Manager) implements the functionalities that are used to control and manage the interaction of a VNF with computing, storage and network resources under its authority, as well as their virtualisation, [15].

Three different types of VIMs are used in the 5GCity Architecture: VIM-Core, VIM-Edge and VIM-Extended Edge. Two of them are based on OpenStack[5] (Core and Edge), while the VIM-Extended Edge is based on the lightweight decentralized open source technology Eclipse fog05, designed to support IoT and Fog computing environments. The interoperability among these three types of VIM will be either native (i.e. at OpenStack

[1] The http://www.fog05.io site is not reachable at the moment as it is moving the domain to the Eclipse Foundation
[2] Please see: https://www.opendaylight.org/
[3] Please see: https://osm.etsi.org/
[4] https://www.ruckuswireless.com/
[5] https://wiki.openstack.org/wiki/Main Page

level between VIM-Core and VIM-Edge) or implemented through the 5GCity Infrastructure Abstraction layer provided by the 5GCity multi-layer orchestration component.

This partitioned VIM architecture and its differentiation strengthens the 5GCity capability to scale. In fact, with multiple VIMs we can more easily address the OpenStack scalability issues related to the number of compute nodes to be managed by a single VIM. Moreover, the differentiation of VIMs assures that 5GCity can benefit from both OpenStack and fog05 key features applied to the different contexts of core, edge and fog computing.

3.3.1. Core

VIM-Core operates homogeneous physical resources located in the city data center. For this role, the project will leverage on the capabilities of OpenStack.

3.3.1.1. OpenStack

OpenStack is developed as an open-source project with the goal of being a cloud operating system managing large-scale compute, storage and networking resources. Its logical architecture (Figure 12) is composed of modules, developed as separate projects.

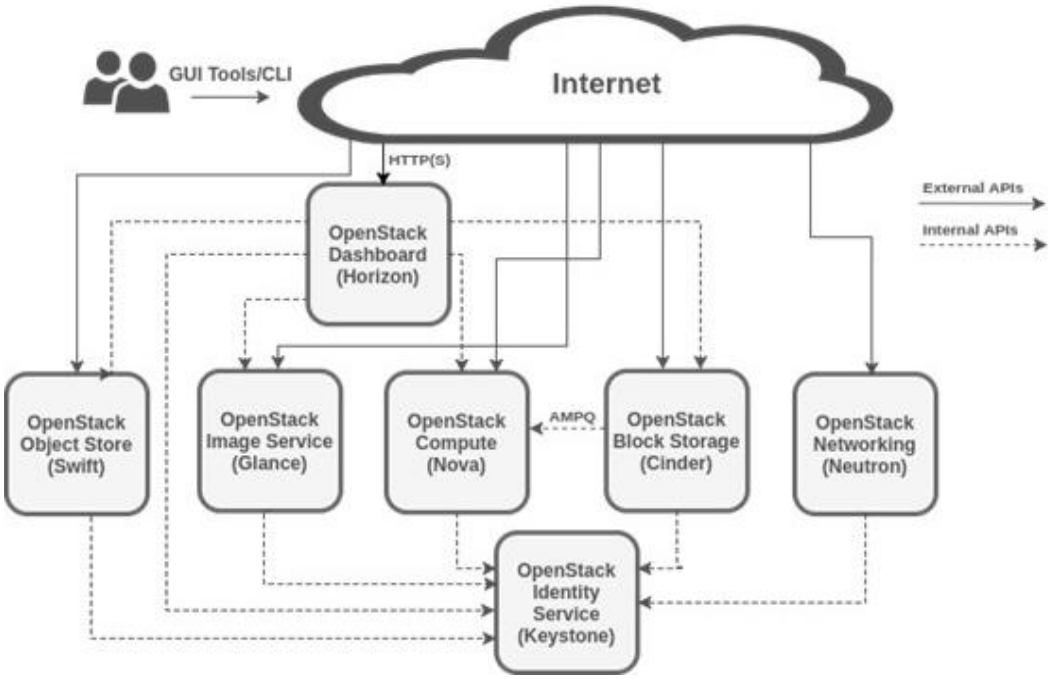

Figure 12 - OpenStack architecture

- *Compute* creates and terminates virtual machines' (VMs) instances, tracks the inventory and usage, takes requests from the message queue and determines on which host to run VMs.
- *Networking* gives full control over creation of virtual network resources to tenants in the form of tunnelling protocols.
- *Image* provides users with the capability to upload and discover VM images and metadata definitions.
- *Identity* has the main purpose of authorization and authentication of users.
- *Object Store* is a highly available, distributed, eventually consistent object/blob store.
- *Block Storage* virtualizes the management of block storage devices.

- *Dashboard* provides a graphical interface to access, provision, and automate deployment.

OpenStack is based on an open-source software with an active community. It has the ability to control pools of compute, storage and networking resources and supports multi-tenancy. Its modularity, extendibility and open APIs bring added value to its users. These qualities make him the appropriate match for the role of the VIM-Core in the project.

3.3.2. Edge

VIM-Edge operates a non–homogenous, wide area, resource constrained set of physical resources located in street cabinets across the City. Such dispersed infrastructure is vulnerable to malicious attacks (man-in-the-middle, device tampering, etc.) and needs extra security measures.

3.3.2.1. EdgeVIM

EdgeVIM is a collection of OpenStack extensions developed during the 5GCity project, adding trust into the edge infrastructure by leveraging ARM TrustZone[6] technology. Its goal is to harden and protect the edge compute infrastructure. OpenStack is chosen as a basis for the EdgeVIM implementation after a comparison of existing open-source VIMs [OSCOMP]), because of its flexibility allowing for the creation of custom solutions and the regular updates and support.

The EdgeVIM consists of two main parts (**Figure 13**):

- OpenStack extensions: Attestation Filter added to the Open Stack Compute (Nova) scheduler and an attestation agent running on each compute node.
- Trusted apps: implemented inside a Trusted Execution Environment provided by ARM TrustZone and OP-TEE[7]. They perform node authentication, kernel integrity verification and geo-fencing.

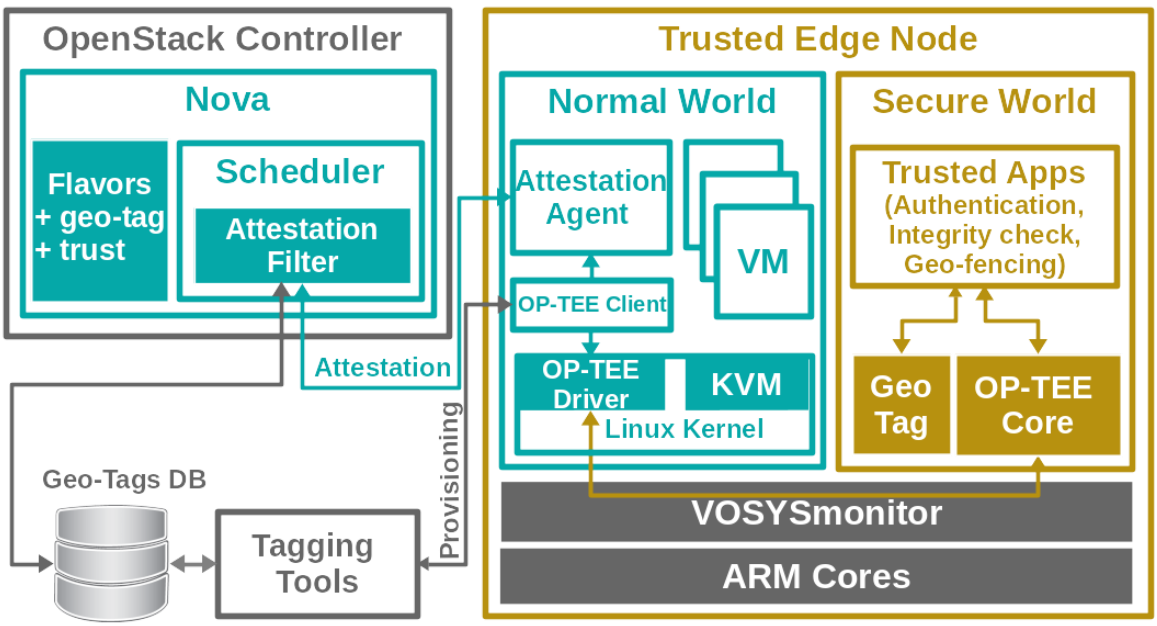

Figure 13 - EdgeVIM architecture

Detailed descriptions for EdgeVIM can be found in deliverable D3.1 [13] and deliverable D3.2 [14].

[6] Please see: https://www.arm.com/products/silicon-ip-security
[7] Please see: https://www.op-tee.org/

3.3.3. Extended Edge

The VIM-Extended Edge operates on a heterogeneous set of resource constrained devices that are typically installed in the lamp post; those kinds of devices are not capable in VMs workloads, but is worth to harvest computer power from those devices.

3.3.3.1. Eclipse fog05

Eclipse fog05 is an IaaS software that implements the VIM for the Extended Edge. The 5GCity project contributed to Eclipse fog05, specifically adding a set of requirements coming from a neutral host perspective.

It is worth to mention that Eclipse fog05 is a different kind of VIM, because of the high heterogeneity and error prone connectivity that can be present in an Extended Edge/Fog environment, it is completely distributed with no master/controller nodes, and state distribution across all the nodes of the system.

Eclipse fog05 is composed by:

- A distributed Key Value (K, V) Store used for state distribution across the system
- An agent running on the nodes that expose them to the VIM
- A set of plugins for managing different kind of hypervisors (VMs, Containers, binaries), and network connectivity

The possible deployments of Eclipse fog05 and his high-level architecture are described in Figure 6.

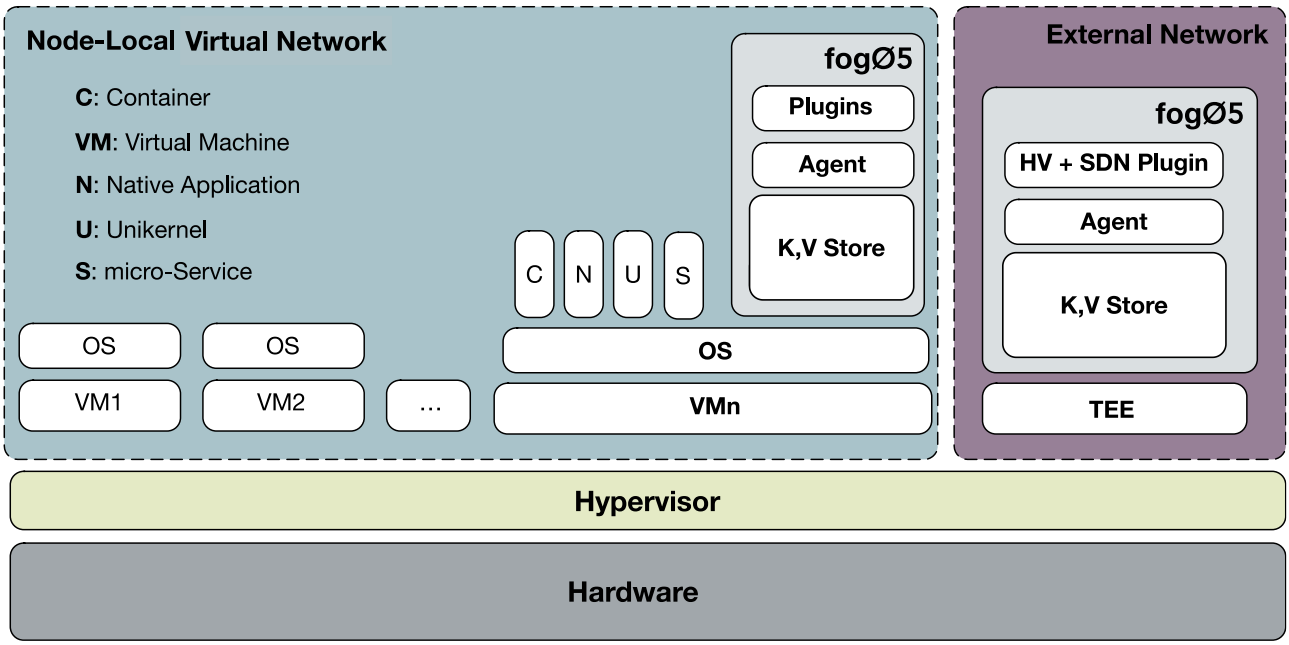

Figure 14 - High-level architecture of Eclipse fog05

3.4. SDN controller and agents

Similar to how VIM instances like OpenStack or fog05 are used to manage and assign compute resources to a slice, 5GCity relies on SDN-enabled RAN controllers to enable slicing of the wireless medium. The RAN controllers are deployed as part of the platform and serve as interface between the WAN resource manager and the physical, wireless devices, such as Wi-Fi nodes and Small Cells. The WAN manager can support several RAN Controllers, translating requests for slices coming from the slice manager and handing them to the responsible RAN controller. The same way different VIMs are required for different types of compute resources (e.g. fog05 for constrained devices), the type of wireless technologies to be controlled can be

different in each deployment. As such, a RAN controller that integrates into 5GCity has to support one or several wireless technologies and it has to expose the function calls that are necessary to implement 5GCity wireless slices to the slice manager.

In 5GCity three wireless technologies are supported: the custom built i2CAT Wi-Fi nodes, the Accelleran small cells and the commercial Ruckus Wi-Fi solution (https://www.ruckuswireless.com/). The custom RAN controller solution (now called *RACOON*) introduced in previous deliverables was built from scratch to support the control of the i2CAT Wi-Fi and the LTE resources, whereas for the Ruckus Wi-Fi nodes used in Bristol the commercial Ruckus controller is used. For each of the two RAN controllers a software module was developed for the WAN manager that translates requests from the slice manager to the specific calls supported by the RAN controllers. Further RAN technologies could be supported if additional modules are implemented in the WAN manager.

Figure 15 depicts the final configuration of the RACOON RAN controller along with the underlying SDN client solutions used in the LTE and i2CAT Wi-Fi nodes. Although the high-level idea of the SDN agents for Wi-Fi radio have not been radically changed, there is an updated internal structure of the RAN SDN controller components with regard to the detailed submodules and the used technology stacks.

In the updated design we have the RAN controller element (RACOON) developed in 5GCity, which has a northbound interface towards the rest of the 5GCity architecture and a REST-based southbound API that talks to i) the NETCONF[8] module ii) the ODL SDN controller. The NETCONF module talks via NETCONF to the RAN elements, whereas the SDN controller talks OpenFlow[9] and OvSDB with the Wi-Fi nodes. Please not that for LTE, the L3 hosted originally in the Small Cells has been virtualized, forming now another platform Element connected to the small cells that can be configured via NETCONF. The 5GCity platform also supports the integration third party RAN controller solutions, such as the previously mentioned Ruckus solution that is deployed in Bristol.

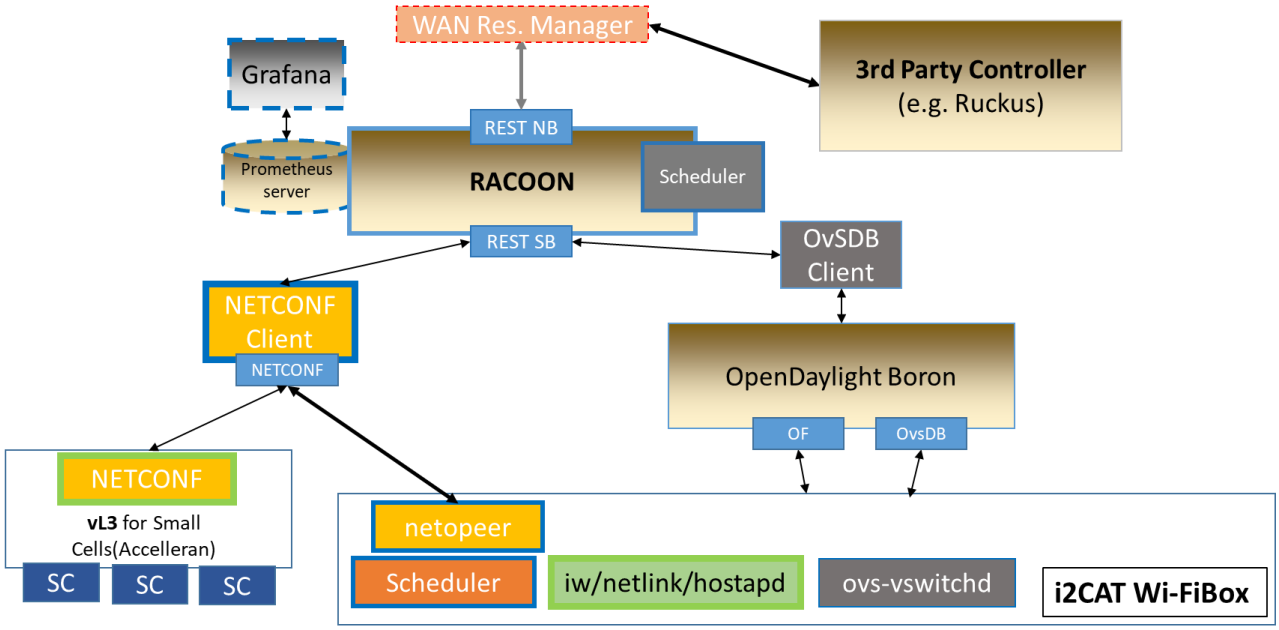

Figure 15 - Internal structure of the RAN SDN controller components

[8] Please see: https://tools.ietf.org/html/rfc4741

[9] http://www.opennetworking.org/wp-content/uploads/2013/05/TR-535_ONF_SDN_Evolution.pdf

3.5. Monitoring Framework

Service monitoring refers to all the activities of collecting, aggregating, analysing and displaying data relating to a system. A monitoring framework allows observing the status of a system and to predict / prevent possible problems before they affect its operation, optimizing user's experience. The data to be analysed are obtained by extracting appropriate metrics from the system, metrics such as the memory, disks, CPU usage, etc., with respect to the system and / or to particular applications instances. This metrics extraction process is called instrumentation. Each service, along with its own characteristics, is different from any other service and there are many parameters that can be monitored. Metrics can range from low-level resources such as CPU usage to high-level business metrics such as the number of registrations.

Based on these definitions, the monitoring system for the 5GCity platform is designed to monitor the overall virtualized resources (calculation, storage and network) of the three-tier architecture and, through an appropriate set of parameters, the applications and services running on the 5GCity infrastructure.

Regarding the ***infrastructure***, the monitoring system includes three different resource domains, i.e. 1) NFVI resources; 2) SDN-enabled elements; 3) physical devices that do not belong to the first two categories; while with regard to the ***applications and services***, the monitoring system includes VNFs and service monitoring parameters (metrics, useful also to check SLA compliances).

The monitoring system can be integrated with the different orchestration layer components to assist in the network system management. The 5GCity system monitoring is able to provide capabilities to monitor both ***network*** and ***cloud infrastructural*** elements and the related services with a full end-to-end view. To get this feature, the monitoring system has been designed in a global perspective, with a view to the different services composing the overall infrastructure.

The monitoring system implemented in the 5GCity project is able to instrument and monitor the different devices composing the overall infrastructure providing a unique and simple-to-access view of the 5GCity platform exposed to both dashboards and analytical techniques collecting and providing all the information needed in a "monitoring as a service" model.

3.5.1. Monitoring Module

The main components of the monitoring system in the 5GCity platform (Figure 16) are the functionalities related to the monitoring of the overall virtualized resources (compute, storage and network) in the three-tier architecture, as well as a set of parameters related to applications and services running on the 5GCity infrastructure.

The group of infrastructure components includes three different domains of resources:

- **NFV Infrastructure (NFVI) resources** that comprise compute, network and storage virtual resources;
- **SDN-enabled elements**, including physical and virtual resources, which are usually controlled by a SDN controller;
- **Physical devices** that do not belong to the previous categories, such as non-SDN compliant network routers and switches, Small Cells, PNFs and other devices for which we are interested in collecting monitoring information.

The second group of functionalities includes:

- **Virtual Network Functions (VNFs**) virtual machines performing specific network functionalities;

- **Service monitoring parameters** that represent metrics; they are tracked (meaning data collection) to check the level of compliance of a specific running service with the agreed SLAs.

The monitoring system is integrated with several orchestration layer components, such as the Resource Manager, the Slice Manager, the SLA Manager, and the OSS/BSS systems to provide decision support for multiple purposes such as security threat detection and mitigation, SLA assurance, resource optimization and root cause analysis.

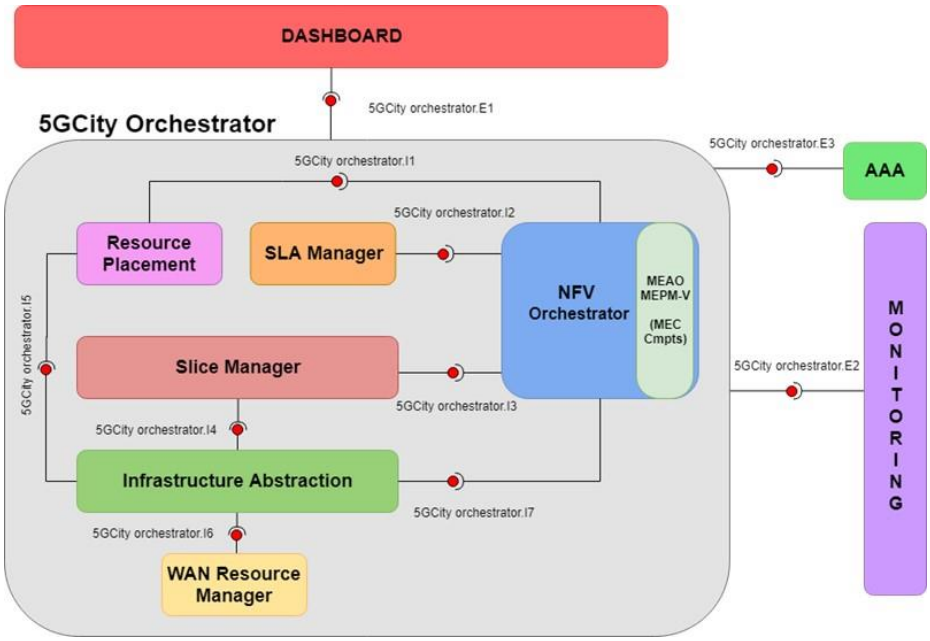

Figure 16 - 5GCity Platform design and interface to Monitoring framework

The monitoring framework keeps track of the key performance metrics within the 5GCity, distributed infrastructure, in particular:

- CPU, Ram and Hard Disk utilisation: especially on the MEC nodes, belonging to cabinets and lampposts, where it is important to monitor the resources allocation and possibly keep it as low as possible;
- Virtual network devices utilisation, i.e. bitrate on virtual link, packets metrics on data flows, etc.
- System physical resources, i.e. Radio resources, LTE and Wi-Fi, PNF (Physical Network Functions).

These parameters can be either VM-related information (e.g. CPU utilisation, bandwidth consumption) or VNF specific such as, calls per second, number of subscribers, *number of rules, flows per second*, VNF downtime, etc. One or more of these parameters, depending on the implemented logics, could also trigger a reaction on the QoS loop.

At Services level, monitoring parameters represent metrics that are tracked to check the level of compliance with the agreed Service Level Agreements (SLA).

3.5.2. Monitoring system

Different tools are available for system monitoring and alerting, including built-in and active scraping, storing, querying, graphing, and alerting based on time series data. Then we decided to use available open source

software for monitoring and visual representation in order to focus the most of the efforts to develop a software layer, over the selected tools, able to provide an ***end-to-end model***.
In 5GCity, Prometheus [10] has been selected as open source to fulfil the monitoring system requirements, along with Grafana for data analytics and visualization [11].
The main features of the monitoring system are summarized in the following:

- System modelling and inventory (node definition and exporters) via GUI or APIs (web services);
- Single node management;
- Service (cluster of nodes) management;
- Collection of metrics related to system status;
- Definition and configuration of monitoring parameters in the Monitoring manager;
- Fetching of monitoring data for a particular service or slice;
- Grafana integration.

Not everything can be instrumented. Applications that do not support Prometheus metrics natively can be instrumented by using exporters. Node Exporters can collect statistics and existing metrics, and convert them to Prometheus metrics. An exporter, just like an instrumented service, exposes these metrics through an endpoint, and can be scraped by Prometheus. Figure 17 synthetizes the deployment scenario selected in the 5GCity Project.

With the proposed monitoring framework architecture based on Prometheus and the exporters in various VNFs and system elements, it can be possible to collect various metrics from the NFVI and from the application layer.

To do this task Prometheus monitoring system talks to the remote exporters (on monitored systems) with a single request (pull mode) to a read-only API exposing gathered metrics.

It has to be pointed up that even if the node exporter can include many metrics, it cannot cover all the possible. To overcome this limit comes in the textfile collector which allows to extend machine instrumentation for the specific needed.

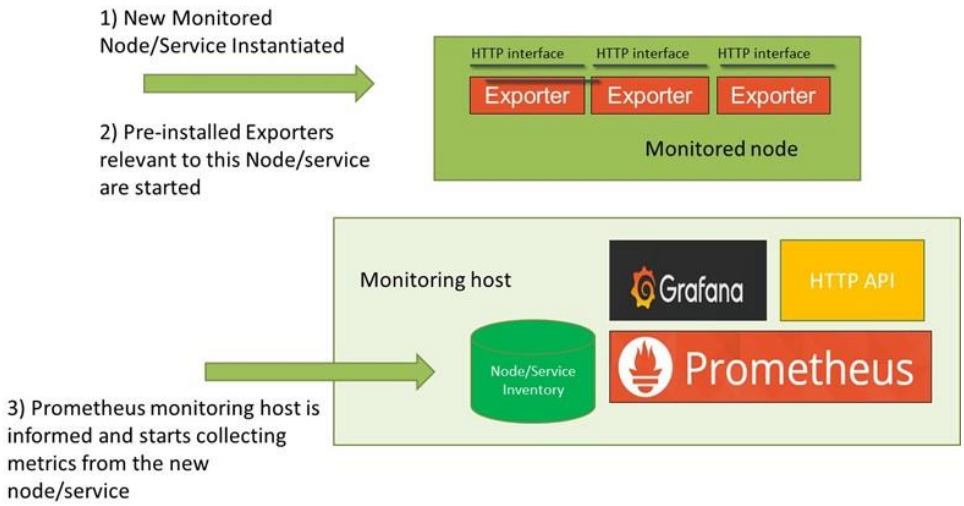

Figure 17 - Deployment scenario selected in 5GCity

General categories of metrics that can be collected include:

- System level parameters (i.e. %CPU, %RAM, %disk)

- Network level parameters (i.e. link bandwidth, TX/RX packets, TCP/IP protocol statistics, packet loss, RTT)
- Service creation times in 5GCity platform
- Utilization of virtualised resource (e.g. processor, memory, disk) that are allocated to a network slice instance

Each specific test on network layer or the 5GCity platform or the use case application will refine the list of parameters and measured KPIs, by adding also the application specific metric that can be used to evaluate the use case performances (e.g. for media applications: Video Streaming Start Success Ratio, Video Streaming Start Delay (s), Video Streaming Stall Frequency, Video Streaming Download Throughput).

4. Infrastructure Layer

In addition to the Service layer and Orchestration & Control layer, 5GCity also make specific improvements at the infrastructure layer to ensure that its KPI targets are achieved. The different improvements and developments at infrastructure layer are detailed in subsequent sub-sections.

4.1. Core NFVI (data center)

As already described in D2.2 [2], in 5GCity architecture cloud and software-defined infrastructure are integrated to offer support for network slicing and automated orchestration. In particular, the NVF infrastructure (NFVI), which includes physical resources (computing, storage and networking equipment) and specifies how these can be virtualized. The 5GCity platform offer different Virtual Network Functions (VNFs) running over the NFVI. The NFV Management and Orchestration (NFV MANO) offered by 5GCity Platform support the infrastructure virtualization and the lifecycle of the different VNFs.

In particular, according to the D2.2 [2] description, at DC level, the NFV and the VIM are deployed to orchestrate and manage the service and resource pool. The Network slicing provides the requested network capabilities to deal with the design, installation, termination sharing and monitoring. Then at DC level we have also to deal with the Network Function discovery and selection and with the Mobility and session management using the 5GCity platform capabilities.

4.2. Backhaul Network

In this section we are going to describe the relevance of the Backhaul network in the 5GCity architectural design, for 5G networks. In mobile networking, in general there is a huge amount of data needed to be carried from the data centers to the Cloud or the Operator CPD. This makes the backhaul to be carefully dimensioned and planned in order to satisfy the needs of the MNOs. Commonly this is an IP/MPLS network and the expected data rates for 5G systems are not less than 10Gbps.

For the 5GCity pilots in Barcelona, Bristol and Lucca, fibre networks form part of the infrastructure on top of which slices can be instantiated. As such, switches and routers that form part of the backhaul network need to support the data transportation of the deployed services and network functions. In the following two examples are given on why the backhaul network plays such a crucial role and how it can be integrated in the neutral host business model developed in 5GCity.

Scenario 1: Neutral Host with own network

Service providers (city council, infrastructure operators) dispose the needed infrastructure and networks to provide their own services or offering their infrastructure to other operators. These operators are characterized by:

- Having own networks to serve services.
- Having enough purchase capabilities to acquire and maintain the telecommunications systems.
- Following new business models in front of the margin reduction in the traditional services.
- Desiring the use of these infrastructures to hold new services and thus incorporating new technologies like NFV and SDN.

Slices provided by 5GCity to operators that host their own services and infrastructure are connected over the backhaul that is deployed in the 5GCity infrastructure, effectively extending the area where their services can be deployed. As a result, slice connected over the backhaul will complement the operator's capabilities allowing it to orchestrate the allocation of resources and services to the operators in an agile, dynamic and efficient way.

Scenario 2: Virtual Mobile Operator

Smallest operators often have many difficulties in offering value-added services. On the one hand, its services replicate those of the largest operators, whose networks they depend on. On the other hand, the large costs they incurr make innovation and differentiation very difficult. The results of this project will make the service component of virtual operators being more independent and productive while reducing their associated operational costs, so that they can provide their own added value services.

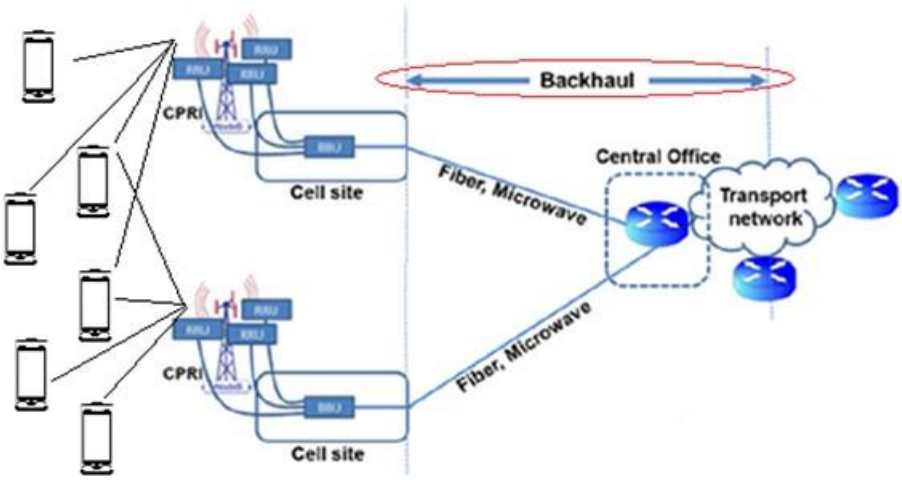

Figure 18 - Backhaul network

In 5G scenarios the existence of multiple virtualized network functions and network slicing mechanisms makes the network to be able to support all these functionalities alongside the delivery of such a huge amount of content. The use of all these functions will allow the consumption of a higher bandwidth and in consequence a higher data rate. All this capacity coming from the access networks must be managed by the backhaul - see Figure 18 - and with the minimum time it must be delivered in one way to the end users and in the other way to the data center, in order to be processed.

There is a set of key requirements for any network wanting to meet 5G service demands. This network will have to be reliable, operational and efficient for the following capabilities:

- More capacity per device (ultra-high capacity per end-device)
- More types of devices (introduction of IoT and M2M services)
- More devices (exponential growth)
- New services (Augmented and virtual reality, autonomous car, etc.)

The case that 5G is slicing the network and by function virtualization is adapting the bandwidth of each small cell leads to a very high data consumption as the spare bandwidth can be harnessed by the end users who need higher consume while current technologies (3G and 4G) does not harnesses this spare bandwidth.

In the 5GCITY architecture at the main DC level different Service VNFs hosting the controller which performs decisions are executed for cope with the heterogeneous access available technologies, like 5G, LTE and Wi-

Fi, as requested by the service-driven architecture proposed by 5G paradigm. Particular attention is focused on QoS and the Policy managements to support QoS for the different 5G services: eMBB, uRLLC, mMTC.

4.3. Edge NFVI - Radio

The Edge NFVI supports the middle tier of the three-tier 5GCity architecture and provides distributed compute and storage resources locally at the network edge (close to the access node) to support NFV instances which implement components of network slice service chains (especially at RAN level) and to provide an environment for Multi Access Edge Computing (MEC) applications. The following sections describe the edge networking architecture (itself implemented through NFV deployments) and the NFVI architecture for orchestrating and executing both networking and MEC application VNFs. Please note that this section focuses on the LTE NFVI; the Wi-Fi functions are not deployed as NFV, but rather as a subset of virtualized elements that are controlled via the SDN controller, as described in Section 3.4 .

4.3.1. 3GPP RAN Network Slicing and Virtualized Edge Architecture

The 5GCity RAN architecture (Figure 19) includes several innovations in the area of RAN functional disaggregation, RAN and Network Slicing with SDN control and RAN function virtualization for LTE.

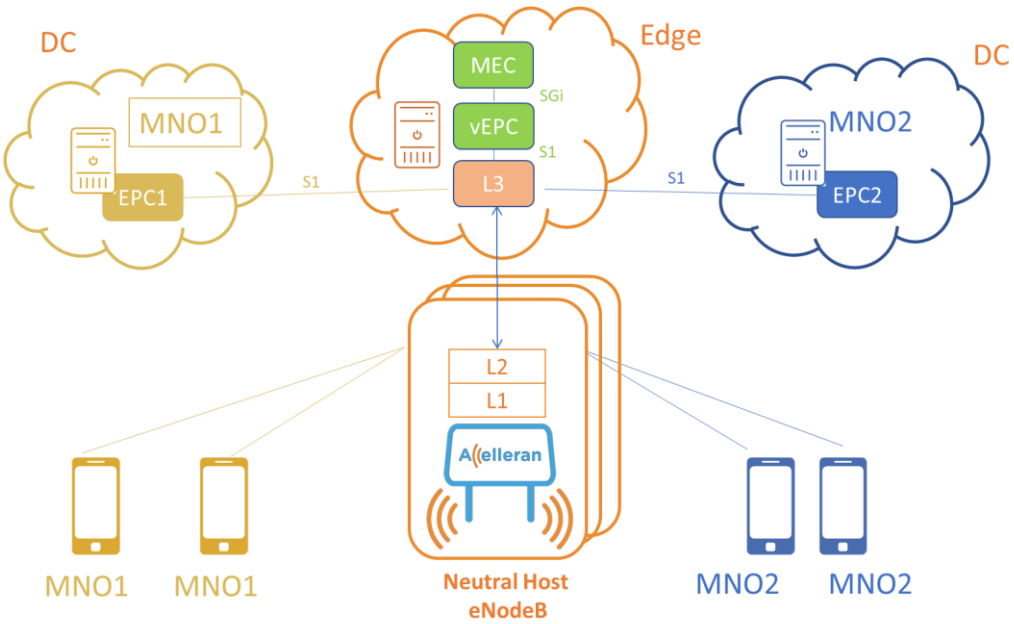

Figure 19 - 5GCity RAN Network Architecture Overview

Network functions in the architecture are:

- **LTE Layer 1**: The physical layer functions of the LTE air interface require specialised processing acceleration for DSP functions such as FFT, Turbo coding, etc and execute in each radio head on specialised DSP silicon. This is not a virtualised function in 5GCity and is common to all network slices.
- **LTE Layer 2:** The RLC and MAC functions of the LTE air interface are required to meet real-time schedules and are closely coupled to the LTE physical layer 1 implementation. These functions also execute in each radio head. RLC/MAC are not virtualised functions in 5GCity and they are common to all network slices.
- **LTE Layer 3**: The layer 3 (control plane) function of the LTE air interface is implemented as a virtual network function which runs in the Edge NFVI. 5GCity L3 supports network slicing and connection to multiple EPC (MME) instances (one per slice).

- **vEPC**: EPC (packet core network) is deployed at the network edge to support MEC access and low latency applications. Each instance of vEPC supports a network slice offering MEC application access.
- **Data center EPC**: The RAN functions support connectivity to EPC functions implemented at the data center. The neutral host use case assumes multiple EPCs and network slices to support access provision to multiple tenant service providers.

4.3.2. Mobility Management at the Edge

Mobility between cells at the edge is managed in accordance with standard 3GPPP procedures and interface. There are two cases to consider:

- Mobility between cells connected to the same edge node and vEPC (Figure 20): In this case mobility is handled within the RAN function with handover between the local eNBs and MEC connectivity maintained via the local vEPC

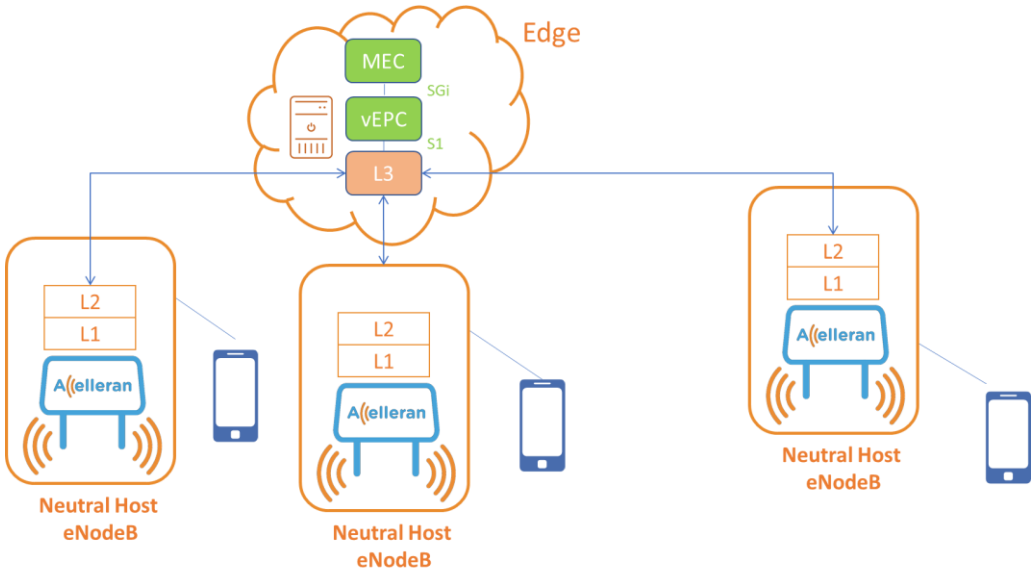

Figure 20 - Intra vEPC Mobility

- Mobility between cells connected to different edge nodes (Figure 21): In this case mobility involves inter-EPC procedures with service continuity to the MEC application maintained. It is for the MEC application to manage service migration between edge nodes should this be desired.

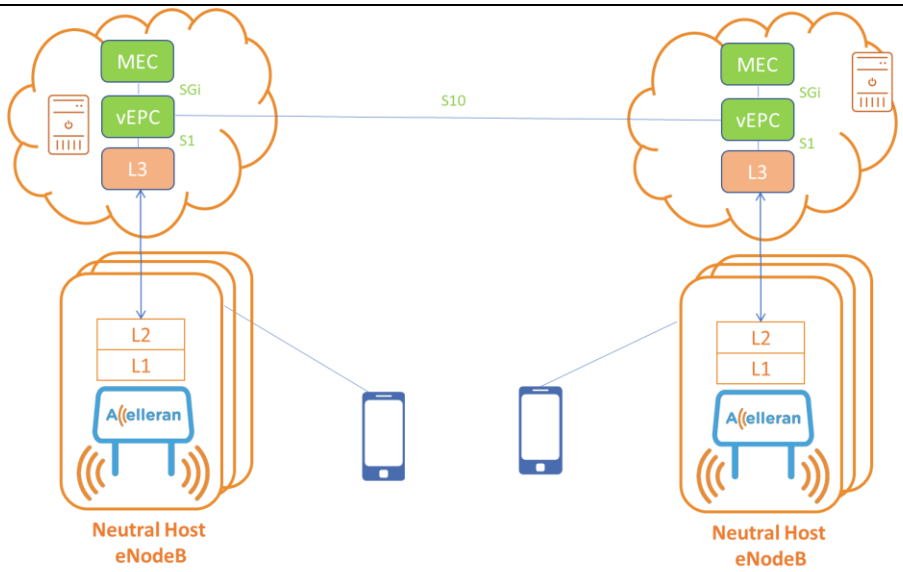

Figure 21 - Inter vEPC Mobility

4.3.3. Management and Orchestration of RAN VNFs

About Management and Orchestration of RAN VNFs, in the following are shortly described the eNB (FCAPS and SDN) and vEPC features.

- **eNB FCAPS**: RAN layer 3 is deployed as a Docker container. In addition to the NETCONF management below, CLI is used to initially configure the basic boot-strapping of each small cell to allow self-discovery via REDIS/NATS servers.
- **eNB SDN control**: RAN L3 provides an SDN control interface supporting the following functions
  - Initialisation
  - Slice Profile Creation
  - Slice Profile Modification
  - Slice Profile Deletion

  SDN control is implemented using the NETCONF protocol via a data-model which is formally defined in YANG. Figure 22 illustrates the RAN orchestration model in NETCONF / YANG developed for 5GCity in order to be manageable by the RAN controller.
- **vEPC:** The vEPC, for edge deployment, provides WebGUI and/or CLI based configuration interfaces.

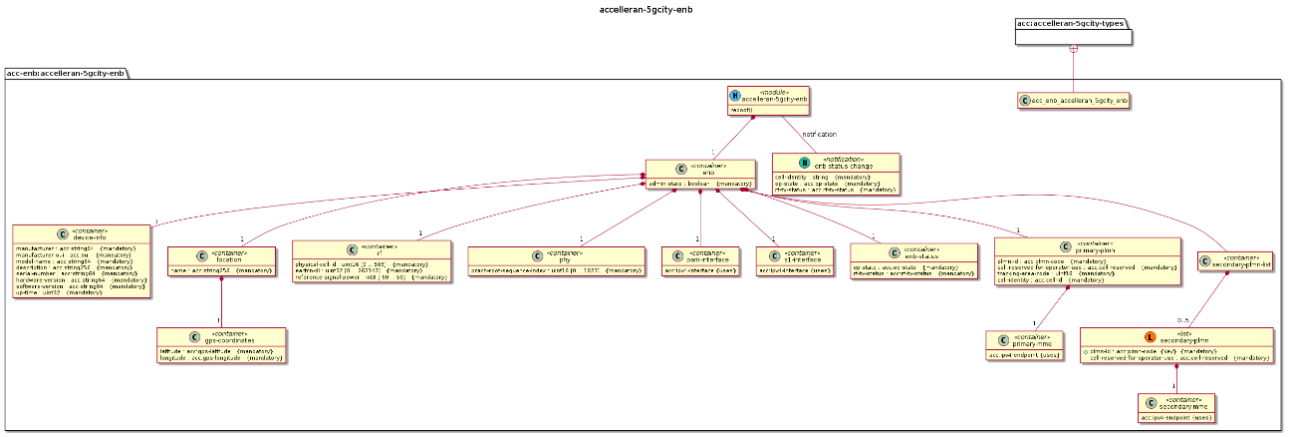

Figure 22 - RAN SDN Control Interface Model

4.4. Front-haul Network

With regard to the Front-haul network, it is worth to point out, in addition to the content in D2.2 [2], the two types of technologies exploited to connect the antennas: Fibre optics and Radio-links.

Optical fibre:
The link by means of optical fibre is what allows the antennas to offer the maximum benefits of the 5G technology to the users with the greater reliability and minor interferences in the signal.
The drawback of this solution is that it requires that the fibre optic network reaches the best antennas. In many cases, this requirement has to do with the roll-out of new civil work in the public highway to extend the network to the new destinations. In other cases, this solution is unfeasible due to the limited deployment of the fibre optic network at present, as it can happen in remote places or in some rural areas.
To minimize the impact of the new civil work, there are municipalities that take advantage of any action on the public space to deploy new fibre optic pipes to the urban furniture of the area, in anticipation of the future connection needs of the 5G.

Radio-links:
The link through radio links allows you to connect the antennas between them without the need of any physical infrastructure. This type of link is useful in those areas where there has not been a deployment of the fibre optic network. It can also be a solution to save costs in areas connected with fibre optic, but without pipes to the elements of urban furniture (where small cell antennas would be placed). In this way the city councils could save the costs of execution of civil works in public space.
The drawback of this solution is that its performance and reliability do not reach the service levels of fibre optic links, being susceptible to signal losses due to interference. In Spain, the different participating actors demanded to establish frequency ranges in addition to commercial frequencies to service these radio links, in order to avoid interference and have a greater amount of spectrum for these communications.

Besides the Small Cells, there are also additional equipment that complements the deployment and allows the operation of the sites, such as:

- Equipment to enable computing at the end of the network (MEC);
- Own electricity generators, to guarantee the operation of the antennas in critical areas, even in situations of general shutdown;
- Equipment for connection to the wireless backbone network.

4.5. Extended Edge NFVI

The Extended Edge NFVI support the closest to the user's tier of the three-tier 5GCity architecture and provides computing, networking and storage resources closest possible to the user (in the access node itself or in the same lamppost). It also provides an environment for Multi-access Edge Computing (MEC) applications.

This NFVI is managed by the Extended Edge VIM, Eclipse fog05, and everything is orchestrated by the Multi-tier orchestrator of the 5GCity platform. Is worth to say that the actual deployment of the instances is done by the ETSI OSM orchestrator that is able to communicate with Eclipse fog05, this connector has been developed within the 5GCity project and the process of contribution to the ETSI OSM community is in progress.

4.5.1. MEC in the Extended Edge NFVI

Within 5GCity project a PoC of MEC Platform was developed and contributed to Eclipse fog05 as well as the Mobile Edge Application Orchestrator and the MEC Platform Manager, those components are the key components of an MEC deployment. The MEC Platform is designed to provide a way to MEC Services and Apps to discover each other and provide or consume services coming from other apps or services. It was also developed with lightness in mind in order to be deployed on those devices that compose the Extended Edge NFVI.

4.6. Radio Element (for Small Cells and/or Wi-Fi)

In 5GCity two types of UE radio technologies are supported: Wi-Fi and LTE (provided via small cells). As described in Section 3.4, an SDN-enabled framework is provided as part of the 5GCity platform, providing the tools to configure and manage the RAN infrastructure for slice creation. In the following, the main characteristics of the RAN technologies are provided.

4.6.1. Small Cell Radio Element

The small cell radio element provided is the Accelleran Local Area base-station Class Outdoor E1000 Series small cells (see Figure 23 and Figure 24) available in different band variants depending on the spectrum available in the 3 cities. Barcelona uses 3.5GHz TDD (B42) variants, Lucca uses 2.6GHz TDD (B38) variants and Bristol uses both 3.5GHz TDD (B42) and 2.6GHz TDD (B38) variants. The E1000 Series is a compact, single carrier LTE eNB supporting 2x2 MIMO, 62 QAM DL/16 QAM UL and up to 64 simultaneously connected UE's. The unit is weatherproof (IP67) and designed for operation in a range of environments by means of its external N-Type RF and GNSS connectors. For the 5GCity use case easy to deploy directly attached fiberglass omnidirectional antennas are used (as in typical high-density urban environment deployment), although if it had been needed also dual pole directional panel antennas could have been used for directionality and higher gain, albeit with more complex installation. The backhaul connectivity is provided by a single Gigabit Ethernet physical connection, which also provides power to the unit (PoE+).

In the 5GCity deployment, the small cell units (normally able to run fully integrated L3/L2/L1) are configured to operate as L2/L1 remote radio heads which are managed by a virtualized eNB L3 control plane (Accelleran dRAX™ Open Interface RAN Intelligence) which runs at the edge. This L3 supports remote configuration via the NETCONF-based software used by the SDN controller.

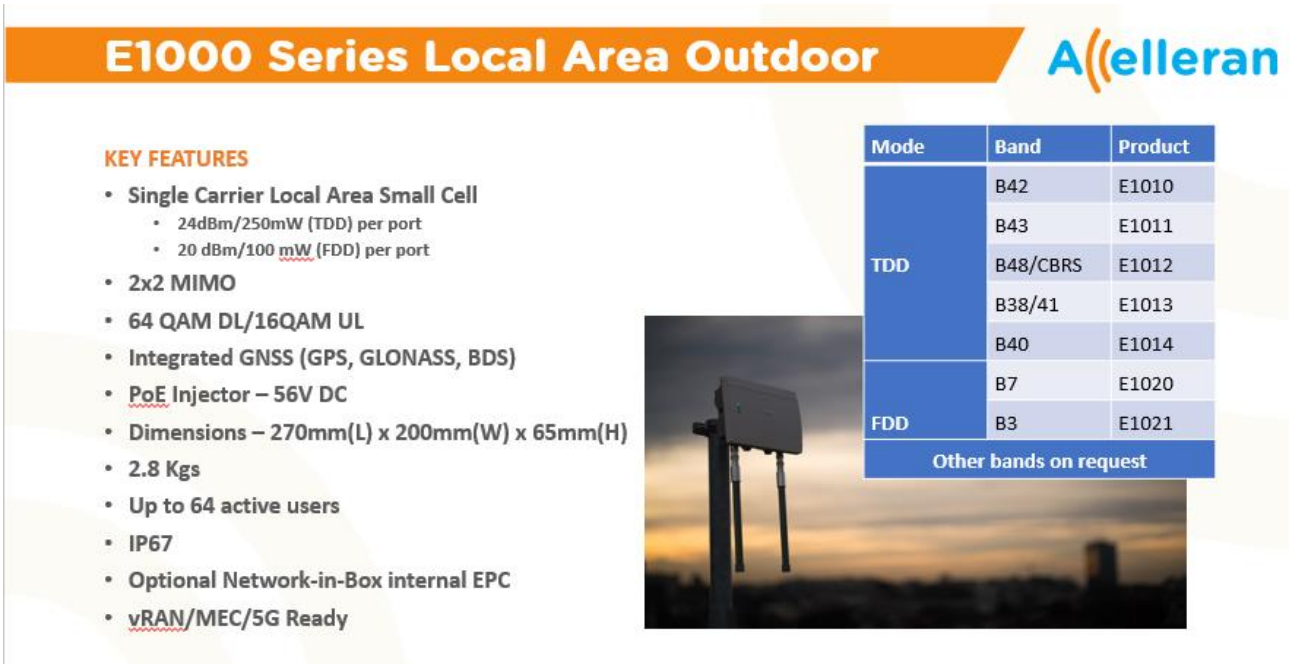

Figure 23 - Accelleran E1000 Series Local Area Small Cell

Figure 24 - Accelleran E1000 Series inside 5GCity Showroom Lamppost

4.6.2. Wi-Fi Radio Element

Wi-Fi (IEEE 802.11) is a common technology that is nowadays present in almost every UE (smartphones, laptops, tablets, etc.). The IEEE 802.11 specification has evolved continuously, introducing higher data rates and additional features. The 5GCity platform supports a variety of Wi-Fi solutions, including the commercial Ruckus solution deployed in Bristol and custom hardware (IEEE 802.11n + IEEE 802.11ac) deployed in Barcelona. The custom WiFi nodes are Gateworks Ventana 5410 boards, equipped with ath10k (QCA 988x chipset, 2x2 MIMO) and ath9k (WLE200NX chipset, 2x2) radio interfaces, deployed within a weatherproof casing and connected via an optical/electrical connector over fibre/Ethernet with the 5GCity infrastructure. Theoretical data rates over the air can exceed several hundreds of Mbps. However, in on-street deployments

the radio channel is heavily affected by interference from other wireless networks (unlicensed band) and the effects of the environment on the signal propagation, such as reflections. Further, the data rates that can be achieved by a UE depend on the characteristics of its hardware, e.g. the support for multiple-input multiple-output (MIMO) and the quality/type of antenna. As such, the real throughput may vary and can reach up to 100-200 Mbps, providing sufficient performance for the 5GCity use cases.

Independent from the actual hardware used in the 5GCity infrastructure, the only hard requirement for a Wi-Fi solution to be integrated with the platform is the support for its remote configuration via the NETCONF-based software used by the SDN controller. In the 5GCity deployment, this is achieved either with a local client that translates the NETCONF calls to local configuration calls (Barcelona) or with a proxy-like entity as it is used for the Ruckus solution in Bristol.

5. Interfaces and Workflow

5.1. Interfaces

5.1.1. SDK Interfaces

As initially specified in Deliverable D2.3 [3], the 5GCity SDK exposes a Northbound Interface (NBI) towards the 5GCity dashboard and consumes services from the 5G Apps and Services Catalogue at its Southbound Interface (SBI). The former is used to perform Create-Read-Update-Delete (CRUD) operations on Services and Functions, based on the high-level, abstract information model, while the latter is used for on-boarding and querying VNF Packages and Network Service Descriptors. Both the interfaces are based on REST APIs, implemented through the HTTP protocol. In the following, we provide the details of the HTTP messages exchanged at the NBI and SBI of the SDK toolkit.

5.1.1.1. Northbound interface (SDK/GUI)

The 5GCity SDK NBI allows communication between the GUI and the internal components of the SDK Toolkit.

Compared to the previous deliverables, all the endpoints have been renamed in order to unify the exposed interface.

The interface exposed by the SDK implements the operations described in Table 7, Table 8 and Table 9 for the three different types of resources (service descriptors, services and functions) managed by SDK. The 5GCity SDK NBI is made available also via SWAGGER documentation as reported at the end of this section.

Resource: /sdk/service_descriptor/

Table 7 - Interface description for the service descriptor resource

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		Authentication Data	Retrieve all service descriptors present in local catalogue	All data and metadata of the retrieved service descriptors
GET	{serviceDescriptorId}	Authentication Data	Retrieve the service descriptor identified by serviceDescriptorId	All data and metadata of the retrieved service descriptor
DELETE	{serviceDescriptorId}	Authentication Data	Deletion of the service descriptor identified by serviceDescriptorId	Deletion of the service descriptor from the local catalogue
GET	{serviceDescriptorId}/nsd	Authentication Data	Retrieve the NSD from a service descriptor identified by serviceDescriptorId	All data and metadata of the retrieved Network Service Descriptor
POST	{serviceDescriptorId}/publish	Authentication Data	Publish the service to the Public Catalogue	The service is published as Network Service Descriptor to the Public Catalogue
POST	{serviceDescriptorId}/unpublish	Authentication Data	Unpublish the service from the Public Catalogue	The service is unpublished from the Public Catalogue

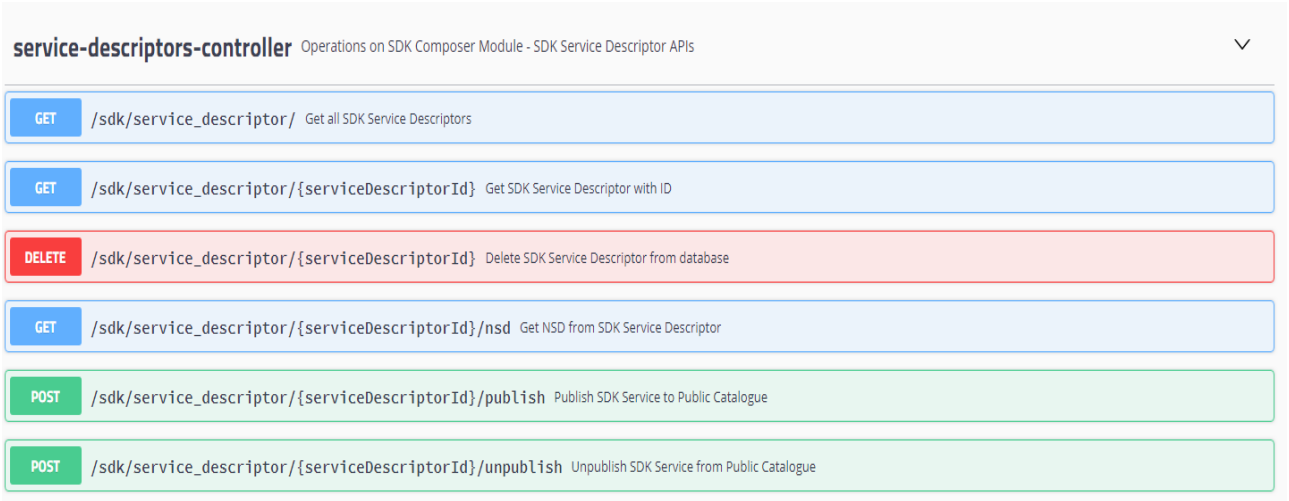

Figure 25 – SWAGGER documentation for service descriptor resource

Resource: /sdk/services/

Table 8 - Interface description for the service resource

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		Authentication Data	Retrieve all service resources present in the local catalogue	All data and metadata of the retrieved services
POST		Authentication Data, SDKService	Creation of service element	Creation of the service in the local catalogue, in simplified model
PUT		Authentication Data, SDKService	Update of service element	Update of the service in the local catalogue, in simplified model
GET	{serviceId}	Authentication Data	Retrieve service resource identified by serviceId	All data and metadata of the retrieved service
DELETE	{serviceId}	Authentication Data	Deletion of service element identified by serviceId	Deletion of the service from the local catalogue
POST	{serviceId}/create_descriptor	Authentication Data, Parameters list	Creation of service descriptor for the service identified by serviceId	Creation of the service descriptor in the local catalogue
GET	{serviceId}/monitoring_params	Authentication Data	Retrieve of Monitoring parameters from an existent service identified by serviceId	List of monitoring parameters related to the service
PUT	{serviceId}/monitoring_params	Authentication Data, MonitoringParameters list	Update of the monitoring parameters from the service identified by serviceId	Service updated w.r.t the modify request on monitoring parameters
DELETE	{serviceId}/monitoring_params/{	Authentication Data	Deletion of the monitoring parameter identified by	Service updated w.r.t the deletion request on monitoring parameters

	monitoringParameterId}		monitoringParameterId	
POST	{serviceId}/publish	Authentication Data	Publish the service to the Public Catalogue	The service is published as Network Service Descriptor to the Public Catalogue
POST	{serviceId}/unpublish	Authentication Data	Unpublish the service from the Public Catalogue	The service is unpublished from the Public Catalogue

Figure 26 - SWAGGER documentation for service resource

Resource: /sdk/functions/

Table 9 - Interface description for the function resource

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		Authentication Data	Retrieve all functions present in local catalogue	All data and metadata of the retrieved functions
POST		Authentication Data, SDKFunction	Creation of function element	Creation of the function in the local catalogue, in simplified model
PUT		Authentication Data, SDKFunction	Update of function element	Update of the function in the local catalogue, in simplified model
GET	{functionId}	Authentication Data	retrieve the function identified by functionId	All data and metadata of the retrieved function
DELETE	{functionId}	Authentication Data	Deletion of function element identified by functionId	Deletion of the function from the local catalogue

GET	{functionId} /monitoring _params	Authentication Data	Retrieve of Monitoring parameters from an existent function identified by functionId	List of monitoring parameters related to the function
PUT	{functionId} /monitoring _params	Authentication Data, MonitoringPar ameters list	Update of the monitoring parameters from the function identified by functionId	Function updated w.r.t the modify request on monitoring parameters
DELETE	{functionId} /monitoring _params/{ monitoring ParameterI d}	Authentication Data	Deletion of the monitoring parameter identified by monitoringParameterI d	Function updated w.r.t the deletion request on monitoring parameters
GET	{functionId} /vnfd	Authentication Data	Retrieve the VNFD from a function identified by functionId	All data and metadata of the retrieved Virtual Network Function Descriptor
POST	{functionId} /publish	Authentication Data	Publish the function to the Public Catalogue	The function is published as Virtual Network Function Descriptor to the Public Catalogue
POST	{functionId} /unpublish	Authentication Data	Unpublish the function from the Public Catalogue	The function is unpublished from the Public Catalogue

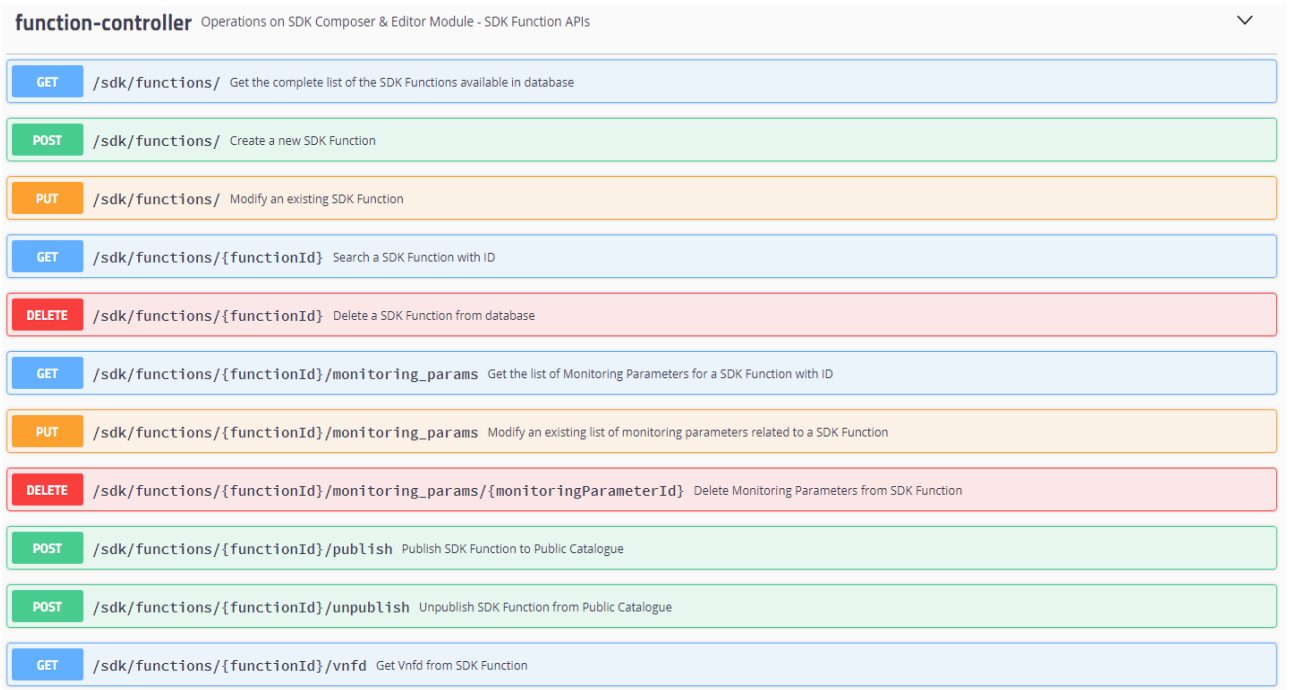

Figure 27 - SWAGGER documentation for function resource

5.1.1.2. Southbound interface (SDK/5G Apps and Services Catalogue)

5GCity SDK southbound interface allows communication between SDK Toolkit and the Public catalogue platform. The SDK acts as a client and can perform the following main operations and described in the following:

- Fetch, on-board and delete Network Service Descriptors from/to the 5G Apps and Services Catalogue (see Table 10)
- Fetch, on-board and delete VNF packages from/to the 5G Apps and Services Catalogue (see **Table 11**)

Resource: /nsd/v1/ns_descriptors

Table 10 - Interface to the Public Catalogue for Network Service Descriptors

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		Authentication Data	Retrieve all NSDs present in the public catalogue	All retrieved NSDs
GET	{nsdInfoId}	Authentication Data	Retrieve specific NSD identified by nsdInfoId	NSD
POST	{nsdInfoId}	Authentication Data, NSD package	Creation of a new NSD into the Public Catalogue	New NSD created
DELETE	{nsdInfoId}	Authentication Data	Deletion of NSD identified by nsdInfoId	Deletion of a specific NSD from Public Catalogue

Resource: /vnfpkgm/v1/vnf_packages

Table 11 - Interface to the Public Catalogue for VNF Packages

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		Authentication Data	Retrieve all VNF packages present in the public catalogue	All retrieved VNF packages
GET	{vnfPkgId}	Authentication Data	Retrieve specific VNF package, identified by the nsdInfoId parameter	VNF Package
POST	{vnfPkgId}	Authentication Data, VNF Package	Creation of a new VNF Package into the Public Catalogue	New VNF Package created
DELETE	{vnfPkgId}	Authentication Data	Deletion of VNF Package identified by vnfPkgId	Deletion of a specific VNF Package from Public Catalogue

This interface is modelled according to the REST APIs defined in ETSI GS NFV-SOL 005 [8], which in turn translates into protocol-specific messages the abstract primitives specified in ETSI GS NFV-IFA 013 [9].

5.1.2. 5GCity Platform interface updates

Deliverable D4.1[12], provided a high-level description of the interfaces between platform components, which was planned to be re-fined and extended during the implementation phase. In the following we provide the interface descriptions of some of the core platform functionalities, which are made externally available by the Slice Manager.

Since the Slice Manager is the main "entry point" for acting upon the system, it provides a REST interface, which can be used for either performing or triggering (via delegation to other components) the following main functionalities:

1. Manage (create/read/update/delete) computes, physical NWs, and Wi-Fi APs (mainly for Infrastructure Owners via the Dashboard)
2. Manage (create/read/update/delete) chunks of the above resources (mainly for Slice Users via the Dashboard)
3. Manage (create/read/update/delete) slices as collections of the aforementioned chunks
4. Manage (create/read/update/delete) users, which are authenticated by AAA
5. Trigger the deployment of Network Services (NS) on specific slices, while performing related configuration actions (that can be useful for diverse NSs during deployment, so that they function properly).

In the following, we document the REST resources that the Slice Manager is using in order to implement the previously listed functionalities.

Resource: /compute

Table 12 - Interface for managing compute resources

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		-	Retrieve all compute resources registered in the infrastructure	Data about capabilities of all compute resources e.g., CPU, RAM
POST		Data about capabilities of the compute node e.g., CPU, RAM	Register a compute resource in the infrastructure	Success or error
DELETE	{compute_id}	-	Delete a compute resource from the infrastructure	Success or error
GET	{compute_id}	-	Retrieve data of a compute resource in the infrastructure	Data about capabilities of the compute resource e.g., CPU, RAM

Resource: /physical_network

Table 13 - Interface for managing physical network resources

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		-	Retrieve all physical networks registered in the infrastructure	Data about features of all physical networks, e.g, quotas, VLAN tags

POST		Data about features of the physical network, e.g, quotas, VLAN tags	Register a physical network in the infrastructure	Success or error
DELETE	{physical_network_id}	-	Delete a physical network from the infrastructure	Success or error
GET	{physical_network_id}	-	Retrieve data of a physical network in the infrastructure	Data about features of the physical network, e.g, quotas, VLAN tags

Resource: /ran_infrastructure

Table 14 - Interface for managing RAN infrastructure resources

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		-	Retrieve all RAN infrastructure controllers registered in the infrastructure	Data about features of all RAN controllers, e.g, location, url
POST		Data about features of the RAN controller, e.g, location, url	Register a RAN controller in the infrastructure	Success or error
DELETE	{ran_infrastructure_id}	-	Delete a RAN controller from the infrastructure	Success or error
GET	{ran_infrastructure_id}	-	Retrieve data of a RAN infrastructure controller registered in the infrastructure	Data about features of the RAN controller, e.g, location, url

Resource: /openstack_project

Table 15 - Interface for managing compute chunk resources

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		-	Retrieve all compute chunks registered in the infrastructure	Data about features of all compute chunks, e.g, reserved RAM
POST		Data about features of the compute chunk, e.g, reserved RAM	Register a compute chunk in the infrastructure	Success or error
DELETE	{openstack_project_id}	-	Delete a compute chunk from the infrastructure	Success or error

GET	{openstack_project_id}	-	Retrieve data of a compute chunk registered in the infrastructure	Data about features of the compute chunk, e.g, reserved RAM

Resource: /openstack_vlan

Table 16 - Interface for managing physical network chunk resources

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		-	Retrieve all network chunks registered in the infrastructure	Data about features of all network chunks, e.g, tag
POST		Data about features of the network chunk, e.g, tag	Register a network chunk in the infrastructure	Success or error
DELETE	{openstack_vlan_id}	-	Delete a network chunk from the infrastructure	Success or error
GET	{openstack_vlan_id}	-	Retrieve data of a network chunk registered in the infrastructure	Data about features of the network chunk, e.g, tag

Resource: /ran_infrastructure/{ran_infrastructure_id}/chunkete_chunk

Table 17 - Interface for managing access network chunk resources

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		-	Retrieve all access network chunks registered in the infrastructure	Data about features of all access network chunks, e.g, access interface names, quotas
POST		Data about features of the access network chunk, e.g, access interface names	Register an access network chunk in the infrastructure	Success or error
DELETE	{chunkete_chunk_id}	-	Delete an access network chunk from the infrastructure	Success or error
GET	{chunkete_chunk_id}	-	Retrieve data of an access network chunk registered in the infrastructure	Data about features of the access network chunk, e.g, access interface names, quotas

Resource: /slic3

Table 18 - Interface for managing slices

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		-	Retrieve all slices registered in the infrastructure	Data about features of all slices, e.g, included chunks, slice name, owner
POST		Data about features of the slice, e.g, included chunks, slice name, owner	Register a slice in the infrastructure	Success or error
DELETE	{slic3_id}	-	Delete a slice from the infrastructure	Success or error
GET	{slic3_id}	-	Retrieve data of a slice registered in the infrastructure	Data about features of the slice, e.g, included chunks, slice name, owner

Resource: /user

Table 19 - Interface for managing users

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		-	Retrieve all users registered in the system	Data about features of all users, e.g, name, e-mail
POST		Data about features of the user, e.g, name, e-mail	Register a user in the system	Success or error
DELETE	{user_id}	-	Delete a user from the system	Success or error
GET	{user_id}	-	Retrieve data of a user registered in the system	Data about features of the user, e.g, name, e-mail

Resource: /network_service

Table 20 - Interface for managing network services

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		-	Retrieve all network services registered in the system	Data about features of all network services, e.g, descriptor id
POST		Data about features of the network service, e.g, descriptor id	Register a network service in the system	Success or error

DELETE	{network_service_id}	-	Delete a network service from the system	Success or error
GET	{network_service_id}	-	Retrieve data of a network service registered in the system	Data about features of the network service, e.g, descriptor id

Resource: /network_service_instance

Table 21 - Interface for managing network services

Operation Type	Resource URI	Input Parameters	Description	Output Parameters
GET		-	Retrieve all network service instances running in the system	Data about features of all network service instances, e.g, slice id, used ip addresses and ports
POST		Data about features of the network service instance, e.g, slice id, used ip addresses and ports	Register a network service instance in the system	Success or error
DELETE	{network_service instance_id}	-	Delete a network service instance from the system	Success or error
GET	{network_service instance_id}	-	Retrieve data of a network service instance running in the system	Data about features of the network service instance, e.g, slice id, used ip addresses and ports

Since the development process is ongoing, extensions and updates of the API are constantly taking place. The internals of the REST resources are documented in all detail using the SWAGGER[10] technology. Two screenshots of the SWAGGER documentation, which show part of the high-level REST structure and details of a selected REST resource, respectively, are pasted in Figure 28 and Figure 29:

[10] Please see: https://editor.swagger.io/

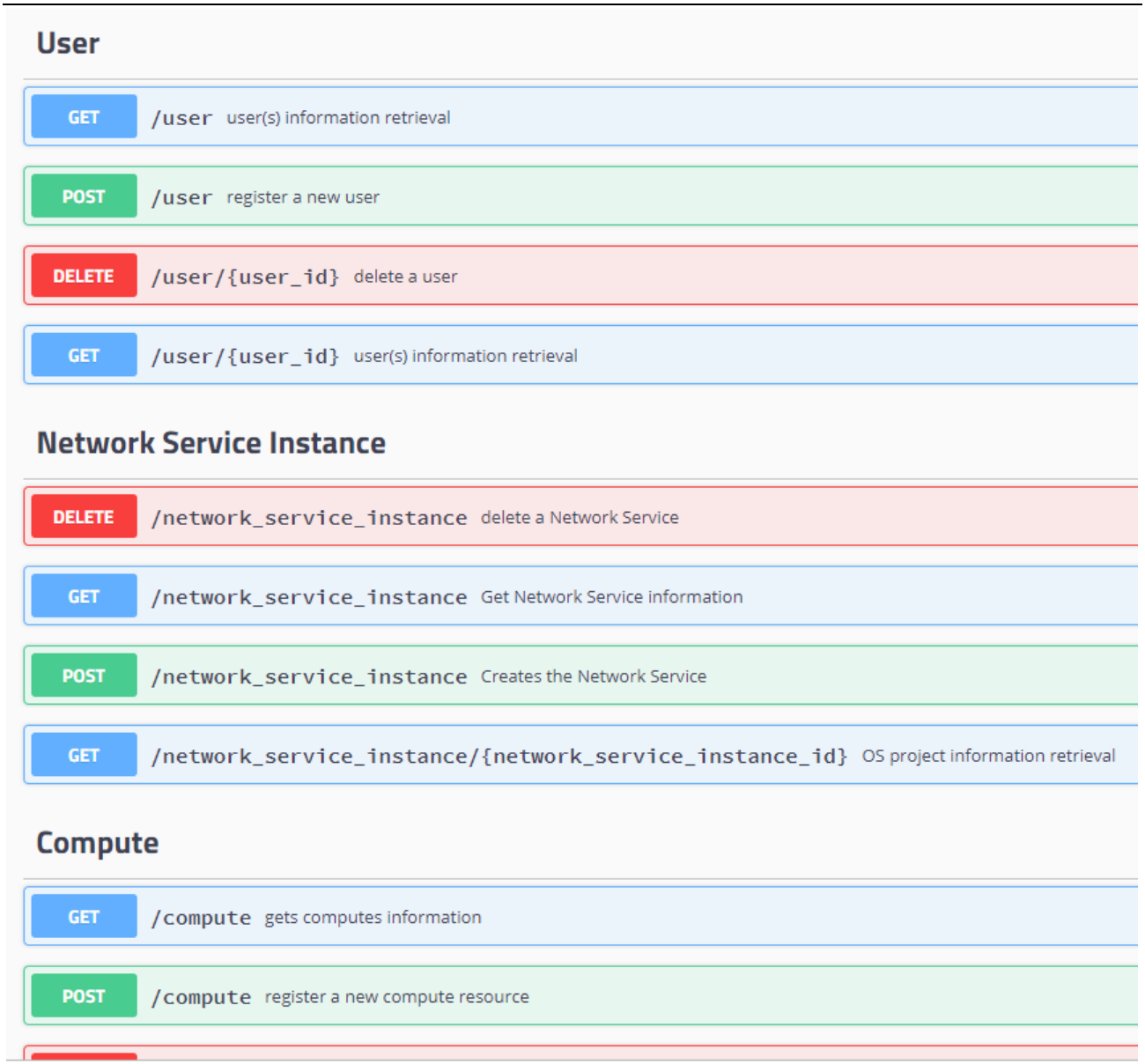

Figure 28 - Snapshot of Slice Manager REST documentation (some high-level resources)

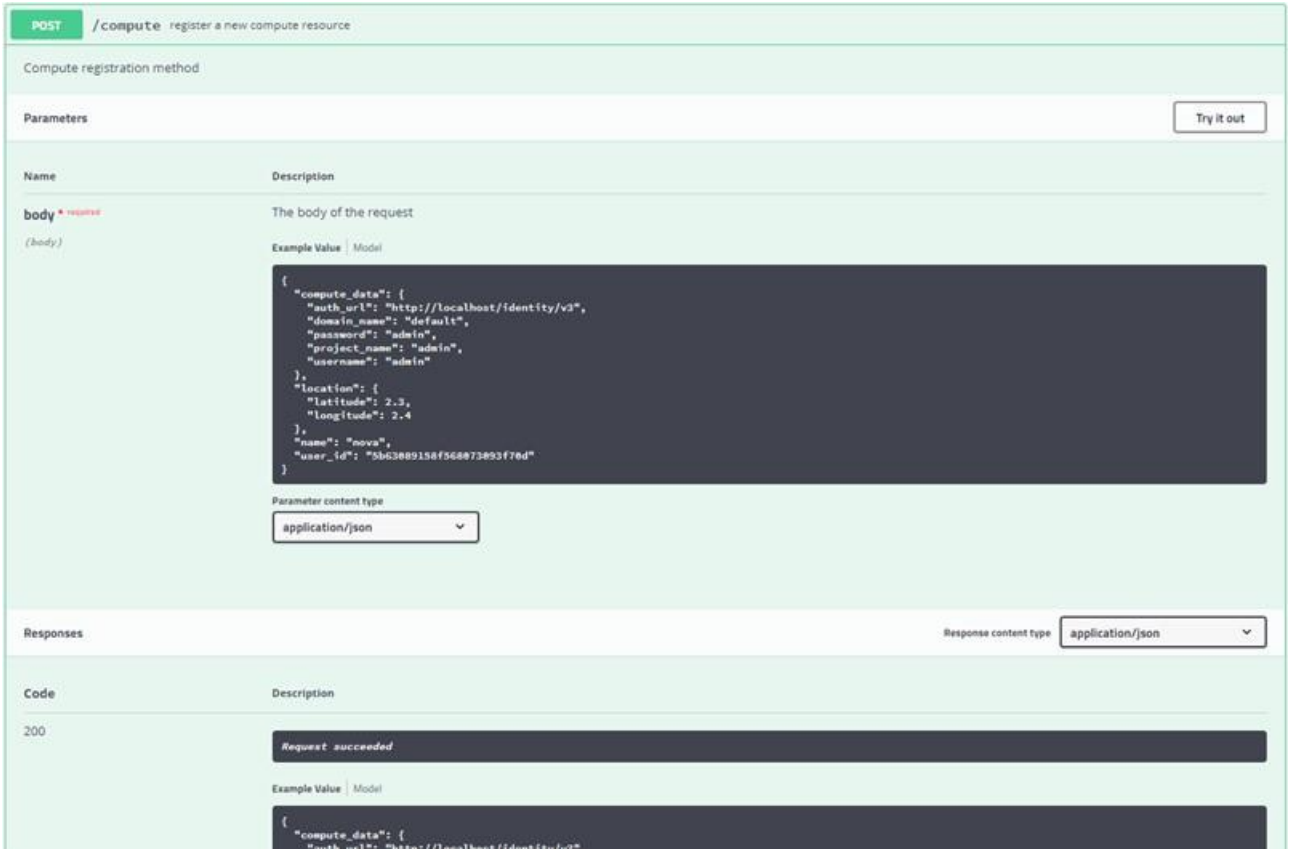

Figure 29 - Snapshot of Slice Manager REST documentation (managing a compute resource)

5.2. Workflows

The Neutral Host platform enables an ICT Infrastructure owner (IO) to logically segment its infrastructure into slices and lease them to different users over the same physical infrastructure. That is, each slice user will have access to their part of the infrastructure resources (where infrastructure may include compute, storage and network resources) which are logically bundled into a slice. The slice user will have the flexibility to choose a network service from a catalogue and deploy, run/operate and manage the network service in the slice assigned to it.

The 5GCity project addresses all the necessary pieces of the puzzle for the neutral hosting ecosystem. Those include the SDK for network service development, as well as the 5GCity platform for converting an IO into a neutral host, allowing them to perform efficient infrastructure slicing and service provisioning.

This section describes high-level 5GCity workflows from network service creation to its deployment and operation within a slice in a multi-tenant SDN/NFV-based Cloud-to-edge infrastructure.

5.2.1. Slice User Registration

Slice user registration is a mandatory step that allows a slice user to request an infrastructure slice from the neutral host and later on provision services within this slice.

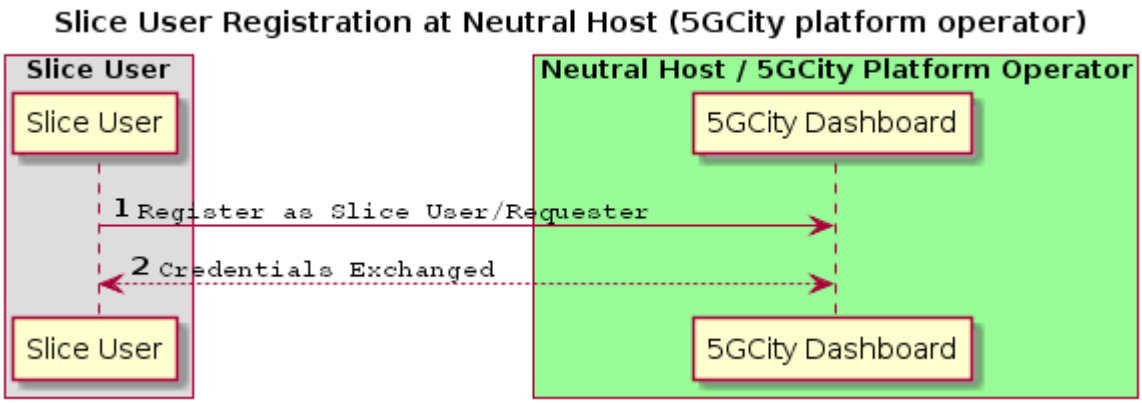

Figure 30 -Slice User Registration at Neutral Host (5GCity platform operator)

Similar registration is required for service developers as well.

5.2.1. Service creation

The result of Service Design in SDK (composition and wiring of functions) and subsequent publishing of generated descriptors into the 5G Service & App Catalogue and the NFV Orchestrator is depicted in Figure 31. The usage of the main SDK NBI methods is depicted in the interactions among modules shown in the diagram.

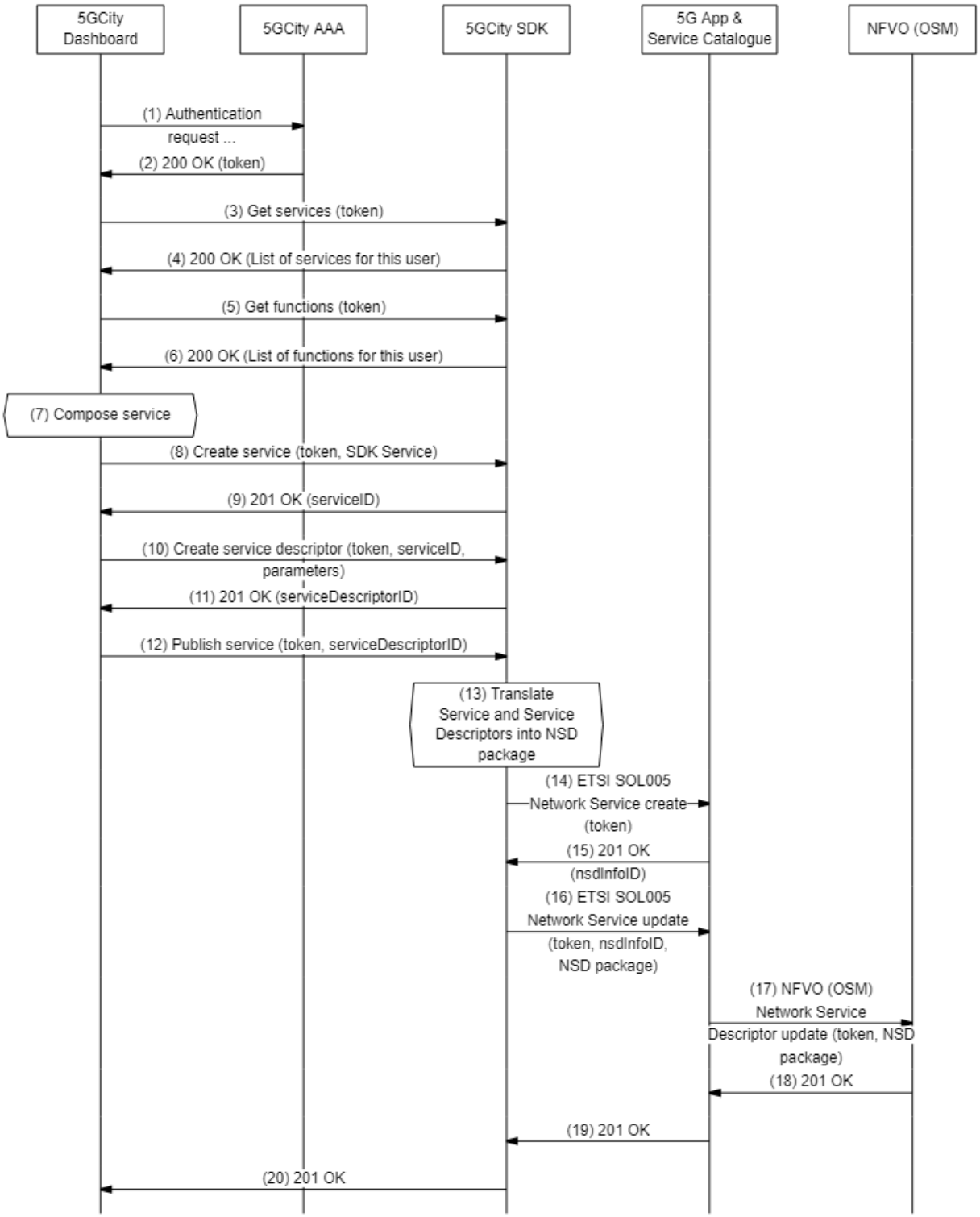

Figure 31 -Service Design workflow

5.2.2. Slice creation

Slice creation is performed whenever a Slice User requests some parts (also called “chunks”) of certain infrastructure resources managed by the neutral host through the Dashboard.

As shown in **Figure 32** this involves the triggering of different underlying controllers for provisioning the chunks in the different parts of the network, as well as a compilation of all these heterogeneous chunks into an end-to-end slice.

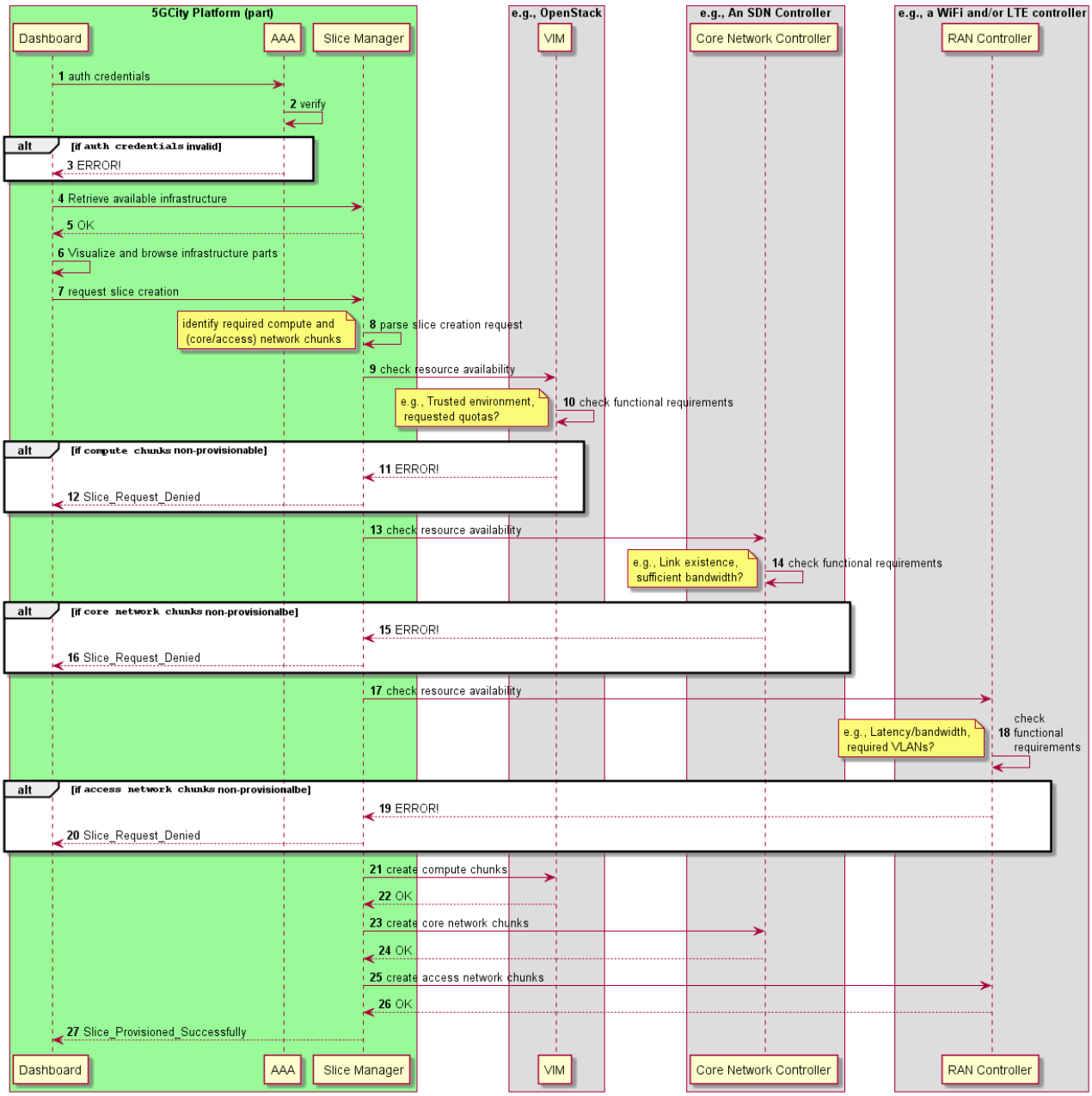

Figure 32 - Slice creation workflow with the 5GCity neutral host platform

5.2.3. Service Provisioning

Service provisioning (or "instantiation") corresponds with the deployment of an instance of a previously created network/vertical service upon a previously created and commissioned slice.

As shown in Figure 33, this involves the triggering of various components, including a Multi-tier Orchestrator which makes sure that the MEC platform is also configured appropriately if the service that is being deployed

is actually a MEC application. The internals of handling MEC applications (i.e., the internals of steps 14 and 15 in **Figure 33**) are shown separately in **Figure 34**.

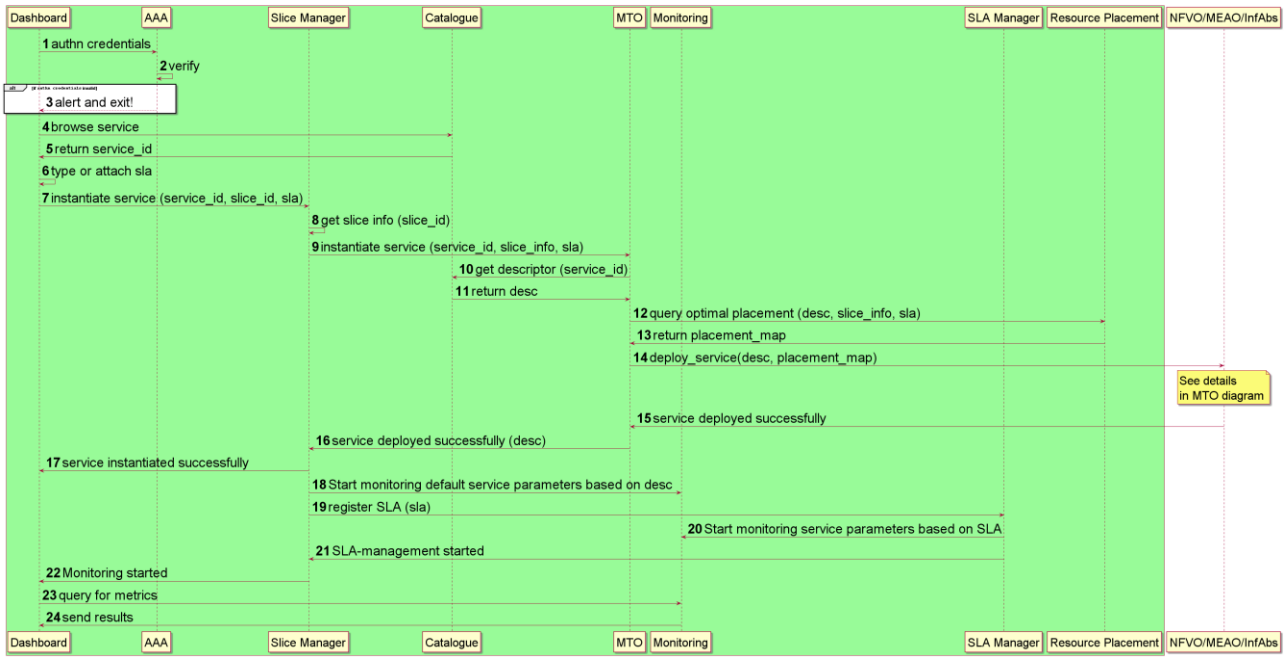

Figure 33 - Service instantiation upon a previously commissioned slice

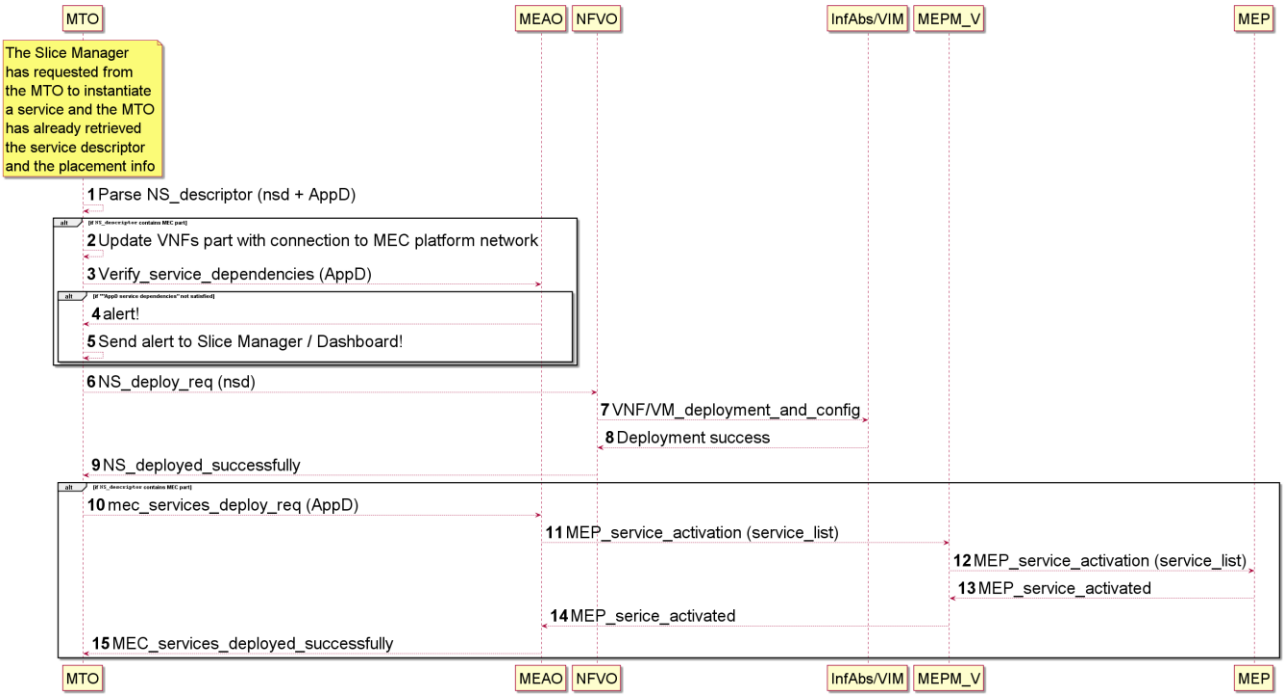

Figure 34 - Handling MEC applications as part of the service provisioning workflow

5.2.4. SLA-based Monitoring

SLAs are registered in the system, including information such as thresholds for specific monitored parameters of the related services.

The enforcement of these SLAs is based on the high-level workflow of Figure 35.

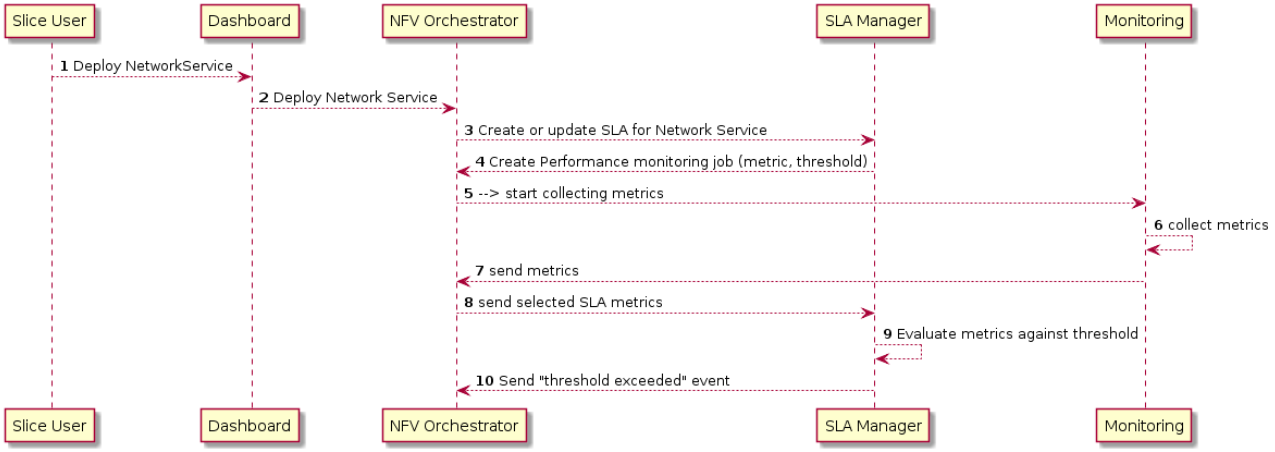

Figure 35. SLA-based monitoring workflow

6. Conclusion

In this deliverable, we presented the final iteration of design and update of the 5GCity architecture. 5GCity architecture is built upon the pillars of SDN, network functions virtualisation, MEC and network slicing over the three-tier domains. The main objective was to design an architecture capable to deploy and to demonstrate, in operational conditions, a distributed cloud and radio platform for municipalities and infrastructure owners acting as 5G Neutral Hosts.

The document presents the overall architecture targeting the neutral host model requirements and to enable the deployment of the 5GCity use cases in Barcelona, Bristol and Lucca. The 5GCity platform, including the Dashboard, the Orchestration & Control layer and the SDK, manage the interaction between the 5GCity infrastructure and the different roles envisaged in the business model. In the context of the NFVI, as innovation aspects, 5GCity architecture integrates two types of radio elements: Small Cells and Wi-Fi access points. The concept of resource slicing is extended up to the radio elements, enabling the inclusion of wireless connectivity in the slice. As the radio parts are managed by SDN, they can be integrated dynamically with other virtualized resources, such as VNFs and the wired backhaul, so an end-to-end slice can be generated. Furthermore, in the last section of the deliverable, we described the high-level 5GCity workflows from network service creation to its deployment and operation in a multi-tenant SDN/NFV based edge infrastructure.

The designs of 5GCity architecture at different layers, reported in this document, provided the blue-print for the implementation and developments achieved in WP3 and WP4, and consequently deployments and validations carried out in WP5.

7. References

[1] 5GCity Project: Deliverable D2.1: “5GCity System Requirements and Use Cases”
[2] 5GCity Project: Deliverable D2.2: “5GCity Architecture & Interfaces Definition”
[3] 5GCity Project: Deliverable D2.3: “5GCity Architecture, and Interfaces Update”
[4] https://github.com/nextworks-it/5g-catalogue
[5] https://github.com/nextworks-it/5g-catalogue/wiki
[6] ETSI GS NFV-SOL 004, v2.5.1, “Network Function Virtualization (NFV) Release 2; Protocols and Data Models; VNF Package Specification”, September 2018
[7] Draft ETSI GS NFV-SOL 001, v0.13.0, “Network Function Virtualization (NFV) Release 2; Protocols and Data Models; NFV descriptors based on TOSCA specification”, work in progress, November 2018
[8] ETSI, Network Functions Virtualization (NFV) Release 2; Protocols and Data Models; RESTful protocols specification for the Os-Ma-nfvo Reference Point, ETSI GS NFV-SOL 005 V2.4.1 (2018-02)
[9] ETSI, Network Functions Virtualisation (NFV); Management and Orchestration; Os-Ma-Nfvo reference point - Interface and Information Model Specification, ETSI GS NFV-IFA 013 V2.1.1 (2016-10)
[10] https://prometheus.io/
[11] https://grafana.com/
[12] 5GCity Project: Deliverable D4.1: “Orchestrator Design, Service Programming, and Machine-learning Models”
[13] 5GCity Project Deliverable D3.1: “5GCity Edge Virtualization Infrastructure Design”, H2020 5GCity project, May 2018
[14] 5GCity Project Deliverable D3.2: “5GCity Virtualization Infrastructure Interim Release”, H2020 5GCity project, May 2019
[15] ETSI, Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV, ETSI GS NFV 003
[16] T. Sechkova, M. Paolino and D. Raho, “Virtualized Infrastructure Managers for edge computing: OpenVIM and OpenStack comparison”, IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (IEEE BMSB 2018), Valencia, Spain, June 2018

Abbreviations and Definitions

7.1. Abbreviations

3GPP	3rd Generation Partnership Project
5G-PPP	5G Infrastructure Public Private Partnership
AR	Augmented Reality
BBU	Baseband Unit
CAPEX	Capital Expenditure
CCAM	Cooperative Connected and Automated Mobility
CDVS	Compact Descriptor for Visual Search
CP	Control Plane
CPE	Customer Premise Equipment
C-RAN	Cloud-RAN
CRUD	Create, Read, Update, Delete
DC	Data Center
DDS	Data Distribution Service for Real Time Systems
DOF	Degree of Freedom
DR	Designated Routers
eMBB	Enhanced Mobile Ultra Broadband
eNB	Evolved Node B
E2E	End to End
EPC	Evolved Packet Core
FCAPS	Fault, Configuration, Accounting, Performance and Security
FDD	Frequency Division Duplex
FFT	Fast Fourier Transform
FIB	Forwarding Information Base
FoV	Field of View
FRR	Fast Reroute
GPS	Global Positioning System
H24	24 Hours a day operation
HD-SDI	High Definition Serial Digital Interface
HDMI	High Definition Multimedia Interface
HR	High Resolution
IaaS	Infrastructure as a Service
ICT	Information Communication Technology
IoT	Internet of Things
IP	Internet Protocol
ITS	Intelligent Transportation System
ITU-T	International Telecommunication Union – Telecommunication Standardization Bureau
LCM	Life Cycle Management
LTE	Long Term Evolution
LTE-A	Long Term Evolution Advanced
LPWA	Low Power, Wide Area (network)
mMTC	Massive Machine Type Communication
MAC	Medium Access Control
M2M	Machine to Machine
M&E	Media & Entertainment

MEC	Multi access Edge Computing (formerly Mobile Edge Computing)
MEILE	Media Engagement In Live Events
MIMO	Multiple-Input Multiple-Output
MR	Mixed Reality
MNO	Mobile Network Operator
MOCN	Multi Operator Core Network
MORAN	Multi-Operator Radio Access Network
MPLS	Multiprotocol Label Switching
MVNO	Mobile Virtual Network Operator
MVPN	Multicast VPN
NATS	Open Source Messaging Service
NGMN	Next Generation Mobile Networks
NFV	Network Function Virtualization
NFVI	Network Function Virtualization Infrastructure
NFVO	Network Function Virtualization Orchestration
NH	Neutral Host
NS	Network Service
OPEX	Operative Expense
OSPF	Open Shortest Path First
OTT	Over-The-Top Player
PaaS	Platform as a Service
PE	Provider Edge
PW	PseudoWire
PoP	Point of Presence
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
RAT	Radio Access Technology
RBAC	Role Base Access Control
REDIS	Remote Dictionary Service
RF	Radio Frequency
RLC	Radio Link Control
RO	Resource Orchestrator
RR	Route Reflector
RRH	Remote Radio Head
RU	Radio Unit
SaaS	Software as a Service
SAFI	Subsequent Address Family Identifiers
SD-SDI	Standard Definition Serial digital interface
SDK	Software Development Kit
SDN	Software Defined Network
SLA	Service Level Agreement
SME	Small Medium Enterprise
TAG	Text and Graphics
TDD	Time Division Duplex
TEE	Trusted Execution Environment
UVDIS	UHD Video Distribution Immersive Services
uRLLC/uMTC	Ultra reliable communication/Low latency Communication
UHD	Ultra High Definition
UHDTV	Ultra High Definition Television
UP	User Plane

URI	Uniform Resource Identifier
vBBU	virtual Baseband Unit
V2I	Vehicle-to-Infrastructure
V2N	Vehicle-to-Network
V2P	Vehicle-to-Pedestrian
VC	Virtual Circuit
VFI	Virtual Forwarding Instance
VPN	Virtual Private Network
V-RAN	Virtual Radio Access Network
VRF	Virtual Routing and Forwarding
VIM	Virtual Infrastructure Manager
VM	Virtual Machine
VoD	Video on Demand
VR	Virtual Reality
VNF	Virtual Network Function
VNFM	Virtual Network Function Manager

<END OF DOCUMENT>