D2.1 Pilot definition and Requirements

<table>
<thead>
<tr>
<th>Project Acronym</th>
<th>5GTANGO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Title</td>
<td>5G Development and Validation Platform for Global Industry-Specific Network Services and Apps</td>
</tr>
<tr>
<td>Project Number</td>
<td>761493 (co-funded by the European Commission through Horizon 2020)</td>
</tr>
<tr>
<td>Instrument</td>
<td>Collaborative Innovation Action</td>
</tr>
<tr>
<td>Start Date</td>
<td>01/06/2017</td>
</tr>
<tr>
<td>Duration</td>
<td>30 months</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Deliverable</th>
<th>D2.1 Pilot definition and Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workpackage</td>
<td>WP 2</td>
</tr>
<tr>
<td>Due Date</td>
<td>M3</td>
</tr>
<tr>
<td>Submission Date</td>
<td>15/09/2017</td>
</tr>
<tr>
<td>Version</td>
<td>1.0</td>
</tr>
<tr>
<td>Status</td>
<td>To be approved by EC</td>
</tr>
<tr>
<td>Editor</td>
<td>Raul Muñoz (CTTC), Ricard Vilalta (CTTC)</td>
</tr>
<tr>
<td>Contributors</td>
<td>Josep Martrat (ATOS), Felipe Vicens (ATOS), George Xilouris (NCSR), Theodore Zahariadis (SY), Panos Trakadas (SY), Antón Román (QUO), Yudani Riobó (QUO), Wouter Tavernier (IMEC), Peer Hasselmeyer (NEC), Michael Bredel (NEC), Michael Christopher Woods (HWIL), Peter Twamley (HWIL), Yash Shekawat (NURO), Marco Polak (NURO), Boris Irmscher (NURO), Raul Muñoz (CTTC), Ricard Vilalta (CTTC), Ricardo Martínez (CTTC), Ramon Casellas (CTTC), Stefan Schneider (UPB), Patrick-Benjamin Böök (WEID), Marcel Müller (WEID), Eleni Fotopoulou (UBI), Anastasios Zafeiropoulos (UBI), José Bonnet (ALT), Mengxuan Zhao (EGM), Dimosthenis Kyriazis (UPRC), Maria Siapera (UPRC), Konstantinos Douloudis (UPRC)</td>
</tr>
<tr>
<td>Reviewer(s)</td>
<td>Yudani Riobó (QUO), George Xilouris (NCSR), Sonia Castro (ATOS)</td>
</tr>
</tbody>
</table>

**Keywords:** challenges, pilots, storyboard, requirements,
### Deliverable Type

<table>
<thead>
<tr>
<th>R</th>
<th>Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEM</td>
<td>Demonstrator, pilot, prototype</td>
</tr>
<tr>
<td>DEC</td>
<td>Websites, patent filings, videos, etc.</td>
</tr>
<tr>
<td>OTHER</td>
<td></td>
</tr>
</tbody>
</table>

### Dissemination Level

<table>
<thead>
<tr>
<th>PU</th>
<th>Public</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO</td>
<td>Confidential, only for members of the consortium (including the Commission Services)</td>
</tr>
</tbody>
</table>

---

**Disclaimer:**

_This document has been produced in the context of the TANGO Project. The research leading to these results has received funding from the European Community’s 5G-PPP under grant agreement no 761493. All information in this document is provided “as is” and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability. For the avoidance of all doubts, the European Commission has no liability in respect of this document, which is merely representing the authors’ view._
Executive Summary:

5GTANGO project targets the following key contributions to enable flexible programmability of 5G networks; i) an NFV-enabled Service Development Kit (SDK); ii) a catalogue and verification and validation (V&V) platform with advanced validation and verification mechanisms for virtualised network functions (VNFs) / Network Services (NSs) qualification (including 3rd party contributions); iii) a modular service platform with an innovative orchestrator in order to bridge the gap between business needs and network operational management systems; and iv) methodology and tools to implement a modern development and operations (DevOps) workflow with a multi-organisational design.

This document first discusses the state of the art of dynamic service development and validation in 5G environments, including five main challenges: programming model and toolset (devs) in combined Network Function Virtualisation (NFV) / Software Define Networking (SDN), management and orchestration architecture, mapping templates vs. instances, verification and validation of VNFs/NSs, DevOps and monitoring loop towards autonomy. Then, it specifies the three pilot scenarios which address the identified 5GTANGO challenges: advanced Manufacturing, immersive media, and real-time communications. This document also outlines the storyboard on how the 5GTANGO project currently expects its developments to be used by different stakeholders and different purposes. It is structured according to three phases; i) development of NS and publishing of NS and test results in a catalogue; ii) testing/validation with a V&V platform and publishing of results in a catalogue; iii) selecting NS from a catalogue, deployment and operation of NS using the service platform. Finally, this document also defines the functional and non-functional requirements of the four main functional blocks considered in 5GTANGO in order to provide inputs to work packages 2-6 to derive functions as well as the overall architecture of the system.
Contents

List of Figures vii
List of Tables ix

1 Introduction 1

2 Challenges of dynamic service development and validation in 5G environments 3

2.1 Challenge 1: Programming model and toolset (devs) in combined NFV/SDN . . . . 3

2.1.1 Existing approach ........................................... 3

2.1.2 5GTANGO approach ......................................... 5

2.2 Challenge 2: Management and orchestration architecture ....................... 6

2.2.1 Existing approach ........................................... 6

2.2.2 5GTANGO’s approach ......................................... 8

2.3 Challenge 3: Mapping templates vs. instances .................................... 9

2.3.1 Existing approach ........................................... 9

2.3.2 5GTANGO’s approach ......................................... 10

2.4 Challenge 4: Verification and validation of VNFs/NSs ......................... 10

2.4.1 Existing Approach ........................................... 10

2.4.2 5GTANGO Approach .......................................... 11

2.5 Challenge 5: DevOps and monitoring loop towards autonomicity ........... 13

2.5.1 Existing approach ........................................... 13

2.5.2 5GTANGO’s approach ......................................... 14

3 Pilot scenarios addressing 5GTANGO challenges 15

3.1 Smart Manufacturing ............................................ 15

3.1.1 Executive summary ......................................... 15

3.1.2 Rationale ...................................................... 15

3.1.3 Detailed description ......................................... 16

3.1.4 Requirements ................................................ 20

3.2 Immersive Media Pilot ........................................... 25

3.2.1 Executive summary ......................................... 25

3.2.2 Rationale ...................................................... 25

3.2.3 Detailed description ......................................... 26

3.2.4 Storyboard .................................................... 29

3.2.5 Architecture .................................................. 30

3.2.6 Requirements ................................................ 30

3.3 Communications Suite ........................................... 32

3.3.1 Executive summary ......................................... 32

3.3.2 Rationale ...................................................... 32

3.3.3 Detailed description ......................................... 33

3.3.4 Architecture .................................................. 33

3.3.5 Storyboard .................................................... 34
3.3.6 Requirements .......................................................... 35

4 Storyboard ................................................................. 37
  4.1 Roles ................................................................. 37
  4.2 Initial state ......................................................... 38
  4.3 Create a service .................................................. 38
  4.4 Create tests ....................................................... 39
  4.5 V&V setup ......................................................... 39
  4.6 V&V tests execution ............................................. 41
  4.7 Run Service ...................................................... 42

5 5GTANGO Requirements ............................................... 45
  5.1 Functional requirements ......................................... 45
    5.1.1 Catalogue .................................................. 45
    5.1.2 V&V Platform ............................................... 49
    5.1.3 NFV-enabled service development kit (SDK) ............ 55
    5.1.4 Extended service platform to support vertical applications .... 57
  5.2 Non-Functional requirements ................................... 60
    5.2.1 Catalogue .................................................. 61
    5.2.2 V&V Platform ............................................... 62
    5.2.3 NFV-enabled software development kit .................. 68
    5.2.4 Extended service platform to support vertical applications .... 71

6 Conclusion .............................................................. 75
  6.1 Challenges ....................................................... 75
  6.2 Pilot scenarios .................................................. 76
    6.2.1 Smart manufacturing ...................................... 76
    6.2.2 Immerse media pilot ...................................... 76
    6.2.3 Communications Suite .................................... 76
  6.3 Requirements .................................................... 77
    6.3.1 Catalogue .................................................. 77
    6.3.2 V&V Platform ............................................... 77
    6.3.3 NFV-enabled SDK .......................................... 78
    6.3.4 Service Platform .......................................... 78

7 Acronyms ............................................................... 79

A Bibliography ............................................................ 82
List of Figures

2.1 ETSI’s Network Function Virtualisation (NFV) reference architectural framework. 6
2.2 T-NOVA’s Network Service Manager (NSM) architecture. 7
2.3 T-NOVA’s Virtual Network Function Manager (VNFM) architecture. 7
2.4 SONATA’s architecture. 8
2.5 V&V Lifecycle. 12

3.1 Issue 1: Offline processing and analysis of data (in the cloud/fog) with synergies and correlations between data from different machines but without the potential to optimize the current cycle. 16
3.2 Issue 2: Online processing and analysis of data (at the machine/edge) with the potential to optimize the current cycle but without the potential of synergies and correlations between data from different machines. 17
3.3 Scenario 1: Online processing and analysis of data (in the cloud/fog) with synergies and correlations between data from different machines due to real-time interaction with multiple machines. 18
3.4 Scenario 2: Online processing and analysis of data (at the machine/edge) for fast feedback without synergies and correlations between data from different machines and online processing (in the cloud/fog) with synergies and correlations between data from different machines due to real-time interaction with multiple machines. 19
3.5 Concept for a machine network architecture. 20
3.6 Motion Control in the network architecture. 22
3.7 Fast process control in the network architecture. 22
3.8 Local cloud based process control in the network architecture. 23
3.9 Local cloud based process control and fast edge based process control in the network architecture. 23
3.10 Symmetrical encryption for cloud based process control and fast edge based process control without re-encryption from edge to cloud. 24
3.11 Immersive Media Architecture. 30
3.12 Real time Communications pilot Architecture. 34

4.1 Mapping of the V&V lifecycle to the stories. 38
4.2 Communication flow for service creation. 40
4.3 Communication flow for test creation. 41
4.4 Communication flow for V&V setup. 42
4.5 Communication flow for V&V selection and test execution. 43
4.6 Communication flow for running a service. 44

5.1 V&V Requirements Overview. 50
# List of Tables

5.1 Low-level functional requirements for key interactions of the Catalogue ................. 46
5.2 Low-level functional requirements for management of metadata ............................ 46
5.3 Low-level functional requirements for provision of decision support ......................... 47
5.4 Low-level functional requirements for provision of continuous optimisation ............... 48
5.5 Low-level functional requirements for revision and version control for all stored items ... 48
5.6 Low-level functional requirements for synchronisation with other stores .................... 49
5.7 Low-level functional requirements for testing infrastructure ............................... 51
5.8 Low-level functional requirements for test management ..................................... 52
5.9 Low-level functional requirements for test execution ........................................ 53
5.10 Low-level functional requirements for test monitoring ...................................... 54
5.11 Low-level functional requirements for test results ........................................... 55
5.12 Low-level functional requirements for development of service and component tests .... 56
5.13 Low-level functional requirements for development of performance characterisation tools ........................................... 56
5.14 Low-level functional requirements for development of performance characterisation tools ........................................... 57
5.15 Low-level requirements for Support the complete DevOps lifecycle for NS/VNFs .... 58
5.16 Low-level requirements for Support the virtualization ....................................... 59
5.17 Low-level requirements for support of network slicing ........................................ 60
5.18 Low-level non-functional requirements for catalogue - scalability .......................... 61
5.19 Low-level non-functional requirements for catalogue - security ............................ 62
5.20 Low-level non-functional requirements for catalogue - availability ......................... 62
5.21 Low-level non-functional requirements for V&V platform - scalability ...................... 63
5.22 Low-level non-functional requirements for V&V platform - customisation .................. 64
5.23 Low-level non-functional requirements for V&V platform - Availability/Reliability ...... 65
5.24 Low-level non-functional requirements for V&V platform - security ....................... 66
5.25 Low-level non-functional requirements for V&V platform - system reporting ............. 67
5.26 Low-level non-functional requirements for V&V platform - Authentication and User Management ........................................... 67
5.27 Low-level non-functional requirements for V&V platform - Availability ........................ 71
5.28 Low-level non-functional requirements for V&V platform - confidentiality ................ 68
5.29 Low-level non-functional requirements for NFV-enabled SDK - scalability .............. 69
5.30 Low-level non-functional requirements for NFV-enabled SDK - customizability ........ 69
5.31 Low-level non-functional requirements for NFV-enabled SDK - security .................. 70
5.32 Low-level non-functional requirements for NFV-enabled SDK - reliability .............. 70
5.33 Low-level non-functional requirements for NFV-enabled SDK - availability & serviceability ........................................... 71
5.34 Low-level non-functional requirements for service platform - scalability ................. 71
5.35 Low-level non-functional requirements for service platform - customizability .......... 72
5.36 Low-level non-functional requirements for service platform - security ................. 72
5.37 Low-level non-functional requirements for service platform - reliability .................. 73
5.38 Low-level non-functional requirements for service platform - availability ............... 73
5.39 Low-level non-functional requirements for service platform - serviceability ............. 74
5.40 Low-level non-functional requirements for service platform - confidentiality
1 Introduction

Software Networks, a combination of Software-Defined Networking (SDN) and Network Function Virtualization (NFV), is a significant area of research and innovation. These technologies are essential to support many aspects of the anticipated functionality offered by 5G Networks. Even at this stage before large production deployments by operators, there is an increasingly large variety of SDN and NFV solutions available from a number of vendors (and also open source initiatives), and it is necessary to provide qualification methods which allow services designed by a third party to be vetted. It is also necessary to provide a flexible service platform which can support Software Networks on a range of different operational models with different maturity levels. Software networks will have to offer the possibility to share infrastructures, customize orchestrations, support multitenancy and deploy user services. Associated to network virtualization, the slicing of resources is the next step in the resource and service provision per vertical industry.

Software networks lower entry barriers of third party vendors that provide simple Virtual Network Functions (VNFs) or composed Networks Services (NSs). This implies the need to carry out a thorough testing of these services and their qualification before going for deployment on operational environments. Even more, third-party entities could also be given permission to control certain aspects of service provision over slicing via a suitable API, in order to provide tailored, vertical-specific Network Services. These may indicate end-to-end QoS/QoE and SLA aspects across network domains (end-to-end) keeping trust levels. Service providers and Over-The-Top (OTT) will be able to programme and combine functions (create service graphs) from compatible Store with qualified VNFs/NSs and deploy them across the points of presence across multiple network segments. The deployment and orchestration will be accompanied by the necessary configuration information to fulfill the vertical user application requirements.

However, several crucial building blocks are still missing to fully realise the concept of dynamic service development and validation for Industry-specific Network Services in 5G environments and to achieve the necessary improvements for telecom sector stakeholders. These missing pieces are: i) actually developing services based on NFV/SDN is cumbersome error-prone and not suitable for daily development tasks in an industry with high turnaround; ii) deciding which resources should execute which service via which combination of NFVs is still a manual and hence tedious and unreliable (error prone) endeavour; and iii) without proper validation of network services and deployment support tools, the adoption of virtualised network services will be slow and not meet today’s needed market agility.

5GTANGO puts forth the flexible programmability of 5G networks with: i) an NFV-enabled Service Development Kit (SDK); ii) a CATALOGUE and verification and validation (V&V) PLATFORM with advanced validation and verification mechanisms for qualification (including 3rd party contributions) of virtualized network functions (VNFs)/Network Services (NSs); iii) A modular SERVICE PLATFORM with an innovative ORCHESTRATOR in order to bridge the gap between business needs and network operational management systems, and; iv) methodology and tools to implement a modern DEVOPS WORKFLOW with a multi-organizational design.

This document first discusses the state of the art of dynamic service development and validation in 5G environments, including five main challenges: programming model and toolset (devs) in combined Network Function Virtualization (NFV)/Software Define Networking (SDN), management and orchestration architecture, mapping templates vs. instances, verification and validation
of VNFs/NSs, DevOps and monitoring loop towards autonomicity. For each challenge, we analyze the current approaches, highlighting its limitations and providing an insight on the 5GTANGO approach to overcome these limitations.

5GTANGO system will be demonstrated in three vertical pilots: advanced Manufacturing, immersive Media, and real time Communications. This document also specifies the pilot scenarios which address the identified 5GTANGO challenges. The description of each pilot scenario includes an executive summary, rationale, detailed description and derived requirements.

5GTANGO proposes an integrated vendor-independent platform applied to the three pilot scenarios where the outcome of the SDK (i.e., service packages), are automatically tested in the V&V platform and stored in the Catalogue for their posterior deployment with the Service Platform. This document also outlines the storyboard on how the 5GTANGO project currently expects its developments to be used by different stakeholders for different purposes. It is structured according to three phases: development of NS and publishing of NS and test results in a catalogue; testing/validation with a V&V platform, and publishing of results in a catalogue; selecting NS from a catalogue, deployment and operation of NS using the service platform.

Finally, this document also defines the functional and non-functional requirements of the four main functional blocks considered in 5GTANGO: catalogue (data store, decision support and optimization), V&V platform, NFV-enabled SDK, and extended service platform. It aims to provide inputs to WP2-6 in order to derive functions as well as the overall architecture of the system, and to assess the outcomes of work packages with respect to completeness and functional correctness.
2 Challenges of dynamic service development and validation in 5G environments

This section discusses the state of the art of dynamic service development and validation in 5G environments, highlighting the main limitations and describing the approach proposed in 5GTANGO.

2.1 Challenge 1: Programming model and toolset (devs) in combined NFV/SDN

2.1.1 Existing approach

Software-Defined Networking and Network Function Virtualisation has introduced high expectations in transforming telecom service implementation and provisioning. As a result software plays an ever-more important role, both in controlling networks and associated services, as well as for the actual execution of the service components.

A range of programming languages and APIs have emerged in the SDN context for programming individual packet forwarding and processing devices and controlling networks of such devices. A wide range of SDN programming frameworks have been proposed as documented in the survey [33]. Despite the above work, so far, there has been no convergence yet to a common north-bound network control interface in the telecom industry.

Borrowing from cloud technology, NFV aims for a rapid development, deployment and adaptive scaling of telecom services. NFV aims to implement services, such as Virtual Private Network (VPN), Content Delivery Network (CDN) or Evolved Packet Core (EPC), in software, enabling them to be deployed on demand and in a flexible manner. The key components of these services are virtualised network functions (or middleboxes), interconnected in an on-demand network involving multiple routers and/or switches. Although telecom services can involve cloud applications, what make them unique are: i) requirements on the efficiency and steering with which network packets are handled, ii) requirements related to the reliability of the resulting services, and iii) the required interaction with existing network configuration and management practices. Having carrier-grade telecom services is of utmost importance in order to support a wide range of businesses and/or dependent emergency services.

Industry efforts towards NFV programming models find their roots in the cloud programming model. Although Amazon, Google and Microsoft were the first to provide programming models for cloud services, the Organisation for the Advancement of Structured Information Standards (OASIS) created the Topology and Orchestration Specification for Cloud Applications (TOSCA) language to describe a topology of web services, their components and their relationships in a uniform and vendor-independent way. Moreover, the TOSCA standard [39] includes specifications to describe processes that create or modify services. While most of the template languages can be used to describe service and compute requirements, and are therefore candidates to describe VNFs, they lacked a detailed description of network functionality and requirements, and therefore failed to describe complete network services. To overcome these requirements, the ETSI NFV ISG extended the TOSCA programming model [28] to support VNFs, virtual links, dedicated forwarding paths
and forwarding graphs between VNFs. The SONATA project [1] built further on the programming model of the T-NOVA project [45], and extended it, amongst others, with various concepts from the TOSCA NFV profile.

At the same time, a range of open-source software projects, as well as research projects provided similar NFV extensions to existing cloud programming models. The commonly adopted open-source Virtual Infrastructure Manager (VIM) OpenStack includes a TOSCA-to-HOT (HOT: Heat Orchestration Template [42]) translator that is developed and maintained by the OpenStack community and which is increasingly supporting TOSCA NFV features. Tacker [2] is an official OpenStack project building a generic VNF Manager (VNFM) and an NFV Orchestrator (NFVO) to deploy and operate Network Services and VNFs on an NFV infrastructure platform like OpenStack. It is based on the ETSI MANO Architectural Framework [25] and provides a functional stack to orchestrate Network Services end-to-end using VNFs, heavily focusing on TOSCA NFV templates. The Open Source MANO (OSM) project [34] is heavily inspired by the ETSI NFV programming model, and provides its own associated descriptors which are internally translated to Juju charms. The ONAP project [40], which is the result of merging ECOMP and Open-O, heavily relies on the TOSCA NFV profile as well. Similar TOSCA NFV interpreters can be found in projects such as Cloudify [56] and OpenBaton [14].

Academic efforts have researched NFV programming models following a more clean-slate programming approach, thus avoiding a direct re-use of the application-focused cloud programming model. The Click Modular Router [32] project is one of the oldest frameworks for building mostly data-plane focused network functions (middleboxes). Click uses fine-grained elements that connect together to realize the desired network function. This allows significant code reuse, as the same element is used for different network functions. Click elements can also be reconfigured on-the-fly and even the entire router functionality may be remotely replaced via hot swapping, without traffic disruption. Click has inspired other modular network function management systems, such as Slick [4] and Openbox [7]. These systems focus on control plane operations, such as data plane element placement, network function scaling, and traffic steering. Efforts like ClickOS [35] focus on reducing the isolation overhead of Click-based NFs by optimizing the interaction between Xen hypervisor technology and its IO subsystem using small virtual machines with a tailored OS which can be rapidly started with low overhead. NetBricks [43] is a more recent approach focusing on the development of NFs and NFV services based on a small set of customizable network processing elements. Based on type checking and safe runtimes to provide isolation in software, NetBricks provides the same memory isolation as containers and VMs, without incurring the same performance penalties. Hyper-NF [13] is a NFV framework specifically designed for implementing chains of virtualised middleboxes. Hyper-NF eliminates redundant packet and IO operations. Given a chain of middleboxes, it uses graph search and set theory to generate a single equivalent middlebox that only uses one read and one write operation per packet.

Other approaches focus on the state involved in NFs and associated services. During failover, scaling or other migration events, the state of running instances needs to be managed and safely transferred between instances. OpenNF [19] proposes a centralised control plane for sharing information between software NF applications, in cases of NF replication and migration. However, it requires the source code of NFs to be modified. Approaches such as StateAlayzr [30] can assist developers in doing so.

One of the consequences of programming services based on software-based components, is the introduced uncertainty in the resulting performance. Whereas the performance of dedicated, and custom-engineered hardware is predicable and stable, the performance of software-based implementations might vary significantly depending on a range of conditions such as: the underlying hardware platform (e.g., cpu architecture, memory available, etc.), the traffic load and its parameters, the
way in which software components (NFs) are (de-)composed, or the used software technologies. In order to avoid that these uncertainties again translate into massive over- or under-provisioning, several research efforts have focused on the characterisation of performance of software-based functionality. The approach documented in [58] investigates a data-driven approach, which is able to build a performance model based on the behaviour of a job on (optimally selected) small samples of data. Only a few, more recent performance modelling studies focus on VNFs. Analytical models such as those documented in [48, 55] exist for selected VNFs, accounting for the low-level architecture of the infrastructure (e.g., number of CPUs, cache memory available, etc.), or for the number of particular machine-level instructions in the source code (e.g., memory copies in I/O). Architectures for more measurement-driven performance estimation approaches are proposed in [47, 11, 31]. They have in common the following subcomponents: i) the definition of a benchmarking setup for the VNF, ii) the definition of a recordable performance profile for this VNF benchmarking setup, and iii) parameters which are able to capture the variability of potential underlaying platforms. A first proof of concept of these ideas has been proposed in context of SONATA in [44]. This work proposes a profiling mechanism that is able to profile virtualised network functions and entire network service chains under different resource constraints.

2.1.2 5GTANGO approach

The NFV programming model of existing approaches has still several challenges to which 5GTANGO plans to contribute:

- there is a wide range of very alike, but syntactically different descriptor formats
- service and function configuration attributes are largely ignored
- there is limited development support and integration for handling stateful functionality and services during the events of scaling or failover
- performance aspects and monitoring attributes are largely missing in existing models.

5GTANGO will build further on the ETSI NFV programming model as introduced in SONATA, and will investigate integration and translation tools between different TOSCA variants as used in leading MANO platforms. For this purpose, 5GTANGO will further work on the descriptor translator functionality, which is one of the latest early-stage additions to the SONATA SDK. In order to streamline the process for NF developers to integrate existing network applications, container definitions or Click configurations, 5GTANGO will provide functionality to rapidly build and translate NF images, supporting an extensible number of target formats. For this purpose, the project can further extend the design of the vmbuilder component of the SONATA SDK.

Virtualising telecom services poses additional challenges with respect to their management and configuration. Although current telecom operations has moved from the old-school script-based management towards more model-based configuration techniques, involving, for example, NETCONF/YANG models, this is clearly not enough. The NFV programming model does not only need to support modelling the bare NFV services, but needs to go beyond and integrate the configuration of the resulting services. 5GTANGO aims to streamline the interaction between service function modelling and its configuration, focusing on robust model-driven deployment mechanisms building further on existing state-of-the-art management technologies such as NETCONF.

5GTANGO’s approach to handling stateful network functions and services during the events of scaling, fail-over or version update of running instances, will build further on the work introduced by OpenNF and will integrate state control mechanisms in re-usable Specific Manager-functionality as supported by the SONATA architecture.
Most predictive models focus on performance modelling and estimation of cloud applications. Although there is an overlap with NFV-related applications, NFV-applications are significantly more dependent on an accurate model for data-plane, packet-level operations compared to cloud applications. Most existing performance modelling approaches suffer from two issues: i) they require significantly manual effort (e.g. characterising the infrastructure, detecting the most costly instructions, tuning their weight, etc.), and ii) they are not generalisable towards other applications or NFs or even slightly differing context. Therefore, 5GTANGO will investigate three levels of performance characterisation frameworks: i) generalisable analytical models (usable beyond a single function), ii) performance simulation environments by modelling the NF under a range of statistical conditions, iii) measurement-based profiling mechanisms. These three levels will enable step-wise refinement of the performance profile of individual and composed VNFs. Based on the groundbreaking work in SONATA related to service and function monitoring and profiling, 5GTANGO will continue this work and integrate a streamlined performance model into the ETSI NFV programming model, which on its turn could be exploited during the orchestration phase in the Service Platform.

2.2 Challenge 2: Management and orchestration architecture

2.2.1 Existing approach

ETSI’s approach ([50], [25]) clearly separates NFV management and orchestration in two (three, if Virtual Infrastructure Managers (VIMs) are also considered) layers (fig. 2.1).

In this architectural framework, there’s a top layer (the Orchestrator) which interfaces with the externals systems, like the Operations’ Support Systems (OSSs) and commands the lower layer, made of one or more Virtual Network Function Managers (VNFMs).

T-NOVA’s [45] approach to management and orchestration closely follows ETSI’s proposals (see
The **Network Service Manager** (NSM, see fig. 2.2) is clearly separated from the VNFMs, see fig. 2.3), which allows for an independent implementation of the later, as long as it responds to the same API that was designed and implemented in the **Generic VNFM**.

T-NOVA’s NSM implements the orchestrator’s *northbound interface* (NBI) and talks to the VNFM (it’s *south bound interface*, or SBI) to have any service’s VNFs deployed in the VIMs and receive monitoring data that permits scaling, changes in configuration, etc.

T-NOVA’s VNFM implements the orchestrator’s *southbound interface* (SBI) to the VIM(s). The orchestrator has access to deploy the necessary infrastructure needed for a given service instance. Virtual Machine (VM) based monitoring data is collected at this level and translated into VNF-based monitoring data, that is then made available to the upper layer for decision support.

In any of these pictures, a **validator** element is represented: an **NSD Validator** in fig. 2.2 and a **VNF Validator** in fig. 2.3. Both levels of descriptors are **syntactically** validated before being accepted in the **Catalogues**.

SONATA’s [53] approach takes this into the next level (see fig. 2.4), by providing:

1. an SDK to help the developer build the **Network Service Descriptors** (NSD) and the **Virtual Network Function Descriptors** (see sec. 2.1);

2. a Service Platform that is able to:
Figure 2.4: SONATA’s architecture.

1. provide by default (Network) **Service** and (Virtual Network) **Function** managers that are specific to the different aspects that can be managed in a service or function, e.g., life-cycle, placement, scaling, etc. (these are equivalent of the NS Manager and VNF Manager mentioned above, but much more granular)

2. accept **Specific Service** or **Function Managers** that manage those different aspects specifically for the service or function it is attached to, instead of the generic ones

3. identify and allow/disallow developers that provide those services and functions, together with their specific managers, through a component named the **Gatekeeper**, thus imposing the needed security for such pieces of code to enter the Service Platform in production

Both the SDK and the Service Platform, through the Gatekeeper, use the same micro-service (not represented in fig. 2.4) to validate all the descriptors that are present in a **package**, upon both **generating it** (on the SDK side) and **on-boarding it** on the Service Platform side. Furthermore, not only the syntax is validated, but also **integrity** and **topology**. This validation was found to be really valuable for developers to quickly iterate and provide corrections or improvements of their service’s or functions’ descriptors, but are rather static.

### 2.2.2 5GTANGO’s approach

5GTANGO will advance one step further in terms of management and orchestration by improving and extending the currently available **validation mechanisms**. This improvement will impact both management and orchestration in (at least) the following aspects:

1. by having an independent **Verification and Validation** component, the interaction between the SDK and the Service Platform will be very enriched (and different), in ways that are still being defined

2. a consequence of that much more rich interaction is that the Service Platform will have to be opened to deploy onto a plethora of testing environments (in the SONATA approach this can already be done, but in a much more controlled and limited way: it is the Service Platform’s owner that defines and configures how many and which VIMs there are onto which services and functions can be deployed)
1. the Gatekeeper will have to deal with many more actors and roles, which are still being defined.

5GTANGO will strongly commit to address the above aspects in the anticipated orchestration and management layer.

2.3 Challenge 3: Mapping templates vs. instances

2.3.1 Existing approach

In legacy systems with network functions deployed in hardware, chaining network functions together into a service is mainly a problem of connecting functions in the right order, which is, eventually, a traditional routing problem. With the introduction of VNFs, running in software, dynamic placement at/migration to any locations with regular x86-hardware is possible. Hence, the actual location of such VNFs becomes a decision variable of a complex optimisation problem that is vastly more challenging than simply chaining legacy network functions at fixed locations. The complexity stems from the many degrees of freedom and from service-specific requirements, e.g., delay guarantees.

Apart from dynamic placement of VNFs at different physical locations, VNFs can be scaled to react to the current demand. Scaling refers both to adding/removing instances of a VNF (horizontal scaling) and to increasing/decreasing the virtual resources assigned to an instance (vertical scaling).

However, the specification model used in most ongoing network function virtualisation works is based on explicit descriptors. Such explicit descriptors are similar to the VNF Forwarding Graph (VNFFG) description defined in the ETSI Network Function Virtualisation (NFV) management and orchestration document [25], for example, in UNIFY [18], T-NOVA [45], OpenMANO [57], and OpenBaton [14] projects. Using such explicit descriptors, service developers define the exact resources required for serving the desired traffic for their services. Scaling instances in or out as well as controlling resource allocation are limited to these fixed descriptions. This lack of flexibility can easily lead to either to reduced service quality when not enough resources are allocated or to waste of resources when too many resources are allocated.

An alternative to these explicit descriptors are service templates. A template describes a service in a generic but precise way, specifying the structure of the service without fixing the number of instances per VNF or their exact absolute resource consumption. Instead, a VNF’s resource consumption can be specified, for example, as a function of the load that needs to be processed. Thus, the number of instances of each VNF and their resource consumption can be determined and adjusted dynamically and automatically according to the current load. This ongoing fine-grained adjustment is especially important as the load and other parameters continuously change during the service lifetime. Embedding services described by such templates requires joint scaling of the template according to the load and placement of the individual instances in the substrate network as well as connecting the instances with each other. Keller et al. [29] investigate this problem of joint scaling and placement in the context of distributed cloud computing. Dräxler et al. [12] refine the model and take vertical scaling into account, adjusting the allocated resources per instance according to the current load.

Complementary to the problem of embedding templates, Mehraghdam et al. [36] as well as Beck and Botero [6] study the possibility to flexibly decide the structure of a service. By optimising the order of VNFs in a service, the total resource requirements can be reduced.
2.3.2 5GTANGO’s approach

5GTANGO will exploit the benefits of flexible templates and go beyond existing approaches for scaling and placement of VNFs. To extend the possibilities of dynamic embedding of services, over-subscription of resources to a certain extend may be allowed. For example, 1100 services might be mapped to 1000 CPUs as most likely not all services will be fully utilised at the same time. In doing so, resource utilisation can be maximised and idle resources can be avoided. Clearly, there have to be mechanisms in place to control the over-subscription of resource to prevent violation of service-level agreements. 5GTANGO will consider both general network constraints and vertical-specific constraints. For example, a manufacturing scenario or a real time video communication scenario can have specific requirements and focus on different aspects (e.g., specialized hardware, delay-sensitive, limited scaling, complex service structure, etc.). Despite their relevance in practice, such vertical-specific scenarios are mostly disregarded in current work.

5GTANGO will support the optimization of different goals such as minimizing the number of used servers, maximizing the utilization of means for production, energy consumption, or service quality (e.g., in terms of maximum/average delay). Such objectives can either be optimized individually or jointly by assigning different priorities to the different goals. This enables infrastructure providers to optimize their resources based on their intended optimization goals.

2.4 Challenge 4: Verification and validation of VNFs/NSs

2.4.1 Existing Approach

The promise of NFV is to relocate network functions from dedicated hardware appliances to software on COTS HW. However, incomplete or inconsistent configuration of VNFs and forwarding graphs could cause breakdown of the supporting infrastructure. In addition, the environments (NFV/MEC Infrastructures) where the NFV Network Services are being deployed are not homogeneous. It is anticipated that except OpenStack [46] which currently is the de-facto implementation of NFVI-PoP and VIM, there will be other virtual infrastructure management and virtualisation enabler platforms that will be exploited towards NFVI PoPs. In this sense, verification and validation of NSs and VNFs is very critical for network operators in order to check that their requirements and network properties are correctly implemented in the execution infrastructures. The main building blocks of the NFV architecture are:

1. VNFs
2. NFV Infrastructure (NFVI)
3. management and Orchestration (NFV MANO)
4. network Services

In 5GTANGO the main targets for validation and verification are the first and the last respectively, however as the pluggable architecture allows for VNF and NS specific management and orchestration components, it will also tackle MANO component. As the issue of validation is critical, ETSI NFV ISG has released a number of documents approaching the need and the framework for pre-deployment validation and testing [26] as well as interoperability and portability [27]. The proposed methodology uses the notion of System Under Test (SUT) in order to categorise test procedures for each unique building block of the infrastructure. The issue of verification of NFV
Network Services is also addressed by NFV Research Group [23]. The current outcomes of the working group mainly focus on the problem statement and the challenging issues that the verification of NSs faces including:

- consistency check in distributed state since NFV states as well as SDN controllers are distributed
- live traffic verification is challenging given the high complexity of verification tasks for real problem
- languages and their semantics are many and thus it is impractical for the verification frameworks to support all of the existing languages and models
- stateful VNFs with multiple physical views should be checked differently from the non-stateful ones since their correctness relates to behaviors that depend on the previous states (packets, actions, etc), while their physical entities are also multiple

In addition to the above standardisation/specification groups, Open Source initiatives are addressing verification and validation. OPNFV Linux Foundation initiative [15] has specific projects, i.e. Doctor [16] and Yardstick [17]. Due to the objectives and focus of OPNFV, these projects mostly focus on NFVI validation and fault monitoring rather than VNF or NS, however both mention fault management issues such as break-down of the supporting infrastructure due to incomplete or inconsistent configuration of NFV services.

In the industry sector, validation frameworks for NFV environment and VNFs have already been proposed. E.g., SPIRENT’s [54] portfolio of products addresses validation by benchmarking individual VNFs and chains for functionality, performance, support for lifecycle operations to identify weak links in the service chain. HPE uses three levels of the Validation process for its validation methodology [20]. To this end, HPE publishes a catalog of VNFs provided by HPE OpenNFV partners and tested in HPE OpenNFV labs [21]. In 5GTANGO cross-comparison of similar functionalities of VNFs or reference VNFs will also be made available through the 5GTANGO catalogue.

Model-based testing is an emerging approach in software and system testing. MBT refers to “testing based on or involving models”. Models represent the System Under Test (SUT), its environment, or the test itself, which directly supports test analysis, planning, control, implementation, execution, and reporting activities. MBT formalises and automates as many testing activities as possible, increasing efficiency and effectiveness. Compared to traditional tests, MBT is able to provide solutions that manage testing complexity in a better and cost efficient way. Instead of writing a test case specification with hundreds of pages, MBT automatically generates test cases from MBT models, to be used by a test execution platform which actually executes test cases. MBT provides several benefits including management of complexity through higher levels of abstraction, communication efficiency through formalisation of activities using models, validation of requirements through early-stage models, increased interactivity through tests visualisation, automation through tools generating test cases from models, and debugging support through traces introduced in the models.

2.4.2 5GTANGO Approach

5GTANGO will provide a holistic breakthrough solution for validation and verification. The current NFV validation propositions (although addressing the validation requirements) are disengaged from the Orchestration Platform itself and are usually pre-active, operating in pre-deployment phases. In addition, they mostly focus on the NFVI validation and VNF performance. 5GTANGO will create
a tighter coupling of the validation framework with the MANO stack, adopting methodologies and mechanisms for conducting validation tests from already established solutions. 5GTANGO will allow both in-vitro validation (sandbox environment) and in-situ (in the place of deployment) by offering high-level orchestration capabilities to the developers. The 5GTANGO framework will allow the validation components at the MANO to exploit the pluggable architecture of the Service Platform. In this view validation tools will exist in the store, so that developers can exploit them to instrument validation tests. Such tests will be generated from MBT models, compiled and made executable in the store (exploiting background experience in the context of web service testing [74]). In addition, the publication of test suites using the ETSI normalised TTCN-3 language (to deliver appropriate TTCN-3 cases [75]), will be considered as an intermediate step between test generation and test execution. Using TTCN-3, a widely accepted test language by ETSI and other Fora such as 3GPP, would further increase exploitation potential of 5GTANGO results. What is more, hybrid monitoring and analysis tools will be integrated in order to achieve state-of-the-art validation focused to VNF and NS. The sandbox environments that will be deployed will be also validated in order to be used as reference validation environments.

The main actor of the V&V Platform into 5G TANGO concept is the NS developer. As already presented the NS developer is the person that develops or reuses available VNFs in order to create a meaningful NS. The NS comprises of VNFs, their lifecycle management (i.e. VNFM or modular Function specific plugins) bundled with their own service orchestrator components (or service specific management plugins), packetised along with descriptors for the deployment and
configuration of VNFs. The V&V framework that is being developed in the frame of 5GTANGO allows the bundling of specific experiments along with the service package. These experiments are automatically executed from the V&V platform in a sandboxed environment operating under the V&V platform control, so that functional and non-functional metrics are acquired leading to the verification and validation of the NS. These metrics are then stored along with the package in the Catalogue. Additionally, the 5GTANGO SDK platform will provide the capability (in terms of libraries and helper functions) to the developers to design and describe the experiments based on the actual infrastructure capabilities.

In summary the lifecycle of service validation and verification is illustrated in fig. 2.5. This lifecycle definition is only at the initial stages, drafting the current status of conversations within the members of the project’s consortium. It will be further refined in forthcoming deliverables.

2.5 Challenge 5: DevOps and monitoring loop towards autonomicity

2.5.1 Existing approach

Recently, DevOps has become a buzzword due to its ability to decrease IT operational costs while simultaneously improving source code quality and reduce time-to-market [5]. DevOps reorganises software development and operations departments in a company by implementing processes for continuous integration & delivery, service performance monitoring and automated testing. This process is well aligned with the objectives of 5GTANGO. One of the main issues to be tackled in a DevOps approach is network and service monitoring, requiring accurate and timely statistics not only on network but also on computation and storage resources involved at different aggregation levels. Currently available monitoring tools, frameworks and platforms fall short in satisfying in an efficient way requirements arising from all involved stakeholders and including dynamically generated data in large volumes and from several technologies. As a result, these proposed solutions are in many cases: (i) intrusive and heavy-handed for short-lived, lightweight cloud instances - e.g. [9], [10], [22], (ii) not able to follow the fast pace of deployment changes enforced by continuous dynamic scheduling, provisioning and auto-scaling - e.g. [38], [59], (iii) not covering deployments in both hypervisor-based and containerized manner [8], [49], and (iv) they are bounded to specific technologies - e.g. [37], [41], making them inappropriate for open environments such as the one envisioned in 5GTANGO. In this perspective, the development of a monitoring framework is not trivial in today’s complex and heterogeneous infrastructure environments, given the applications diversity that are instantiated therein. An interesting approach for these obstacles was proposed by ETSI Industry Specification Group (ISG) Autonomic network engineering for the self-managing Future Internet (AFI). ETSI AFI Generic Autonomic Network Architecture (GANA) instruments the nodes / devices / modules with Autonomic Manager Components/ Elements (called Decision-Making-Elements (DMEs) or Decision Elements (DEs)), which automates network operations by realising control loops. These control loops operate on knowledge of events and state of network resources or functions, at different abstraction levels of functionality, and regulate the resources or functions of a system/network according to the holistic goals of the network [24]. This rather innovative concept, supported by the European standardisation body, is fully aligned with SONATA’s architecture, in particular the implementation of the Function-Specific Manager (FSM) and Service-Specific Manager (SSM), complemented by the open-source SONATA monitoring framework built on top of state-of-the-art technologies.
2.5.2 5GTANGO’s approach

5GTANGO introduces these concepts as an extension of SONATA 5G’s Architecture. Its Service Platform provides a customisable plugin mechanism. Using this mechanism, the management and orchestration behaviour of the service platform for individual network functions and services can be customised through FSMs and SSMs. Within 5GTANGO, the focus will be on the design and implementation of an autonomic framework supporting autonomic management for VIM, VNFM, and VNF components. 5GTANGO will innovate in autonomicity introducing and extending a complete monitoring framework as well as Function Specific Managers (FSMs) and Service Specific Managers (SSMs) enabling autonomic management. In greater detail, the monitoring framework will close the virtuous cycle of DevOps paradigm in 5G environments providing to the developers the required visibility upon use of resources and network service operation in-situ (in the actual deployment environment). Autonomicity with respect to monitoring is defined as the main enabler for closing the loop between network service development, deployment and operation as defined in the Application Lifecycle Management and achieving enriched and advanced self-manageability of cloud environments, nodes and networks by design. 5GTANGO will capitalise and extend the (open source) monitoring framework developed in SONATA by properly enhancing it with tools to facilitate cross-layer mechanisms for supporting functionalities of the extended 5GTANGO DevOps approach, addressing different levels: (i) VNFM: 5GTANGO will exploit the flexibility offered by the autonomicity concept in order to offer reusable software defined DEs that can be exploited by the developers and modified at runtime supporting the DevOps model. (ii) VNF: 5GTANGO will support through its SDK the embedment of Layer 2-3 and Knowledge plane DEs in the VNFs in order to allow the collection of VNF specific metrics supporting the autonomic management of each VNF. (iii) VIM: 5GTANGO will deliver the Network DE enabling autonomic management over the network resource allocation (mainly within the NFVI-PoP). Thus, the ultimate goal of 5GTANGO is to design an effective monitoring framework, compliant with the ETSI NFV MANO and ETSI GS AFI specifications. Taking into account the list of functional and non-functional requirements imposed by the vertical pilots and use cases foreseen to be fulfilled within 5GTANGO, the proposed design aims to provide an interactive monitoring framework capable of offering personalised, real-time data collection and alerting to all stakeholders of an SDN/NFV-enabled service platform, i.e. service developers, service platform operators and end-users, under heterogeneous cloud-enabled computing environments.
3 Pilot scenarios addressing 5GTANGO challenges

5GTANGO system will be demonstrated in three vertical pilots: Advanced Manufacturing, Immersive Media, and Real Time Communications. This section specifies the three pilot scenarios which address the identified 5GTANGO challenges. The description of each pilot scenario is organised as follows: an executive summary, rationale, detailed description and derived requirements.

3.1 Smart Manufacturing

3.1.1 Executive summary

In the context of smart manufacturing, the given advantage is based upon the processing of operational data, machine data, and process data (O/M/P data) generated. Using this data, the process of production, the control of the intra-logistics supply chain as well as the control of components becomes possible. It builds the basis for future digital pervasive supply chains.

The allocation of computational resources and its placement impose two main challenges:

1. any kind of additional equipment or equipment with high performance at the edge of the machine increases costs and reduces sustainability due to waste of computation power. Nevertheless, being at the edge and next to the machine allows real-time feedback to the machine

2. transferring the amount of measured information from the machine to the cloud requires mechanisms that allow real-time communication and, thereby, online processing of data. Due to the lack of availability of network functions on the full path from edge over fog to the cloud, existing solutions can hardly address these challenges. In this frame 5GTANGO will address these challenges for the smart manufacturing context by providing the framework to be able to flexibly allocate resources depending on the requirements of the vertical case.

With 5GTANGO platform it will become possible to proactively access functions and services on the path that fulfil the order- and machine-dependent requirements in connectivity and services.

3.1.2 Rationale

Smart Manufacturing in current contexts suffers from the fact that data generated in machines or equipment is contextualised and processed in order to be usable, too late in the process due to the restrictions of pervasive and flexible services. Currently, no efficient online processing is possible.

3.1.2.1 Issue 1: Offline processing and analysis of data but without the ability for optimising the current cycle

As illustrated in fig. 3.1, operational, machine and process data (O/M/P data) is generated by the machinery present in the manufacturing site. This data is generated by the machine itself or with a variety of additional sensors. The generated data may be accessed directly at the machine or at
the sensor; maybe it can be accessed in an aggregated, but not reduced way at the machine, so that no relevant information is lost and, e.g., peaks are still visible.

The information is sent to somewhere in the cloud for further asynchronous processing. Afterwards, this information is used for offline processing and offline analysis for manual optimisation. The information still has its relevance but cannot be used in the same cycle. The constraints that disallow the online usage, to name a few, are: missing dynamic network services; pervasive service chains; and poor real-time response over the path from the machine to the cloud.

3.1.2.2 Issue 2: Online processing and analysis of data but without the ability for using synergies and correlations between data from different machines

One solution could be to process the data on the machine itself, as described in fig. 3.2. In this case, data is processed at the edge and results are online sent for later aggregation to some cloud. With this architecture, the constraint of online processing in the cloud is not given, thus, requirements for the communication between the machine and the cloud are not given as in the prior scenario.

Nevertheless, this constraint reduces the accuracy of any function that is given over the available data because synergies and correlations between data from different machines that have the same setup, material, etc. are not given. At the same time, inefficient setups of processing elements occur because computation power is not shared between systems, but network capacity must be given for additional aggregation.

The logic given by 5GTANGO will improve both scenarios. It will allow to deploy functions to the edge, fog or cloud, and will also allow for appropriate network functions that fulfil the requirements for the data stream connections.

3.1.3 Detailed description

The smart manufacturing use case describes two scenarios that will be addressed by 5GTANGO. The two scenarios are directly addressing the two issues described above. Common to both scenarios is that network functions must be available on the full path from the machine to the fog or cloud.
Figure 3.2: Issue 2: Online processing and analysis of data (at the machine/edge) with the potential to optimize the current cycle but without the potential of synergies and correlations between data from different machines.

Additionally, in both scenarios the machines are running in parallel and the same material is used, but for different variants of products.

3.1.3.1 Storyboard

Initially, both scenarios proceed similarly:

1. machines are running and generate data regarding operations, machine status as well as process data (O/M/P data) derived from different sensors

2. the data generated is made available on a local Industrial PC or HMI, but not directly on the machine computer due to compatibility problems. The Industrial PC or HMI builds a standardised interface independent from the machine vendor

3. the deployment of network functions is performed across the connection between each machine (in this case Industrial PC or HMI) and the fog or cloud system. This includes the following functions as example:
   1. Secure stream separation
   2. Firewalling
   3. Differentiation of performance classes

The next sections highlight the differences between both scenarios.

Use case scenario for issue one

The first issue addressed by 5GTANGO is the sensor data streaming that must allow for real-time interaction with multiple machines. This interaction includes the application response time from cloud/fog, as illustrated in fig. 3.3. As described above, initially, both scenarios proceed similarly. The procedure of the use case scenario for issue one continues as follows:

4. the deployment of services is performed to each machine (in this case Industrial PC or HMI) and the fog or cloud system. This includes the following functions as example:
Figure 3.3: Scenario 1: Online processing and analysis of data (in the cloud/fog) with synergies and correlations between data from different machines due to real-time interaction with multiple machines

1. streaming service for data from the machine to the fog or cloud
2. online processing service of data in the fog or cloud, including data validation, sorting, scaling, aggregation, etc.
3. parameter optimisation service
5. data is streamed using the services from the edge to the cloud
6. data of one or more machines is processed online in the cloud
7. optimised parameters are generated based on the processed data
8. optimised parameters are sent back to the edge and used for next cycle on the machine

Use case scenario for issue two

The second issue addressed by 5GTANGO is the sensor data streaming that must allow for real-time interaction with a single machine for distributed functions of parameter optimisation as illustrated in fig. 3.4. As described above, initially, both scenarios proceed similarly. The procedure of the use case scenario for issue two continues as follows, additionally, with online processing at the edge:

4. the deployment of services is performed to each machine (in this case Industrial PC or HMI) and the fog or cloud system. This includes the following functions as example:
   1. background bulk-transfer service for data from the machine to the fog or cloud for later usage
   2. online processing service of data at the edge
   3. parameter optimisation service
Figure 3.4: Scenario 2: Online processing and analysis of data (at the machine/edge) for fast feedback without synergies and correlations between data from different machines and online processing (in the cloud/fog) with synergies and correlations between data from different machines due to real-time interaction with multiple machines.

5. data is processed by an online processing service on the machine (in this case Industrial PC or HMI)

6. optimised parameters are generated locally on the edge based on the processed data and used for next cycle on the machine

7. aggregated information is streamed using the services from the edge to the cloud

5GTANGO addresses the two mentioned issues. It allows the deployment of required network functions as well as the deployment of the required services that allow on the one hand a more efficient cloud processing, and, on the other hand, enables a much more efficient processing of information on the edge or in the fog that allows for dynamic service allocation based on the decision required. Within the process, it improves the situation that data becomes information too late in the process.

3.1.3.2 Architecture

Fig. 3.5 shows the concept of an industrial network architecture to demonstrate the scenarios described above. The complete network is split into a machine network and an office network. This strategy prevents users of the office network from disturbances caused by users of the machine network and vice versa. Both networks are protected by firewalls. In this architecture a computer which is located in the office network can at most be used to visualise process data from the machine network, but it can not be used to manipulate process data in the machine network. This strategy prevents a machine from undesired data manipulation from the office network. In addition to this, each machine inside the machine network is, strictly speaking, located in its own network and protected by its own firewall. This strategy prevents a machine controller from undesired data manipulation from other machine controllers. Furthermore, it increases the machines’ degree of flexibility because the security options are integrated. This means, machines can physically be relocated without considering the machine’s security. As a result a machine can easily be relocated within the same building, to an other building, or to an other facility.
Each moulding machine is controlled by a programmable logic controller (PLC). Sensors (S1...Sn), e.g., temperature sensors, and actuators (A1...An), e.g., hydraulic cylinders, are connected with the PLC. The PLC reads sensor input data for monitoring the moulding machine or, more precisely, for monitoring the process/cycle and it generates output data for the actuators, which execute/perform the process. The PLC is connected with a machine computer (MPC). The MPC aggregates data from the PLC and offers a human–machine interface (HMI) which visualises process data set and the actual process data, and makes process data modification possible. Depending on the machine, the PLC, the MPC, and the HMI can be represented by one or multiple devices.

### 3.1.4 Requirements

The following requirements are elicited by this use case for the 5GTANGO platform.

#### 3.1.4.1 Virtualization

Several network components are required to create an industry-grade network, e.g., switch, router, firewall, load balancer, etc. Often, multiple devices of the same component are required to create a network; e.g., three switches are required to link geographically distributed users of the same network, which is shown in fig. 3.5 by the simple route from Factory Building 1 to Factory Building 2. Such devices are configured manually by network engineers which takes time, and the requirements to a network can change which makes reconfiguration necessary. Considering the shop floor, machines are added, removed or relocated which requires activation or deactivation of ports. Due to relocating machines, data transfer relationships vary and new network components and new links are required in the new location, while single components in the old location are not longer required. SDN makes networks programmable. SDN makes it possible to orchestrate, manage and control the network by a central configuration manager, which prevents the network engineer from time intensive manual (re)configuration.

NFV makes it possible to reach the full potential of SDN because the network can react autonomously. This means, components and links needed can autonomously be instantiated by the network configuration manager, while components which are not longer needed can autonomously be terminated. This makes the network agile and lean which can reduce energy and costs.

For this purpose, besides switching and routing, the following VNFs are necessary:
- load Balancer
- gateway
- firewall
- intrusion detection
- analytics Engine

Load balancer can be used to distribute the network load, so that the throughput can be increased while the response time can be reduced. In addition, it can be used to prevent a single component from overload. Considering objectives and rules, to be detailed, e.g., costs and delay, the orchestrator decides where the VNFs are placed. At automation level, special protocols are used for machine inter-communications; e.g., EtherCAT, PROFINET, Modbus TCP, etc. The interoperability with such networks can be useful to aggregate much more data. For this purpose, gateways can be used to make communication possible. The 5GTANGO platform should be able to make such gateways possible. Intrusion detection (IDS) and Firewall can be used to prevent unauthorized access; security is crucial for smart manufacturing.

The ultimate target is to securely collect information from the machine network and enable distributed analytics, so that the following services become possible:

- multi-device analytics at the edge
- big data analytics in the cloud
- pre-analytics for reducing data transfer

### 3.1.4.2 Timing requirements

As illustrated in fig. 3.3 and fig. 3.4, data should be processed online in the cloud/fog with synergies and correlations between data from multiple machines. As illustrated in fig. 3.4, data should also be processed online at the edge but for immediate, fast process optimisations of single machines. In both scenarios, timing requirements have to be met to make the tactile Internet possible. Otherwise, the results generated are not useful for optimisation. For the purpose of controlling the process, the 5GTANGO platform has to make low cycle times with low jitter and low latency possible. The cycle time needed depends on the application. Fig. 3.6 shows the application of motion control, which necessitates determinism, a cycle time of less than 1 ms and a jitter less than 1 µs. When the 5GTANGO platform is able to make such applications possible, it could be an alternative to real-time Ethernet protocols, e.g., EtherCAT, PROFINET, etc. Fig. 3.7 shows the application of fast process control, and Fig. 3.8 shows the application of local cloud based process control. In both cases cycle times of up to several hundred milliseconds may be good enough whereas a non-cyclic event-driven process-control could be an alternative application to the cyclic process-control.

Nevertheless, a low delay is required to generate a positive impact with the results generated. It should be able to detect if timing requirements can be met or if timing requirements will be violated. Therefore, mechanism/services for Quality of Service (QoS) are necessary. For this purpose, a classification of data streams regarding timing requirements could be applied. A concept could be to prioritise data streams. A rough subdivision could be the following grouping:

- hard real-time for control with zero tolerance - (missing a deadline is not tolerable; it is a total system failure)
soft or near real-time - e.g., for data acquisition and multimedia applications such as video streaming, Hololens sessions, etc.

- no real-time requirements - e.g., for data transfer: uploading or downloading a file

However, further and more detailed subdivision becomes necessary to prioritise within a group. In this context the capability for determinism using QoS/SLAs will be analysed.

3.1.4.3 Placement

As described in sec. 3.1.4.1 the network becomes programmable, flexible and automated. Considering fig. 3.5 and the timing requirements, described in sec. 3.1.4.2, the placement of VNFs and services plays a major role due to delays caused by distances. Therefore, the service platform should be able to support policies for automatic placement of VNFs and services, e.g., by considering the timing requirements.

3.1.4.4 Security

As described in sec. 3.1.3, a route to the cloud is required for a big data analytics scenario. This implies that data might travel through the public Internet. In order to avert security threats, we need a well-designed security concept. Unauthorised access at the machines, needs to be prevented.
For this purpose, a firewall is shown in sec. 3.1.4.1. Nevertheless, further security components are necessary. The 5GTANGO platform has to support virtual private networks (VPN) for authorised remote access. For example, a VPN is necessary for internal and external remote maintenance. Internal remote maintenance means maintenance by an employee, while external remote maintenance means maintenance by the machine manufacturers.

In addition to this, the 5GTANGO platform should be able to support encryption and security certificates. As described in sec. 3.1.4.2, the real-time and timing requirements have to be considered. Considering challenge two, data is used in the cloud for big data analytics to find synergies and correlations between data from different machines, while reduced data, e.g., 1/10, is used at the edge (near the machine) for analytics for fast feedback, see fig. 3.4 and fig. 3.9. The security concept for data transfer has to be deterministic and it should be able to ensure the timing requirements. For this purpose, a symmetrical end-to-end payload (Ethernet frame) encryption (E2EE) with shared keys, as illustrated in fig. 3.10, is useful to prevent the edge analytics services from re-encryption to meet timing requirements.

3.1.4.5 Compatibility

The 5GTANGO platform should be compatible with standardised infrastructure, e.g., copper cable Cat7a and optical fiber. This is necessary to keep installation costs low. Ideally, it is also downward
Figure 3.10: Symmetrical encryption for cloud based process control and fast edge based process control without re-encryption from edge to cloud

compatible with standardised infrastructure which is already installed, e.g., Cat6 cable and RJ45 modular connectors. This is useful to keep upgrading costs to 5G low.

Following [3], OPC UA is the sole approach for the implementation of a Communication Layer for the Reference Architecture Model Industrie 4.0 (RAMI4.0). Therefore, OPC UA should be supported to guarantee sustainability.

3.1.4.6 Reliability

The V&V platform should be able to prevent the system from unreliable services. For example, considering that TCP and UDP are allowed, and the network is congested, the data rate of the TCP data stream is automatically reduced. In the worst case, the data rate is excessively reduced and the TCP service can not be used; i.e., the service is not reliable. One solution could be to forbid TCP, but TCP is required by particular protocols; e.g., Modbus TCP has become a de facto standard communication protocol in the industrial sector and it uses TCP/IP, and OPC UA becomes a further de facto standard but as communication layer for the Reference Architecture Model Industrie 4.0 (RAMI4.0) and it also uses TCP/IP. Therefore, mechanisms on the service platform are required to prevent services from being unreliable (e.g., QoS guarantees). On the SDK side, mechanisms to specify required QoS properties are needed.

A central control unit for the configuration management makes sense, but the 5GTANGO platform should be able to avoid particular problems such as a Single Points of Failure (SPoF). For this purpose, no central control unit for communication management should be allowed, but could be okay for configuration management.

3.1.4.7 Scalability

It makes sense to separate the machine network logically but not physically from the office network to keep installation costs low. For this purpose, virtual local area networks (VLANs) are used. Therefore, VLANs should be supported.

Considering the Industrial Internet of Things (IIoT), the number of devices supported for integration should not be limited, or, more precisely, it should not be too low. For example, IPv4 limits it to approximately 4.3 billion ($2^{32}$) addresses, while IPv6 supports up to $2^{128}$. Therefore, the 5GTANGO platform should support IPv6.
Several topologies are used in machine networks, e.g., tree, star, ring or daisy-chain. Such topologies should be supported by the 5GTANGO platform.

The Service Platform itself must be able to grow and shrink automatically with its workload, thus minimising operating expenditures (OpEx), as well as supporting the scalability of the services and functions on-boarded and instantiated in it. Specially valuable is the capability of starting small, to support quick proof-of-concept deployments. For example, VNFs/services are automatically added when they are required, maybe due to workload increased due to machines added. In addition, VNFs/services are automatically removed when they are not longer required, maybe due to machines removed.

3.1.4.8 Reconfigurability

The configuration and reconfiguration of the network should be done automatically; i.e., Plug and Play should be supported, so that new devices and services can be added to the 5GTANGO platform and removed but without powering down it. Machines added or removed should not require a manual redeployment of the network service.

3.2 Immersive Media Pilot

3.2.1 Executive summary

The Immersive media use case is designed to use the capabilities of new technologies such as VR, 360° Video Capturing cameras, 5G networks, vCDNs and ability of new GPUs to render 360° content with low latency. The pilot will capture a live event using a 360° camera at 4K resolution and stream it to users, sitting at the leisure of their home, on their VR head mounted display (HMD), desktops or smartphones with enrichment content added to it enabling the personalisation of the live stream. The development for different platforms will be done using Unity game Engine due to its compatibility, APIs available and a great community support of developers. Unity is a cross-platform game engine developed by Unity Technologies. Unity is an all purpose game engine that supports 2D and 3D graphics, drag and drop functionality and scripting through C# and JavaScript.

New forms of media content such as very high resolution content (4K and above), High Dynamic Range (HDR) content, or immersive VR and 360°, are quickly being adopted by the market. Today, nearly every major player in the video streaming industry has made its first steps into adopting some or all of these technologies, which will soon find their way in our daily life. End-users consume video streaming typically relying on different devices such as smartphones, tablets, PCs, SmartTVs, etc. The content must undergo different network connectivity conditions on the basis of the user location (whether home, office, or on the road), and the time the content is being accessed. To counteract fluctuating network conditions and also to cope with heterogeneous user requests, efficient delivery streaming frameworks such as Dynamic Adaptive Streaming over HTTP (DASH) were recently introduced.

3.2.2 Rationale

5GTANGO will enable distribution of these new forms of media content by devising novel technology (but based on the actual State of the Art) for media content compression and distribution, efficiently exploiting adaptive streaming protocols. Moreover, 5GTANGO will investigate merging of different forms of media, including personalised content and interactive gaming experiences, to support a unique way of experiencing media, while also constantly monitoring end user QoE to
ensure that content is seamlessly delivered at the highest levels of quality. So future media solutions need to overcome several challenges: * to compress, stream, fuse and synchronise new forms of media (UHD, HDR, 360° in 4K, AR/VR) * to create an immersive viewing experience

The use case will validate the following system characteristics: 1. to deliver an adaptive and immersive end-to-end live streaming service as the combination of 360 degree in 4K streaming customized using AR/VR contents 2. to deal with user co-ordination and media synchronization. The two streaming channels (live video and enrichment) will be fused in user terminals (in the edge) 3. the live-streams will be platform and device agnostic. Relying on DASH/HLS and an existing HTML5 adaptive player, it will be consumed by different devices: mobile, tablets, laptops and SmartTV and the streaming will be adapted to the characteristics of such devices 4. to deliver a Hyper-personalized experience developing a personalized AR enrichment channel using Facebook data, Twitter data and youtube recommendations. (manual and automatic personalisation of the experience, deciding contents to see, people to reach, etc.)

### 3.2.3 Detailed description

#### 3.2.3.1 Virtual CDN

As it is widely known, CDNs allow the efficient and scalable distribution of contents. CDNs are widely used for the distribution of audiovisual contents in Internet. 5GTANGO proposes an innovative perspective: the use of virtual CDNs, taking advantage of cloudified environments. Since 5GTANGO aims at the maximisation of the QoE, a network elastic topology is required to meet the service instantaneous requirements, such as new nodes (depending on the number of users), load balance and server replication. A specific challenge is also the operation of the Wowza engines in the virtualised network. Wowza Streaming Engine is a unified streaming media server software developed by Wowza Media Systems. The server is used for streaming of live and on-demand video, audio, and rich Internet applications over IP networks to desktop, laptop, and tablet computers, mobile devices, IPTV set-top boxes, internet-connected TV sets, game consoles, and other network-connected devices. In principle, two alternatives are possible: multiple CDN nodes distribute the streaming content coming from a Wowza engine or multiple Wowza nodes adapt the content across the CDN. Unfortunately, the use of virtualization technology has a slight impact on the performance of the video servers and CDN nodes, there will be a loss of performance compared with the same application running on the “bare metal”. One of the aspects to analyse is the penalty imposed by the different virtualization technologies. 5GTANGO platform allows this kind of experimentation and validation through the V&V environment and also in-situ monitoring. DevOps approaches will allow modification of the service during runtime in order to optimise the platform. The use of scalable NS and VNFs will allow to diminish the overhead caused by virtualisation by distributing the effort to more instances.

#### 3.2.3.2 Streaming Software

This streaming technique allows the video player on the terminal to dynamically adapt the video bitrate (quality) to the available bandwidth at the user side. Thus, the video starts faster, buffering during the streaming session is prevented, and a higher quality viewing experience is delivered. The use case will demonstrate how 5GTANGO platform is designed to support global audiences streaming with high-quality and immersive video experiences. Adaptive streaming improves QoE since it avoids streaming interruptions. If the network experiences instantaneous problems, a lower quality is admissible but never the service interruption. It is especially important for mobile networks, where the user can be moving and the level of service of the network can change very
quickly. This use case considers a variety of user devices, such as tablets and head-mounted devices (HMD). As these devices support different characteristics and technologies, the automatic adaptation will enable the right video resolution for each device (e.g., HMD equipment may not support 4K and it would require a smaller resolution). This use case proposes Wowza as video server due to these advantages:

- it is a mature commercial product widely used, thus it offers a good reliability for the service provision
- it integrates adaptive streaming capabilities, as required in this use case. Wowza supports the main adaptive streaming technologies: HLS, MPEG-DASH, HDS and Smooth Streaming
- it integrates transcoding capabilities (by means of an additional product) to create the multiple adaptive streams in real time

Wowza brings some challenges too. In the case of 360° video, an equirectangular projection is used for mapping the original video signals. Additional metadata are required to encode the scene spatial information. As Wowza is not 360° aware, this use case will analyse the preservation of these metadata through Wowza streaming engine and Wowza transcoding capabilities.

### 3.2.3.3 Enrichment Channel

The enrichment channel will be basically a server or an integration to end-user applications to enable the user to apply personalised comments, profile and other features for example recommendations on the video feed they receive. This will work on VR HMD, Desktop as well as smartphones that we plan to target. The enrichment channel will feature the following services:

**YouTube API**

YouTube API, allows developers to access video statistics and YouTube channels’ data via two types of calls, REST and XML-RPC. The data and analytics API lets developers incorporate YouTube functionality into their applications. They can use the API to fetch search results and to retrieve, insert, update, and delete resources like videos or playlists. youtube.videos.list method in the data API allows the developers to get the list of videos that match the API request parameters. This combined with the youtube.videoCategories.list method which gets all the keywords (categories) associated with the video will allow to get similar video recommendations. One of the advantages of using the YouTube Analytics API in the system so that one could easily implement pay-to-watch model or an advertisement based model during exploitation.

**Facebook API**

Facebook “platform” is a term used to work with the services Facebook provides to developers to create their own applications and services to access data in Facebook.

**Facebook Login** “Facebook Login” helps developers use the Facebook accounts of people to log into their applications across multiple platforms. There are 3 major experiences in the login plug-in of Facebook.

1. **account Creation**: This helps in logging into an application using their Facebook password. No separate password is therefore required to enable login. The developer will receive a validated email address to enable multi-platform login authentication
2. **personalisation**: Facebook login gives us the ability to access information regarding users' preferences, their personal data, location and their pictures for profile pictures and other purposes.

3. **social**: The login gives access to users’ friends and helps them connect with their friends on our platform and thus be more socially active on the platform.

Some of the features of the login plug-in include: real identity, cross platform login, granular permissions, and gradual authorisation.

**Twitter API**

For live streaming devices, it has become common to attach Twitter's tweet streaming service to the view to be more immersive. Twitter provides several APIs to login, tweet and to fetch tweets. For authentication, Twitter uses OAuth for authentication of external applications with the data on twitter. **Twitter Kit** provides an SDK which can be integrated into Unity for sharing tweets which will be used in the 5GTANGO media pilot. The Twitter>Login method is used to log into Twitter. The user can compose tweets and share them with their followers. **Streaming API** provides low-latency access to Twitter’s global stream of Tweet data. A streaming client will be pushed messages indicating Tweets and other events have occurred, without any of the overhead associated with polling a REST endpoint. The public streams, once authorised using OAuth, will be received as JSON-encoded data.

**Native Profile and Comments**

For the 5GTANGO platform to target any type of audience and work independently, we will incorporate native comments and a profile for each user where he can set up his homepage, preferences and other settings to have the best experience. The platform will be device-agnostic and work as a progressive web app for the desktop and smartphone and natively on VR devices. The user can log into Facebook, add privacy settings, friends and use the platform to comment on the content they are viewing. The personalisation of the UI will be based on these preferences and can determine what types of live streams users will be interested in watching once the product is commercially available. The user should be able to communicate with their friends and others using the comments section on the screen. The front end development will be done on Unity3D for native application and HTML, JavaScript for the progressive web application. The server for this can be done as a virtual server or as a physical server, needs to be decided based on the capabilities of 5GTANGO.

### 3.2.3.4 End-User Device

**Desktop and Smartphone/Tablet**

For desktop, we plan to develop a web app interface instead of a native application as we can provide ability for 3rd party developers to build on top of the web application and for the users to use the application on any operating system. Web sites most likely to be referred to as “web applications” are those which have similar functionality to a desktop software application, or to a mobile app. **HTML5** introduced explicit language support for making applications that are loaded as web pages, but can store data locally and continue to function while offline. The web app will be based on progressive enhancement methodology which means that they are technically regular web pages (or websites) but can appear to the user like traditional applications or (native) mobile applications. To this end, we will develop a hybrid app which mimics the native mobile UX and needs app store download. Partially running in a mobile browser, hybrids do not have a URL, support a rich
user interface, and can access system capabilities. The combination of markup, styling sheets, and scripts enables custom interactive page elements without the use of closed systems such as Flash. For the live streaming capabilities on the platform we intend to use Valiant360, Axis, Google's VR view or similar open source tools which support HLS capability of the browsers, video rendering capability of the user hardware and can also support DASH streaming.

3.2.3.5 Virtual Reality Display

For 5GTANGO pilot we plan to use HTC Vive as the head mounted display for the virtual reality experience. Unity3D game engine has introduced inbuilt support for VR devices (HTC Vive included) and virtual reality development. Due to the inbuilt system, Unity3D allows to see the previews of the application on the VR device during development and not only after exportation. Virtual reality headsets have significantly higher requirements for latency—the time it takes from a change in input to have a visual effect—than ordinary videos. If the system is too sluggish to react to head movement, then it can cause the user to experience virtual reality sickness, a kind of motion sickness. Due to this it is important that we use a rendering hardware compatible of decoding 4K 360 stream in realtime and supports network connectivity to get the stream. Unity3D asset store and the strong user community provides a lot of open source resources to play 360 videos in VR devices. For the input devices, we plan to use a VR keyboard such as the Cutie Keys drum keyboard open project and use a drum stick methodology for mouse as well using the HTC Vive controllers.

3.2.4 Storyboard

Lisa is a technology enthusiast and has interest in Tennis and other sports activities. She loves watching Tennis matches but it is hard to attend all the matches in person since they happen all over the world. She recently bought a HTC Vive to test games and media applications in VR and see the reason for the emergence of the technology. She downloads and installs the HTC Vive setup application and Viveport (application that connects HTC Vive applications to the device) on her PC and downloads the TANGO Media Application for HTC Vive.

Lisa starts the TANGO Media Application and logs into her newly created account on the platform. The application supports Lisa in watching her favourite matches of tennis. Rafael Nadal, one of her favourite Tennis players is playing against Roger Federer in Spain where a 360° video system is able to capture multiple videos and stitch them. Using the HTC Vive and the 5GTANGO application, Lisa can feel herself in the tennis court, she can watch freely all aspects of the game, including the public surrounding her (virtually). The sport watching experience is improved.

Marc is the production manager of the company in charge of the production and the distribution of the 360° video service, i.e. the service provider. Marc is the responsible for the programmability and the initialisation of the service. For this purpose, he has edited a descriptor, which deploys a set of concatenated functions required for the acquisition, composition, encoding and distribution of the 360° videos. In this way, the high quality 4K 360° video signal is available by users, like Lisa.

Lisa can choose the Social Media accounts she would like to attach to, like Facebook, Twitter and Youtube. Apart from this, Lisa can post comments about the game, have conversation with other watchers/users on the 5GTANGO Media Application on any of the targeted platform. Lisa can change the interface of the application on her device, the streams she wants to watch and the people she wants to interact with.
3.2.5 Architecture

See Figure fig. 3.11

3.2.6 Requirements

- programmability:
  - reparation of the Media Streaming Service – composition of several functions - “prepare service” (a chain) in the flow cameras (4K 360°) and vCDN streaming elements
  - ability to integrate cameras (physical devices, non-virtualized)
  - initialisation of the service
  - easy on-boarding process to the Service Platform
  - capability to verify service functionalities, consistent interconnection and security aspects

- multi Point of Presence:
  - the deployment of the service goes across several network segments and PoPs

- placement of virtual instances at the edge. It is necessary to indicate to which concrete edge nodes certain functions need to be deployed (selection over available topology, consistency with localisation of service)
• automatically place and interconnect the VNFs, assuring SLAs to support live streams.

• dynamicity in the created service chain

• chained elements can change or being modified during execution and its configuration or behaviour

• insertion of monitoring elements in the chain to check QoS/QoE of service (sniffer VNF)

• multi VNF support:
  – Facebook VNF
  – Twitter VNF
  – Media server VNF
  – Data collector VNF

• encoding VNF: Support for multiple Virtualisation technologies
  – Docker containers (light functions)
  – VMs over OpenStack

• multi-Infrastructure Manager

• session handling
  – Disconnection of the Application. Close sessions
  – Control lifecycle and finalise (Destroy) the Media service.

• scalability may be needed at both edges of the network, to serve more end users (edge) as well as content producers (backhaul part)

• low latency (for enabling live streams)

• dynamic(Different) latencies for live streams and communication

• H264 encoding bitrates to ensure video quality: 50 Mbps – 100 Mbps

• may have:
  – security protocols and Scalability
  – mobility Handling (handovers)
  – support for 3Rd Party VNFs
  – support for 3rd party data
  – stream protection / encryption
3.3 Communications Suite

3.3.1 Executive summary

In real-time unified communication system implementations it is necessary to deal with some difficulties of security (firewall rules setup, routing setup, permissions to access servers and start/stop/modify services, etc.), when monitoring and debugging (setup service and agents to monitor the machines, set the log level on a per service basis, gathering and processing of logs in a centralised storage, etc.), when scaling up the solution (due to human intervention to clone machines and modify them, resource allocation, etc.), and validation and performance tests must be adapted to each environment.

The goal of this pilot is to deploy an operator real-time communication platform using the 5GTANGO platform to provide a self-scalable and easy to provision system.

The user will access the service using a Unified Communication client which allows to create videoconferences with multiple participants. The solution also supports chat, group chat, file sharing, desktop sharing and other valuable features provided by services which can be easily scaled up like integration with Public Switched Telephone Network (PSTN).

3.3.2 Rationale

This pilot fits very well in the set of features provided by the 5GTANGO platform as it requires the deployment of many different components, some of them provide standard services used by other architectures (i.e. Reverse Proxy and TURN server) and others are especially designed to cover communications. All the service instances can be scaled up by deploying additional instances in order to cover the load required by the users.

The signals which trigger the need for more resources will depend on the intensity of use of the different functionalities provided by the Communications Suite. For example, when the aggregated number of video-conference participants reaches a threshold, a new SFU (Select Forwarding Unit, the service in charge of sending/receiving unicast video flows) instance is needed and the same happens with the rate of chat messages per second which may require an additional instance of the chat service. The platform has also a recording service which requires a post-process to mix the recordings. This service is CPU-intensive but flexible in terms of time. The assignment of CPU resources defines the time span the user will have to wait to get the recording file once the videoconference has finished.

Another key point is that the different services will require different resources. The SFU, for example, requires a significant amount of bandwidth with low latency while other services such as the processing of video recording are very intensive in terms of CPU.

- relation to Challenge 3 Mapping templates vs. instances:* This pilot includes services which are intensive in different resources (CPU, Bandwidth) or sensitive to server/network performance (delay-sensitive services) so it will take advantage of the templates that will be included in TANGO. They will allow to define all the requirements of the services when deploying the instances

- relation to Challenge 4: Verification and validation of VNF:* All the present elements should be verified and validated in order to assure the stability and the correct behaviour of the service

- relation to Challenge 5: DevOps and monitoring loop towards autonomicity:* This pilot will take advantage of the autonomicity features to be included in 5GTANGO. The utilisation
of appropriate monitoring probes will allow to monitor the state and the conditions of the
different virtual network functions in real-time and will instruct the FSM/SSM to carry out
the required actions in order to scale and maintain the service performance according to the
SLA. For example, if the use of the SFUs (Selective Forwarding Units) is very high due to an
increase in the number of simultaneous videoconferences, then the system should be able to
increase the number of SFU instances

3.3.3 Detailed description

This pilot addresses the implementation of a real-time unified communication system in order to
test how the 5GTANGO platform helps to deal with security, monitoring and debugging issues
when scaling.

The services needed to setup the communication pilot can be divided in three categories:

1. **signalling/functionality services.** These specialised services are in charge of the exchange
   of messages to set up videoconferences and also to provide services such as chat or file transfer.
   The WAC (WebRTC Application Controller) developed by Quobis and the QSS (Quobis
   Signalling Service) are the main services

2. **media services.** These services are focusing exclusively in dealing with the media. The
   main media service is the SFU (Selective Forwarding Unit) already mentioned and also
   the recording processing server, in charge of processing the raw video files captured by the
   SFU and produce video mixing the different flow in a reproducible format. The Audio-
   mixer service is in charge of the combination of audio flows from different sources. Both
   the SFU and the audio-mixer can record video and audio, respectively. The SFU gets the
   raw video from an RTP stream. In a post-process phase, it mixes the videos coming from
   the different participants and converts them to a format which can be reproduced by the
   browser. This process is carried out in external server (Recording-processing server) and it is
   a CPU-intensive task

3. **auxiliary services.** These non-specialised services may be used by other systems in order to
   provide their functionality in various settings. In order to speed up and make the deployment
   of the Communication Suite easier, it should be possible to deploy those services from pre-
   existent images. The required proxy server (i.e. Apache or NGINX), a TURN server (which
   is in charge of relaying media flows to and from endpoints behind NAT and Firewalls), a
   RabbitMQ and a MongoDB service lie within this category

3.3.4 Architecture

The architecture is depicted in Figure fig. 3.12. It shows the required components and their inter-
actions. N instances of the application services (WAC) can be running at a same time depending
on the service load. The signalling service receives the connections from the external users. It
has several micro-services that could be split into separate services. The SFU is the element that
receives the traffic containing the media from the different users and send the media back to all
the participants in the same session or video conference room. It is a very bandwidth consuming
element in the architecture. The audio-mixer manages the media and integrates different audio
sources into a same conference call. The TURN server allows to fix networking problems caused
by NAT, firewalls, etc. Both the RabbitMQ and the MongoDB can have more than one instance
running at the same time in cluster configuration. The dispatcher is a registry for SFU instances
3.3.5 Storyboard

The steps below define the basic flow of actions that happens since the user login until joins a videoconference room:

1. the users will login and register against the WAC. Additionally to the login and registration, the WAC can provide other features to the apps such as contact management, call history and gathering statistics of use. The request to login is sent to the reverse proxy and it is sent in turn to a WAC instance

2. once the app has correctly logged into the WAC, it receives the URL of the signalling service to register

3. once app registers against the signalling service is ready to create videoconferences and be invited to other user’s room

4. when the app decides to start call, the QSS will contact the dispatcher and an SFU will be assigned. This information is passed to the app which creates the room

5. the signalling service notifies the videoconference to the rest of participants and includes the SFU information to all of them, it means that all the members of the room will use the same SFU instance
6. reached this point, all the participants will create a WebRTC real-time audio/video session and will start publishing the video flows to the room and receiving the flows from the room created in the SFU. The media can be sent directly to the SFU from the device or it can be sent through a TURN server, in case the users are behind a restrictive firewall.

7. monitoring probes will collect information from the communication service as well as from the other VNFs in the NS. Service-specific management modules will assess the collected information and decide on the placement of new instances to scale the service or optimise its delivery.

8. at any point the users can leave and join the room until it finishes and the room is deleted from the system.

9. once the videoconference is finished, the recording server retrieves the files from the SFU and the audio-mixer in order to convert the recording file in a playable format.

### 3.3.5.1 Triggers to modify the number of required instances

The list below gathers the main triggers to increase the number of instances of the system:

1. if the number of provisioned users increases by 50,000, then a new WAC instance must be added to the cluster.
2. if the number of registered users increase by 20,000, then a new WAC instance must be added to the cluster.
3. if the number of calls increases by 500, then add a SFU instance must be added to the cluster.
4. if the usage of BW in the SFU exceeds 70%, then a new SFU instance must be added to the cluster.

### 3.3.6 Requirements

#### 3.3.6.1 Functional requirements

1. the SDK should allow to define the thresholds that trigger different actions (e.g. the creation of new instances, increase of resources, etc.)
2. the 5GTANGO platform must provide the capability to the developer/user to modify the monitoring metrics thresholds that trigger actions and/or the addition/deletion of metrics per network service and during run-time.
3. the actions triggered must include the creation/deletion of new instances.
4. the thresholds must be able to be set at least by CPU, bandwidth consumption and numeric values which can be set by some instances (e.g. number of registered users).
5. the platform must provide synchronous and asynchronous mechanisms to the developer/user to retrieve data and alerts and act accordingly.

#### 3.3.6.2 Non-functional requirements

1. when two different instances work combined, the SFU instance must be deployed along with an audio-mixer instance in the proposed system.
Network Non-functional requirements

1. the latency of the SFUs should not be higher than 20ms and they should be as close to the human user as possible

2. the RTT between the device of the user and the WAC should be lower that 40ms in order to keep the app as responsive as possible

3. the system must be able to handle BW intensive apps

4. considering X as the number of simultaneous users of an SFU and Y the max number of participants supported by a single videoconference room ,then the SFU instance must be able to support X uplink 300kbps video flows from the end-users and Nx(Y-1) 300kbps downlink video flows from the SFU to the users. For example, let’s assume X=100 simultaneous users and Y=5 max participants per room, then the SFU instance must be able to receive 100x300kbps=30Mbps and 100x4x300kbps=120Mbps downlink

Monitoring Non-functional requirements

1. due to the real-time nature of the communications suite scenario, the collection and processing of the monitoring data as well as the alerting mechanism to the developer/user must be (near) real-time

2. the collection of data from different types of “sources”, such as virtual machines, containers, OpenFlow controllers, cloud infrastructures, etc, must be supported by the 5GTANGO platform

3. the scenario will include the instantiation of distributed network service, whose virtual network functions could be deployed in more than one NFVI and/or controlled by more than one VIMs/WIMs. Collection of monitoring data from several NFVI and processing on a centralised manner must be supported

4. the monitoring framework must facilitate the accurate measurement of active and passive networking and simplify to the maximum extent the conditions for the integration of monitoring metrics specific to the network function/service

5. the communications suite comprises of signalling/media/auxiliary services and thus there is a need for an enhanced list of probes to be supported covering networking, computation and memory resources
4 Storyboard

The short stories in this section outline how the 5GTANGO project currently expects its developments to be used by different stakeholders for different purposes. The stories are described from an infrastructure point of view. They do therefore not differentiate between the three pilot scenarios described before. In fact, we expect all stories described here to be used as part of all the pilots. What will vary across pilots is which particular stakeholders assume which specific roles in the stories. The role “operator” might be assumed by a manufacturing company in the smart manufacturing pilot, while an Internet Service Provider (ISP) might assume that role for the communications suite pilot.

The stories represent the expected main way of using the 5GTANGO system, but deviations are possible and expected, depending on the environment and circumstances the components are used in. Some of the anticipated deviations are marked with the “Variant:” prefix in the stories below.

The flow of actions and the component interactions represent the project’s current thinking. These can change over time with the acquisition of more knowledge and experience. In particular, it might happen that additional components will be added to the flows. Likewise, some components might be removed from the flows.

In general, we have to distinguish between different phases that may overlap but happen at different timescales and are performed by different actors:

1. phase 1: development of network service (NS), and publishing of NS (+ test results) in a catalogue
2. phase 2: testing/validation with a V&V platform, and publishing of results in a catalogue
3. phase 3: selecting NS from a catalogue, deployment and operation of NS using the service platform orchestration: scaling, placement, etc.

fig. ?? presents the mapping to the V&V lifecycle as described in sec. 2.4. The stories below are structured according these three phases, but the phases are broken down into more specific stories and explained with more detail.

4.1 Roles

We have the following roles that can be mapped to persons, companies, startups. Of course, multiple roles can be taken by a single individual and can somewhat overlap.

- developer: Develops NS, tests it, publishes it
- operator: Selects existing NS, tests it, deploys it
- V&V provider: Provides testing environment, executes tests, returns signed results, maintains the results in catalogues
4.2 Initial state

- There are people in three roles: developer, V&V provider, operator
  - people can assume multiple roles; in the degenerate case, one person assumes all three roles
- Developer has set up the 5GTANGO SDK, including his own local catalogue, local V&V instance and infrastructure emulator
  - Variant: instead of using an emulator, the developer operates his/her own real infrastructure, connected to his own local V&V instance
- V&V provider has a bunch of servers, storage, and networking equipment
- Operator has set up his/her own 5GTANGO service platform (SP)

4.3 Create a service

This story describes the workflow for creating a 5GTANGO network service. The sequence of actions is shown in fig. 4.2. The numbers in brackets refer to the steps in the figure.

- developer wants to create an awesome network service (NS) using the 5GTANGO tools
- she/he browses some VNF catalogues to find the VNFs he needs
  - he describes the needed VNFs’ functionality and non-functional requirements with the SDK (1)
…he searches the catalogues for matching VNFs and selects the ones to use (2-5)

- Variant: instead of using external VNFs, developer might have written his own VNF(s) and select them from his local catalogue

• with the SDK, developer designs the network service
  - he connects the needed VNFs (i.e. he creates a service graph) (6)
  - he adds metadata to the service (7)

• with the push of a button, developer create the NS (8)
  - based on the metadata, the SDK generates all the needed descriptors (9)
  - the SDK bundles all required information up into the service package (10)

• developer now uploads the service package to his local catalogue instance (12-13)

• optional: Developer uploads the service package to other, remote catalogue instances

• optional: Developer tests the NS on his local (emulated) infrastructure and with his local V&V instance, using the tests created as described below

4.4 Create tests

This story describes the workflow for creating test plugins for the 5GTANGO V&V infrastructure. The sequence of actions is shown in fig. 4.3. The numbers in brackets refer to the steps in the figure.

- developer (could be a different one than in previous story; without loss of generality, assume same one for now) writes tests with his favourite development tools for testing functional and non-functional (e.g., throughput) aspects of his service (1)

- developer uses the 5GTANGO SDK to annotate the tests with metadata, describing purpose of tests, link to tested NS, etc. (2)

- developer uses the 5GTANGO SDK to create a test package, including test code, VMs, containers, models and descriptors, based on metadata (3-5)

- developer uploads test package to a catalogue (6-7)

4.5 V&V setup

This story describes the workflow for setting up a V&V environment. The sequence of actions is shown in fig. 4.4.

- V&V provider wants to provide verification & validation services to 3rd party developers

- V&V provider sets up servers, storage, networking gear

- installs and configures the 5GTANGO service platform (SP)
Figure 4.2: Communication flow for service creation
• installs and configures the 5GTANGO catalogue and V&V components on her infrastructure

• adds tests created by herself or some other developers (taken from catalogues) to the V&V component

• opens the installation to 3rd party developers for them to run tests on

• optional: The developer sets up a local installation of the V&V components that execute the services and tests in an local environment either based on the SDK’s emulator or a local service platform installation

4.6 V&V tests execution

This story describes the workflow for running tests of a network service in an external V&V environment. The sequence of actions is shown in fig. 4.5.

• developer wants to test his network service on an infrastructure more capable than his own local setup

• She/he selects an appropriate V&V provider
  – He browses the offerings of some V&V providers
  – He selects the one with the best offering: most suitable infrastructure and appealing price tag

• developer uploads service package to V&V provider’s catalogue
developer uploads test package to V&V provider’s catalogue
developer triggers test execution on V&V’s infrastructure
V&V runs tests
  – service is deployed and started
  – tests are deployed
  – tests are run
  – service and tests are undeployed
  – statement on test runs and collected information (pass/fail, throughput figures, etc.) is created
    * includes information on services run, tests run, infrastructure used, V&V provider, ...
  – test statement and test results are bundled and put into catalogue
V&V notifies developer of test result availability

4.7 Run Service
This story describes the workflow for deploying and starting a service on an operator’s service platform and infrastructure. The sequence of actions is shown in fig. 4.6. The numbers in brackets refer to the steps in the figure.
Figure 4.5: Communication flow for V&V selection and test execution
• operator looks for a particular service with specific properties, e.g. a secure Industry 4.0 service able to handle up to 100M packets per second with 1ms latency

• operator learns about developer’s awesome service by querying the V&V provider’s catalogue with his functional and QoS requirements (1)

• operator downloads the service and attached test statements from V&V provider’s catalogue (2)

• operator looks at test statements and decides that it trusts V&V provider, and used infrastructure is representative of his own (3)
  – Variant: operator downloads NS and runs (additional) tests on his own infrastructure using his own V&V instance

• operator deploys service on his infrastructure, using the 5GTANGO service platform (4-6)

• operator is starting the service (7-8)
  – optional: Developer can have specified custom lifecycle operations for her service using the specific manager approach (FSM/SSM) offered by the service platform
  – optional: The service platform can do smart resource assignment decisions throughout the lifecycle of the service by utilizing available test results
5 5GTANGO Requirements

5.1 Functional requirements

In software engineering, a functional requirement defines a function of a system or its component. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that defines what a system is supposed to accomplish. A function is described as a set of inputs, the behavior, and outputs. Usually the description also contains a unique name and number, a brief summary, and a rationale.

5GTANGO considers four main functional blocks; catalogue (data store, decision support and optimization), verification and validation (V&V) platform, NFV-enabled software development kit (SDK), and extended service platform to support vertical applications. The following section addresses the high-level and low-level requirements for 5GTANGO’s main functional blocks.

5.1.1 Catalogue

5GTANGO’s catalogue needs:

i) to store the VNFs/NS data and the associated metadata information (V&V results, deployment patterns, monitoring information, QoE/QoS parameters, SLAs and policies);

ii) to provide decision support for the automated analysis of metadata in order to enable efficient selection and use of VNFs/NS;

iii) to enable continuous optimisation mechanism through analysis of the updated VNFs/NS metadata in order to trigger new V&V tests (feedback loop towards V&V) and actions towards the provision of QoS guarantees;

iv) to facilitate revision and version control for all information included in the Catalogue;

v) to allow synchronisation with other catalogues.

5.1.1.1 Key interactions of the Catalogue

- description: the 5GTANGO Catalogue should be offering interfaces for interacting with other components

- KPIs:
  - interfaces for searching, storing, retrieving VNFs/NS and their metadata
  - interface for enabling the deployment of VNFs/NS
  - number of stored VNFs/NS
  - interface to V&V for obtaining results
  - interface to V&V for triggering new tests based on continuous optimisation outcomes
  - interface to Profiler for obtaining profiling results
- interface to Policy Management Framework for obtaining associated policy information
- interface to Monitoring Framework for obtaining monitoring information
- interface to Network Infrastructure Management for proposing actions regarding QoS enforcement

- related pilots: all
- category: must
- target WP: WP3

### Table 5.1: Low-level functional requirements for key interactions of the Catalogue

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAT01</td>
<td>The catalogue must provide interfaces for storing, retrieving and searching VNFs/NS and their metadata</td>
<td>Interfaces to all 5GTANGO components will be developed</td>
</tr>
<tr>
<td>CAT02</td>
<td>The catalogue must provide an interface for enabling the deployment of VNFs/NS</td>
<td>Interface supporting an identifier to obtain the NFV to be deployed</td>
</tr>
</tbody>
</table>

#### 5.1.1.2 Management of metadata

- description: the 5GTANGO Catalogue should be offering support for metadata annotation and update of the stored VNFs/NS

- KPIs:
  - interface to V&V for storing test results as metadata as well as updating metadata from new test results
  - interface to Profiler for storing and updating profiling metadata
  - interface to Monitoring framework for storing and updating monitoring information
  - interface to Policy management framework for storing and updating associated policy information
  - stored metadata
  - up-to-date metadata

- related pilots: all
- category: Must
- target WP: WP3

### Table 5.2: Low-level functional requirements for management of metadata

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAT03</td>
<td>The catalogue must provide an interface to V&amp;V for storing test results as metadata</td>
<td>Test results with several “metrics”</td>
</tr>
<tr>
<td>CAT04</td>
<td>The catalogue must provide an interface to Profiler for storing profiling metadata</td>
<td>Metadata related to profiling outcomes</td>
</tr>
<tr>
<td>CAT05</td>
<td>The catalogue must provide an interface to the Monitoring Framework for storing monitoring information</td>
<td>Storage of monitoring metrics</td>
</tr>
<tr>
<td>#ID</td>
<td>Description</td>
<td>Comments</td>
</tr>
<tr>
<td>------</td>
<td>------------------------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>CAT06</td>
<td>The catalogue must provide an interface to the Policy Management Framework for storing associated policy information</td>
<td>Metadata related to policies</td>
</tr>
<tr>
<td>CAT07</td>
<td>The catalogue must provide an interface to V&amp;V for updating metadata from new test results</td>
<td>Updated “metrics” from new results</td>
</tr>
<tr>
<td>CAT08</td>
<td>The catalogue must provide an interface to Profiler for updating profiling metadata</td>
<td>New profiling outcomes</td>
</tr>
<tr>
<td>CAT09</td>
<td>The catalogue must provide an interface to the Monitoring Framework for updating monitoring-related information</td>
<td>Updating of stored monitoring metrics</td>
</tr>
<tr>
<td>CAT10</td>
<td>The catalogue must provide an interface to the Policy Management Framework for updating associated policy information</td>
<td>Metadata from updated policies</td>
</tr>
</tbody>
</table>

### 5.1.1.3 Provision of decision support

- **description**: by using technologies for the automated analysis of metadata (including V&V metadata, deployment patterns, monitoring information, QoE/QoS parameters, SLAs, policies, etc) the catalogue should enable efficient selection and use of VNFs/NS

- **KPIs**:
  - interface to the policy management framework related to the selection of VNFs for the provision of quality guarantees
  - optimal selection of VNFs/NS by the network operator, based on their requirements and constraints

- related pilots: all
- category: must
- target WP: WP3

**Table 5.3: Low-level functional requirements for provision of decision support**

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAT11</td>
<td>The catalogue must incorporate a mechanism to analyse all metadata in order to provide the optimum selection of VNFs/NS</td>
<td>Metadata to be analysed include V&amp;V metadata, deployment patterns, monitoring information, QoE/QoS parameters, SLAs, policies</td>
</tr>
<tr>
<td>CAT12</td>
<td>The catalogue must provide an interface to the policy management framework related to the selection of VNFs for the provision of quality guarantees</td>
<td>Supports information that enables the optimal selection of VNFs/NS</td>
</tr>
</tbody>
</table>

### 5.1.1.4 Provision of continuous optimisation

- **description**: by analysing the updated VNFs/NS metadata (updated based on new information from SLAs, policies as well as monitoring – QoS and QoE data -) new V&V tests should be triggered (feedback loop towards V&V) and actions towards the provision of QoS guarantees should be proposed.

- **KPIs**:
  - interface to V&V for triggering new tests based on continuous optimisation outcomes
interface to Network infrastructure management for proposing actions regarding QoS enforcement

– number of triggered tests

– actions for QoS enforcement

• related Pilots: all

• category: must

• target WP: WP3

Table 5.4: Low-level functional requirements for provision of continuous optimisation

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAT13</td>
<td>The catalogue must enable the continuous optimisation of the overall V&amp;V and selection process</td>
<td>By analysing the actual deployment information and the collected monitoring data of the deployed VNFs/NS</td>
</tr>
<tr>
<td>CAT14</td>
<td>The catalogue must provide an interface to V&amp;V for triggering new tests based on continuous optimisation outcomes</td>
<td>Automate the V&amp;V testing process</td>
</tr>
<tr>
<td>CAT15</td>
<td>The catalogue must provide an interface to Network infrastructure management for proposing actions regarding QoS enforcement</td>
<td>Supporting information for enforcing QoS guarantees</td>
</tr>
</tbody>
</table>

5.1.1.5 Revision and version control for all stored items

• description: different versions of the same VNF/NS and its metadata should be present in the Catalogue tracking every change in its entire lifecycle and who made them.

• KPIs:

  – stored versions

  – tracked changes

• related pilots: all

• category: optional

• target WP: WP3

Table 5.5: Low-level functional requirements for revision and version control for all stored items

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAT16</td>
<td>The catalogue should store every version of every VNF/NS and track any changes</td>
<td>Previous versions of VNFs/NS will be obtainable</td>
</tr>
<tr>
<td>CAT17</td>
<td>The catalogue should store every version of every update to the metadata of every VNF/NS for all kinds of them(i.e. V&amp;V results, profiling outcomes)</td>
<td>Previous versions of metadata will be obtainable</td>
</tr>
</tbody>
</table>

5.1.1.6 Allow synchronisation with other stores

• description: it is possible for many instances of the 5GTANGO catalogue to exist at the same time. Synchronization between those instances is needed to obtain additional information
(such as VNFs/NS, metadata, selected assets, monitoring data) enriching the catalogue and providing additional data to the continuous optimisation and decision support mechanisms.

- **KPIs:**
  - interfaces to other catalogues for synchronisation
  - synchronised information between different 5GTANGO catalogue instances

- related pilots: all
- category: optional
- target WP: WP3

### Table 5.6: Low-level functional requirements for synchronisation with other stores

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAT18</td>
<td>The catalogue should provide interfaces to other catalogue instances for synchronisation</td>
<td>Instances could belong to the same or different owners</td>
</tr>
</tbody>
</table>

#### 5.1.2 V&V Platform

5GTANGO’s V&V platform is required to perform tests of VNFs/NS and return test results that are digitally signed and include all necessary details. The V&V mechanism is required to take a DevOps approach by setting up an automatic test and validation chain in order to reduce the effort and time to the market for third-party VNFs/NSs.

##### 5.1.2.1 V&V Platform Overview

At a high level, the V&V requirements articulated here describe the following:

1. Incoming tests are supplied with associated Metadata markers. The Metadata markers will be used to help correlate the associated tests that should be run for a particular VNF/NS.

2. Tests are supplied to the V&V along with a set of metadata which describes them. The metadata associated with a test contains two important bits of information, firstly each test has a corresponding test infrastructure requirement. These infrastructure requirements are captured in the meta data associated with a test. These outline the configuration type required to run the test. Secondly the metadata describes the NS and therefore provides guides on what tests should be selected.

3. As customer will have different hosting network environments the V&V is supplied with enough configuration information to allow it to communicate with various test infrastructure providers. This data should also include information of the software test harnesses required to be deployed in the infrastructure.

4. The V&V should produce a set of test results, signed showing which V&V instance operated by which user / company verified the NS. To support the network / service operator the V&V should be able to export the results of a test. These results should be human and machine readable.

5. During the course of executing the tests the V&V will utilise a test infrastructure. The test infrastructure itself is outside the scope of the V&V.
5.1.2.2 V&V Users

It is envisaged that the V&V will support a number of users, these are:

1. The NS developer, who will need to do some preliminary testing of their NS, and potentially recreated tests from other V&V for debugging purposes.

2. The network service operator, who will need to validate a 3rd party VNF/NS in a test infrastructure similar to their own environment, prior to deployment.

3. A third party stand alone V&V service provider, who could offer NS V&V services independently.

5.1.2.3 Cooperation between multiple V&V Instances and Replicating Tests

It should be possible to recreate test environments with ease; this is the set of tests, associated metadata markers, and test infrastructure requirements. To this end, all the information necessary to recreate the test environment should be easily exported from one V&V instance and imported into a second V&V environment. This needs to also include any generated data used during a test.

5.1.2.4 Test, Infrastructure and Network Service Consistency

The environment that Network Services can find themselves in can change rapidly. To cope with this, V&V aims to provide consistency of validation: 1. When a new test is introduced all applicable NS must be flagged for re-testing. 2. When a new NS is introduced all the applicable tests must be selected for testing. 3. When a new infrastructure is provided, or a change to an infrastructure is made all the necessary NS and corresponding tests should be flagged for re-testing.
5.1.2.5 Detailed Requirements

The following more detailed and numbered requirements break down the V&V in more detail and provide the necessary check points to ensure that the resulting V&V design, and implementation reach the goals outlined in the the overview provided here.

5.1.2.6 Test Infrastructure

- **description:** the V&V shall interact with test infrastructure.

- **KPIs:**
  - an interface should exist which allows the caller to configure multiple test environments (virtual infrastructure providers, and the test harness / code expected within it)
  - an interface should exist which allows the caller to assign test environments to specific tests, or sets of tests. For each test or set of test it should be possible to specify the necessary test harness / dummy system components that need to be specified and their configuration

- **related pilots:** all

- **category:** must

- **target WP:** WP3

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV01</td>
<td>Testing infrastructure configuration</td>
<td>It shall be possible to setup, manage and utilize multiple target V&amp;V test infrastructure instances</td>
</tr>
<tr>
<td>VV02</td>
<td>Test infrastructure hardness</td>
<td>It shall be possible to specify the necessary test harness (dummy network elements, etc.) needed for each test, in each testing infrastructure instance</td>
</tr>
</tbody>
</table>

5.1.2.7 Test Management

- **description:** test management refers to test generation, test storage and test metadata management

- **KPIs:**
  - process to verify the correctness of generated tests
  - interface to upload / download / delete / update verified tests
  - interface to upload / download / delete / update (edit) metadata attached to each test (for example, test category, required test infrastructure and configuration)
  - interface to edit metadata associated with each test
  - an interface should exist which allows metadata to be set / removed / added to new / pre-existing test
  - an interface should exist which allows metadata to be set / removed / added to new / pre-existing NS
– an interface should exist which allows the matching criteria between the set of metadata reported by a NS and the metadata set for each test to be set / changed / removed. If removed a default matching logic should be applied

- related pilots: all
- category: must
- target WP: WP3

Table 5.8: Low-level functional requirements for test management

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV03</td>
<td>Test verification</td>
<td>Mechanism or process to verify the tests are correct</td>
</tr>
<tr>
<td>VV04</td>
<td>Test API</td>
<td>A CRUD (Create, Read (download), Update (replace), and Delete) interface for managing tests (executable or textual) via web API</td>
</tr>
<tr>
<td>VV05</td>
<td>Test metadata API</td>
<td>A CRUD (Create, Read (download), Update (replace), and Delete) interface for managing metadata of a test via a web API</td>
</tr>
<tr>
<td>VV06</td>
<td>Export replication data</td>
<td>It shall be possible to export a sub-set (including all) tests and their associated meta. A human and machine interface via web API shall be provided.</td>
</tr>
<tr>
<td>VV07</td>
<td>Test Categorization</td>
<td>It shall be possible to categorize tests according to a dynamic set of criteria for the purposes of appropriate test selection</td>
</tr>
<tr>
<td>VV08</td>
<td>Import replication data</td>
<td>It shall be possible to import a set of tests and their associated meta data. Duplicate tests and conflicting metadata shall be handled by the automatic creation and adoption of name-spacing. A human and machine interface via web API shall be provided.</td>
</tr>
<tr>
<td>VV09</td>
<td>NS Management</td>
<td>There shall be an interface, human and machine, that allows a NS and its associated metadata to be submitted for testing, specifying the set of tests to run, returning the automated set of tests the V&amp;V requires that the NS be verified with, and the platform configuration upon which each test should be executed.</td>
</tr>
</tbody>
</table>

5.1.2.8 Test Execution

- Description: the V&V Platform shall offer the ability to test a NS, this requires automatically selecting tests for execution, manually selecting them, and of course the ability to execute and monitor the tests progress. This functionality is needed to ensure consistency of validation. Additionally, for debugging and issue resolution it must be possible to replicate previously executed tests

- KPIs:
  - the V&V will communicate with an virtual infrastructure provider and standup an virtual infrastructure configuration corresponding to a request to execute a test
    * The necessary networking infrastructure to host an NS shall be provided, including test harness code for expected network infrastructure end points
  - an interface exists which allows the caller to search for tests based on metadata properties.
  - an interface exists which provides notification of test progress, on a regular (timed basis)
  - an interface exists which provides access to the latest metric information for a running test
- an interface exists which allows a running test to be cancelled
- an interface exists which allows a test to be executed, replacing generated data with prerecorded data
- an interface exists which allows a set of test to be executed against a specific NS
- an interface exists which allows a NS to be submitted for testing with no specific tests specified, the V&V will automatically select a set of tests for execution based on the metadata presented with the NS
- an interface exists which allows a customised test plan (test configuration parameters) to be executed with a specified set of tests against a specified NS
- an interface exists which allows the modification of NS metadata, these metadata changes will prompt a re-evaluation of the test set to be executed. The invoker of the metadata change interface will be informed on the new tests which now need to be executed
- an interface exists which allows tests to be modified, removed or added. When a test is changed or added via this API, a re-evaluation of all known NS and their metadata should occur and the NS recorded shall be automatically scheduled for execution

- related pilots: all
- category: Must
- target WP: W3

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV10</td>
<td>Test configuration</td>
<td>It shall be possible to configure the test with actual information of the SUT (i.e. endpoint address; target performance value)</td>
</tr>
<tr>
<td>VV11</td>
<td>Test search and selection</td>
<td>It shall be possible to search for tests based on metadata, and compose a test suite using selected tests</td>
</tr>
<tr>
<td>VV12</td>
<td>Testbed configuration</td>
<td>It shall be possible to configure the virtual test infrastructure in terms of allocated resources and available testing components</td>
</tr>
<tr>
<td>VV13</td>
<td>Dynamic test execution</td>
<td>It shall be possible to automatically re-execute test sets if a given set of criteria changes (for consistency of validation)</td>
</tr>
<tr>
<td>VV14</td>
<td>Selective Test execution</td>
<td>It shall be possible to execute a customized test plan from a given set of tests</td>
</tr>
<tr>
<td>VV15</td>
<td>Selective Test execution with prerecorded data</td>
<td>It shall be possible to execute a customized test plan from a given set of tests, with a set of re-recorded test data from a previous test run, allowing tests to be replicated.</td>
</tr>
<tr>
<td>VV16</td>
<td>Test Execution Interfaces</td>
<td>It shall be possible to search for, invoke, monitor and cancel tests</td>
</tr>
<tr>
<td>VV17</td>
<td>Test Execution Operations</td>
<td>The V&amp;V shall be able to communicate with a test infrastructure provider to request the necessary infrastructure resources for NS testing</td>
</tr>
</tbody>
</table>

5.1.2.9 Test Execution Monitoring

- Description: the V&V Platform should be able to instrument and collect the resulting metrics from any tests run
• KPIs:
  – new metrics specified in metadata and published by SUT and collected and present in recorded metrics by V&V
  – interface exists which provides access to recorded metrics captured by a system under test
  – a SUT which does not implement the probe endpoint will be rejected by the V&V and an error report provided
  – a documented metric aggregation algorithm which produced a star metric summation
  – a star metric is available for any set of combined test results, including a set of one result
  – metadata specifying a test infrastructure configuration results is reflected in the corresponding test infrastructure hosting the necessary virtual test infrastructure being instantiated and configured to host the SUT. This information is reported as part of the test results
  – data generated automatically as part of a test run and pushed through the NS is recorded and is available to download via an interface after a test run as completed

• related pilots: all

• category: must

• target WP: W3

Table 5.10: Low-level functional requirements for test monitoring

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV18</td>
<td>User Defined Metrics</td>
<td>The V&amp;V developer shall use the NS metadata to specify additional metrics. These metrics should be published by the NS, and once specified, they will be collected by the V&amp;V.</td>
</tr>
<tr>
<td>VV19</td>
<td>Test result and Metric Aggregation</td>
<td>The V&amp;V Platform shall have a method for aggregating metrics against KPIs</td>
</tr>
<tr>
<td>VV20</td>
<td>Test result and Metric metadata</td>
<td>It shall be possible to add metadata, such as testing environment configuration, to the metric data</td>
</tr>
<tr>
<td>VV21</td>
<td>Probe point in SUT</td>
<td>SUT (VNF or NS) shall implement required probe point enabling the monitoring</td>
</tr>
<tr>
<td>VV22</td>
<td>Test result star level compilation</td>
<td>The V&amp;V platform shall have a method for compiling a “star” value as an abstraction of test results to give an overall confidence level</td>
</tr>
<tr>
<td>VV23</td>
<td>Generated test data collection</td>
<td>The V&amp;V platform shall have a method to record and store any generated test data used during a test run. This will allow the test to be replicated at another time</td>
</tr>
</tbody>
</table>

5.1.2.10 Test Results

• description: the V&V Platform should be able to provide notification when test runs complete, and also provide access to the results of tests which have been executed

• KPIs:
  – able to download test results
  – web page listing test results
– digitally signed download of test results
– machine interface which provides notification when a test is complete, access to download the results, which should be signed

• related pilots: all
• category: must
• target WP: 3

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV24</td>
<td>Test Result Views</td>
<td>It shall be possible to present and view both raw and aggregated test results</td>
</tr>
<tr>
<td>VV25</td>
<td>Test Result Download</td>
<td>It shall be possible to download the raw, aggregate, or both test results</td>
</tr>
<tr>
<td>VV26</td>
<td>Test Result Digital Sign off</td>
<td>It shall be possible to obtain sign off the test results as a method of authentication and validation and as a prerequisite for upload to the catalogue</td>
</tr>
<tr>
<td>VV27</td>
<td>Test Result Notification</td>
<td>It shall be possible to obtain a web based notification when a test run has completed, and new results are available.</td>
</tr>
<tr>
<td>VV28</td>
<td>Test Result Interfaces</td>
<td>The functionality (VVTR01-04) shall be made available to both a human operator, and via an API or other interface allowing integration with other systems</td>
</tr>
</tbody>
</table>

### 5.1.3 NFV-enabled service development kit (SDK)

The Service Development Kit in 5GTANGO will assist the developer in composing network functions, services, and associated tests. The 5GTANGO SDK will further extend upon the functionalities of the SONATA SDK. Whereas the latter focuses on baseline composition, packaging and evaluation of NFs into NFV services using the SONATA editor as well as its associated emulator, the 5GTANGO SDK will extend the support for: i) test development, ii) performance characterisation, and iii) specific manager support. Each of these will be discussed in more detail below. The requirement list in this document might be updated later in the project given the input of pilots and experiences in other work packages.

#### 5.1.3.1 Development of service and component tests

- Description: a DevOps approach must guarantee correct functioning of the service and its components. This involves support for the development tests at both levels. Tests which can be executed at the local developer’s laptop, as well as on external platforms such as the V&V store.

- KPIs: Total number of tests, number of successfully executed tests, number of test platforms supported, re-usability of tests

- related pilots: all
- category: must
- target WP: WP4
Table 5.12: Low-level functional requirements for development of service and component tests

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDK01</td>
<td>The SDK must support the development of unit tests of individual Network Functions</td>
<td>This involves pre- and post-deployment tests (runtime).</td>
</tr>
<tr>
<td>SDK02</td>
<td>The SDK must support the development of tests of an entire service composed of individual Network Functions</td>
<td>This involves pre- and post-deployment tests (runtime).</td>
</tr>
<tr>
<td>SDK03</td>
<td>The SDK must support integration testing with (given releases of) the Service Platform. Developers of services or NFs in the SDK should be able to write tests to validate if the impact of deploying their component(s) on given release of the SP.</td>
<td>A common test format should be supported, such that tests can be re-used by the V&amp;V store.</td>
</tr>
<tr>
<td>SDK04</td>
<td>The SDK must support the development of tests which can be executed by the V&amp;V store.</td>
<td></td>
</tr>
</tbody>
</table>

5.1.3.2 Development of performance characterisation tools

- description: NFV-services are largely based on software components. These introduce uncertainties in the resulting performance. The SDK will provide tools to characterise the performance of developed components under different circumstances.
- KPIs: number of performance tests, number of supported performance metrics, performance intra/extrapolation techniques
- related pilots: selected pilots
- category: must
- target WP: WP4

Table 5.13: Low-level functional requirements for development of performance characterisation tools

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDK05</td>
<td>The SDK must support the definition of a performance profile of individual NFs and services.</td>
<td>A profile should define the metrics that can be measured during the performance recording.</td>
</tr>
<tr>
<td>SDK06</td>
<td>The SDK must enable performance recording after deployment of the components and service.</td>
<td>Ideally, the performance recorder can be executed on multiple platforms, not only locally on the laptop of the developer, but also on the V&amp;V store, as well as on the Service Platform.</td>
</tr>
<tr>
<td>SDK07</td>
<td>The SDK should support the design of performance intra/extrapolation techniques based on limited performance profile recordings.</td>
<td>The goal of intra/extrapolation techniques is to assist the developer in the development of scaling mechanisms despite limited performance data being available.</td>
</tr>
<tr>
<td>SDK08</td>
<td>The SDK should support performance bottleneck detection in NFs and services.</td>
<td>Performance bottleneck refer to those parts of the service or component which limit overall performance.</td>
</tr>
</tbody>
</table>

5.1.3.3 Service- and Function Specific Manager development support

- description: one of the main innovations of the SONATA platform is the support for specific manager functionality, both at the service, as well as at the NF-level. The 5GTANGO SDK will further extend the SONATA SDK to develop SSMs and FSM which are able to provide high certainty in the performance, availability and consistency of the resulting runtime instances under a range of conditions.
- KPI: SM patterns for service resiliency, SM patterns for scaling, state migration support
mechanisms

- related pilots: selected pilots
- category: must
- target WP: WP4

Table 5.14: Low-level functional requirements for development of performance characterisation tools

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDK09</td>
<td>The SDK must support the design of FSMs and SSMs providing resiliency against single or multiple component failures.</td>
<td>This involves interaction with: i) monitoring functionality to detect liveness of involved components, ii) the service platform to dynamically instantiate new service components. The proposed SSMs should be as re-usable as possible.</td>
</tr>
<tr>
<td>SDK10</td>
<td>The SDK must support the development of mechanisms to identify, and migrate service- or function-related state under dynamic conditions.</td>
<td>Dynamic conditions could involve: i) failure conditions, requiring the service and associated state to be replicated, ii) scaling conditions, and iii) new software updates available to already instantiated components.</td>
</tr>
<tr>
<td>SDK11</td>
<td>The SDK must support the design of scaling SMs supporting dynamic control loops</td>
<td>This involves support for designing re-usable load balancing mechanisms relying on dynamic interaction with the Service Platform to instantiate additional/reduced number of instances.</td>
</tr>
</tbody>
</table>

5.1.4 Extended service platform to support vertical applications

5GTANGO’s service platform is required to support the particular needs of vertical applications, including initial placement and orchestration aspects.

A set of reactive characteristics have to be supported with regards to management of VNFs and NSs, targeting at their dynamic management and re-configuration (where required) for achieving objectives defined by communication providers (e.g. high performance, energy efficiency).

For example, we intend to support oversubscription vs. guaranteed resources (selectable per slice) and the particular sharing characteristics of a manufacturing use case (machinery cannot be time-shared). 5GTANGO will also emphasise on autonomicity, dependability and resilience, e.g., by introducing automatic restarts for plugins.

Autonomic management of network functions and services can be realised through the specification of set a of decision making mechanisms based on monitoring a set of parameters through monitoring feeds. Decision making functionalities can be embedded within the software or triggered through real-time policies enforcement.

This list is a first take on the requirements, which will be detailed later, as we learn more about the Pilots. This list is an extension of the requirements we have gathered in other projects, such as SONATA (see [51]).

5.1.4.1 Support the complete DevOps lifecycle for NS/VNFs, including NS/VNF on-boarding, scaling, (re-) configuration, and updating

- description: the iterative and agile nature of DevOps imply that network services and virtual network functions can be automatically on-boarded on the Service Platform’s Catalogues, fetched for service instantiation/deployment, (re-)configured, monitored and scaled or updated if needed;
- KPIs: Total number of NSs/VNFs successfully on-boarded, instantiated, monitored, scaled, re-configured and updated (two full cycles, demonstrating the DevOps concept of iteratively building the final result)

- related pilots: all

- category: must

- target WP: WP5

This high level requirement can be decomposed into the following lower level requirements.

Table 5.15: Low-level requirements for Support the complete DevOps lifecycle for NS/VNFs

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP01</td>
<td>The Service Platform must be able to on-board descriptions of Network Services (NSs), composed of one or more Virtual Network Functions (VNFs), Physical Network Functions (PNFs) and their inter-connections.</td>
<td>‘On-boarding’ here means accepting the content of the descriptors, by making some syntactic/semantic validations, even considering that these descriptors may come from the SDK and V&amp;V platform.</td>
</tr>
<tr>
<td>SP02</td>
<td>The Service Platform must be able to store valid NSs/VNFs/PNFs descriptors in its internal Catalogue.</td>
<td>This includes overall life-cycle management of every asset, e.g., dealing with multiple versions of the description of a service/function, refusing to delete descriptors that have instances running.</td>
</tr>
<tr>
<td>SP03</td>
<td>The Service Platform must be able to store the VM images implementing the VNFs.</td>
<td>These images come either included or referenced in the VNF descriptors. This is crucial both from a performance perspective and for multi-PoP environments, where the exact PoP where the function is to be deployed is not known at on-boarding time.</td>
</tr>
<tr>
<td>SP04</td>
<td>The Service Platform must be able to fetch a service descriptor from the Catalogues.</td>
<td>This includes fetching for multiple criteria.</td>
</tr>
<tr>
<td>SP05</td>
<td>The Service Platform must be able to instantiate/deploy a service from its descriptor.</td>
<td>Service instantiation is currently a manual process, but more for demo-ability reasons, and not technical reasons. It must result from the need to update the service instance, after an update of it’s descriptors.</td>
</tr>
<tr>
<td>SP06</td>
<td>The Service Platform must be able to (re-)configure a service.</td>
<td>(Re-)Configuring a service includes configuring its functions and connections between them. The simplest case is to configure these before starting the instance, but configuration when the service is already running might be addressed as well.</td>
</tr>
<tr>
<td>SP07</td>
<td>The Service Platform must be able to start a service.</td>
<td>Currently, the service instance starts automatically after instantiation, but separating both events is more realistic.</td>
</tr>
<tr>
<td>SP08</td>
<td>The Service Platform must be able to monitor a service instance.</td>
<td>Monitoring parameters depend on the service and functions, so they should be defined in their descriptors. These monitoring parameters should be added dynamically.</td>
</tr>
<tr>
<td>SP09</td>
<td>The Service Platform must be able to scale a service instance.</td>
<td>Scaling at the VIM level is currently easily achievable. Scaling at the service level is complex and has to be worked. The mid-level, to scale at the VNF level is becoming common.</td>
</tr>
</tbody>
</table>
#ID | Description | Comments
--- | --- | ---
SP10 | The Service Platform must be able to accept Service/Function Specific Managers concept (see [52]) | The idea of having the possibility of changing the default behaviour of the platform, without compromising security, availability etc., gives the developer/platform owner the possibility of treating the edge cases, which might be the competitive advantage that is needed to differentiate from the competition in terms of disruptive services. The mechanism can be expanded to other aspects of the platform, e.g., Resource Allocation Conflict Resolution.
SP11 | The Service Platform must be able to monitor resources usage of a VNF or a service instance. | Monitoring information may be collected from the VNFs, the resource managers or network controllers.
SP12 | The Service Platform must be able to process sets of collected monitoring data per network service or VNF and extract profiling information. | A set of data mining and analysis process can be supported for profiling (near real-time triggering of events like scaling (e.g.) can be supported by the Service/Function Specific Managers concept).

5.1.4.2 Support the virtualization of available vertical resources

- description: Smart controlled resources (e.g., manufacturing machinery, cameras) need to be virtualizable, in the sense that can be shared among different tenants and applications in order to be consumed or actuated upon them. When the resource virtualization is not possible, a descriptor should be generated

- KPI: Virtualized vertical resources
- related pilots: Smart manufacturing, Immersive media
- category: must
- target WP: WP5

Table 5.16: Low-level requirements for Support the virtualization

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP13</td>
<td>The Service Platform must be able to deal with Physical Network Functions (PNFs).</td>
<td>Work is needed in defining a common API to support these PNFs, namely to enable them to enter the existing workflow.</td>
</tr>
</tbody>
</table>

5.1.4.3 Support of Network Slicing considering resource allocation, service contextualization, and monitoring.

- description: the usual processes available in a Service Platform will have to be slice-adapted, meaning that resources to be allocated should be restricted to a given slice, taking into account the service context, resource availability, etc. Monitoring data will have to reflect slice boarders as well, reflecting the intended isolation of resource allocation

- KPIs: System utilization, bandwidth sharing, service availability
- related pilots: all
Table 5.17: Low-level requirements for support of network slicing

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP14</td>
<td>The Service Platform must be able to on-board slice descriptions.</td>
<td>These descriptions should be in the form of NS descriptors or VNF descriptors, and not made interactively (e.g., through a Graphical User Interface) by a human.</td>
</tr>
<tr>
<td>SP15</td>
<td>The Service Platform must be able to allocate resources within a specific slice.</td>
<td>We can consider that currently there is only one slice, with all the resources.</td>
</tr>
</tbody>
</table>

5.1.4.4 Support of carrier-grade functionalities for highly reliable and large scale deployments.

- description: high reliability and large scale deployments, in which most of failures and needs can be recovered automatically (i.e., without human intervention), are crucial both for a service platform to be considered carrier-grade and for network services deployed in it to share such high quality label. This implies a significant number of dependencies that must be addressed since the beginning of the project

- KPIs:
  - Mean Time Between Failures (MTBF, both of the platform and of services deployed in it)
  - Mean Time To Recover (MTTR, both of the platform and of services deployed in it, though the later strongly depends on the kind of service provided, it’s architecture, etc.)

- related pilots: Immersive media, Real time communications

- category: must

- target WP: WP5

5.2 Non-Functional requirements

Non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system. The following non-functional requirements have been considered for the 5GTANGO four main functional blocks:

- Scalability: capability of a system, network, or process to handle a growing amount of work, or its potential to be enlarged to accommodate that growth

- Customizability: ability of a piece of software to be customized to suit a particular need

- Security: is the protection of the system from the theft or damage to their hardware, software or information

- Reliability: the ability of a system or component to function under stated conditions for a specified period of time
• Availability: the ratio of the total time a functional unit is capable of being used during a given interval to the length of the interval

• Authentication: is the process of actually confirming the identity

• Confidentiality: is the non-disclosure of information except to another authorized person

5.2.1 Catalogue

This sub-section covers the non-functional requirements for the Catalogue. These requirements are:

5.2.1.1 Scalability

• description: the Catalogue should be able to support an increasingly large number of VNFs and NSs and their respective metadata

• KPIs:
  – consistent data stored across all instances of the scalable catalogue
  – minimum performance overhead due to scaling

• related pilots: All

• category: must

• target WP: WP3

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAT19</td>
<td>The catalogue must be able to scale based on the stored items</td>
<td>Expand or shrink based on the number of items</td>
</tr>
</tbody>
</table>

5.2.1.2 Security

• description: the Catalogue should be able to support a range of security mechanisms that assure that the items stored are not accessed or read by unauthorized entities. Enabling authentication, authorization, data protection and confidentiality is a common requirement together with the Service Platform and V&V platform

• KPIs: authentication/authorization mechanisms supported, Security controls in place

• related pilots: all

• category: must

• target WP: WP3
5.2.1.3 Availability

- description: the Catalogue should be available whenever it is being used.
- KPI: “Near-zero” downtime
- related pilots: all
- category: must
- target WP: WP3

5.2.2 V&V Platform

This sub-section covers the non-functional requirements for the V&V platform. These relate only to the non-functional requirements of the V&V platform. Note that the Test Infrastructure is out of scope of the V&V. This section will concern all the elements necessary to process tests, metadata, and the results of the tests themselves, but it will not concern the mechanics of actually executing those tests, beyond the necessary Test Infrastructure interfaces required to drive the tests.

5.2.2.1 Platform Configuration and Deployment Scenarios

The non-functional requirements of the V&V platform are derived from the anticipated deployment scenarios. Three scenarios are anticipated in which the V&V platform is deployed as part of a:

1. Development platform: As a development platform, the V&V will be deployed in a small scale in a development environment (i.e. workstation / small server environment).

2. A validation platform for a mobile operator: As a service wide validation platform, the V&V will be deployed to enable the execution of a number of validation tests for new network services, which will be deployed across the network. Additionally the service will need to support higher levels of security, and availability.

3. A 3rd party V&V platform: As a platform provisioned by 3rd parties, V&V will be offered as a service, providing signed test results to customers. Different 3rd party platforms can focus on different aspects and provide different features (e.g., different infrastructure and tests). Again, a high level of security and availability are essential.
The V&V will need to scale from being deployed within a development platform (1, above) to the 3rd party V&V Platform configuration (3, above). Indeed differing levels of performance, and of security behaviour are required in both configurations.

The requirements that are elicited for the above scenarios are:

### 5.2.2.2 Scalability

- **description:** the V&V platform must be able to grow and shrink regarding to the requirements of resource allocation to a given test suite (i.e a collection of test upon a SUT)

- **KPIs:**
  - it should be possible to submit 10 NS for testing to the V&V every second
  - the V&V should respond to all network interface requests in under 500. The V&V can respond with error responses, e.g. HTTP 502 is acceptable. Timeout’s are not acceptable
  - it should be possible to observer the Test Infrastructure running multiple tests concurrently

- **related pilots:** all
- **category:** optional
- **target WP:** WP3

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV29</td>
<td>The V&amp;V platform may be able to run multiple</td>
<td>Parallel execution of tests would speed up the number of NS which can be</td>
</tr>
<tr>
<td></td>
<td>tests concurrently</td>
<td>validated, either by running multiple instances of a single NS with</td>
</tr>
<tr>
<td></td>
<td></td>
<td>different tests, or alternatively, by running different NS and different</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tests in parallel.</td>
</tr>
<tr>
<td>VV30</td>
<td>The V&amp;V platform should try to scale to handling</td>
<td>It is important that while the time for test execution to complete is</td>
</tr>
<tr>
<td></td>
<td>100s of requests per second</td>
<td>beyond the scope of the V&amp;V that the tool should remain responsive and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>provide feedback on progress. This should include submitting 10s of NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>for testing to the V&amp;V every second, polling for progress, notifications</td>
</tr>
<tr>
<td></td>
<td></td>
<td>of progress etc.</td>
</tr>
<tr>
<td>VV31</td>
<td>The V&amp;V platform should respond to all exposed</td>
<td>The V&amp;V platform should respond to all requests with a max of 500ms. Even</td>
</tr>
<tr>
<td></td>
<td>network interface requests</td>
<td>if that is to return an error response</td>
</tr>
</tbody>
</table>

### 5.2.2.3 Customisation

- **description:** customisation is crucial for the V&V platform in terms of validation/verification capabilities regarding to diverse SLA/policies/NS functional requirements

- **KPI:**
  - automated deployment recipes available for all the scenarios defined in [@sec: platform-conf()]
– a developer should be able to setup the V&V on a single machine, with a set of default tests, using a single command line invocation. The developer may be asked for more information via an installation wizard / script or other such tool. This installation and configuration should take no more than 20 minutes

– A service provider should be able to easily provision and configure the V&V on multiple machines; for 10 machines, process should take less than 2 hours

• related pilots: all
• category: must
• target WPs: WP3, WP4, WP5, WP6

Table 5.22: Low-level non-functional requirements for V&V platform - customisation

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV32</td>
<td>V&amp;V to support developers</td>
<td>V&amp;V should be able to be easily deployed and used by developers. This deployment and configuration should be quick, with little manual installation effort, allowing the developer to spend their time creating the NS and not worrying about maintenance or configuration of the V&amp;V. The developer can guide the installation via various configuration options.</td>
</tr>
<tr>
<td>VV33</td>
<td>V&amp;V to support service providers</td>
<td>V&amp;V should support service providers, and be easily configured as a long running highly available service, with strong security settings. In this role it will act as vetting tool for NS before they are allowed into the service providers catalogue</td>
</tr>
</tbody>
</table>

5.2.2.4 Availability / Reliability of the V&V

• description: the V&V platform should be highly available and reliable

• KPIs:

  – [must] it should be possible to manually repeat a test run and obtain the same result as that produced by the V&V. The V&V should provide enough information to allow a user to do this

  – [optional] if the V&V is deployed across 5 servers, it should be possible to pull the network connectivity / power supply from 1 server and still have the V&V function correctly, with no data loss

  – [optional] a single command line tool should be able to update the V&V ensuring that the system remains available, with no down time

  – [optional] a regular report from the active testing probes should be recorded in a system log

  – [must] upon startup the system should log and (optionally) display the status of each of its components, and of the external components, including the Test Infrastructure upon which it relies

  – [optional] the system log should contain information on the status of each of its components, and of the external components, including the Test Infrastructure upon which it relies
– \([\textit{optional}]\) the V&V can replicate data to an external data repository, with at most a 10 minute lag

• related pilots: all

• category: must and optional (indicated in table and in \([\textit{italics}]\) in the KPI descriptions above)

• Target WPs: WP3, WP5

**Table 5.23: Low-level non-functional requirements for V&V platform - Availability/ Reliability**

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Category</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV34</td>
<td>The testing result from the V&amp;V platform must be reliable</td>
<td>Must</td>
<td>It should be possible to replicate a test, using the information available from the V&amp;V and achieve the same results.</td>
</tr>
<tr>
<td>VV35</td>
<td>The V&amp;V should be configured for high availability</td>
<td>Optional</td>
<td>As a service provider configuration the V&amp;V must be highly available, it should be able to withstand 20% of the hardware it is deployed across being lost and still function.</td>
</tr>
<tr>
<td>VV36</td>
<td>The V&amp;V platform must support live updates/upgrades</td>
<td>Optional</td>
<td>It should be possible to upgrade the V&amp;V without down time</td>
</tr>
<tr>
<td>VV37</td>
<td>The V&amp;V should support active testing of its own systems.</td>
<td>Optional</td>
<td>The V&amp;V should support active testing; V&amp;V supplied external probes should regularly invoke the V&amp;V network interfaces, pushing a workload through the V&amp;V and monitoring the V&amp;V’s responses. This ensures that the V&amp;V is actually working. The results of these probe runs should be logged.</td>
</tr>
<tr>
<td>VV38</td>
<td>Startup system check.</td>
<td>Must</td>
<td>When the system starts up it should validate each of its internal components are functioning. Any external components should also be validated. A report should be produced (logged) for reference. On system errors enough information should be exposed to allow an operator to correct the issue. If the V&amp;V is started at the command line then the console should include information about the status of the V&amp;V system.</td>
</tr>
<tr>
<td>VV39</td>
<td>The system check run at startup should be repeated periodically (configurable) during the runtime of the V&amp;V</td>
<td>Optional</td>
<td>The startup check should be repeated by the V&amp;V on a regular, configurable basis.</td>
</tr>
<tr>
<td>VV40</td>
<td>Data replication</td>
<td>Optional</td>
<td>Data held by the V&amp;V should be replicated on an external system, ensuring that no more than the last 10 minutes of data (at most) can be lost due to a complete system failure.</td>
</tr>
</tbody>
</table>

5.2.2.5 Security

• description: security is crucial for V&V platform in terms of testing result credibility and privacy. The following Authentication and Confidentiality are aspects of security

• KPIs:
  – a Thread model should be produced
  – details of non conforming / unexpected data supplied to the V&V should be logged and recorded.
  – tests which exceed the specified execution time should be killed and details logged
– it shall be impossible for any code executing within the Test Infrastructure to access public (internet) IP addresses, IPv4 & IPv6

– a specific log of all user account creation should be created

• related pilots: all

• category: must

• target WP: WP3

In line with industry, best practice a threat model should be produced which details how the development team behind the V&V anticipates attacks and their identified solutions to address it.

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV41</td>
<td>Threat Model Creation</td>
<td>The V&amp;V should be provided with a threat model which details the assets, attack vectors and mitigation steps.</td>
</tr>
</tbody>
</table>

The creation of a threat model is only possible once a detailed design of the V&V has been produced. However based on the overview of the V&V and the functional requirements that have already been identified, the following commonly implemented mitigation steps can be identified.

### Table 5.25: Low-level non-functional requirements for V&V platform - security

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV42</td>
<td>Vet all incoming data</td>
<td>Must assume that all incoming data from every interface is an attempted attack. This data must be checked and formatted (escaped) such that SQL injection, HTML, and JS injection etc. cannot occur. Details about bad data should be logged.</td>
</tr>
<tr>
<td>VV43</td>
<td>The V&amp;V Platform must be able to support the most up-to-date standards and recommendations in terms of security.</td>
<td>This implies namely the implementation of automatic updates on rules and check list verification that verify known security breaches.</td>
</tr>
<tr>
<td>VV44</td>
<td>Max test runtime</td>
<td>NS submitted for execution could be rouge code. A Max execution time for any test should be specified allowing the V&amp;V to kill tests which over run. Details should be logged.</td>
</tr>
<tr>
<td>VV45</td>
<td>NS Test Sandbox</td>
<td>The NS submitted for testing is untrusted. The sandbox in which this code is executed should be secure. There should be no access to the internet for any hosted SUT</td>
</tr>
<tr>
<td>VV46</td>
<td>Must support user/system auditing</td>
<td>An audit trail of user creation, usage etc should be created in the log system. This should not contain any personal information, and should be recorded in a GDPR compliant way.</td>
</tr>
</tbody>
</table>

### 5.2.2.6 Logging and Reporting

• description: obtaining accurate information about the running of a service is important to those who need to support the service. It helps understand how the service is being used, and how it operates in various situations. It also helps aid with V&V system bug / reproduction of issues / resolution and detection of such

• KPIs:
  
  – the host OS logging mechanism should contain information about the current system status
the host OS logging mechanism should contain information about how the system is being used, including call logs throughout the various system components

- related pilots: all
- category: must
- target WP: WP3

Table 5.26: Low-level non-functional requirements for V&V platform - system reporting

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV47</td>
<td>Use the existing system logging mechanism</td>
<td>Where possible the host operating system logging system should be used</td>
</tr>
<tr>
<td>VV48</td>
<td>The V&amp;V should log access</td>
<td>To support operational maintenance the V&amp;V should produce logging explaining its operation</td>
</tr>
<tr>
<td>VV49</td>
<td>The V&amp;V should record call logs</td>
<td>Call logs, from user requests through to back end logic should be recorded allowing an operator of the service to trace calls through the system</td>
</tr>
</tbody>
</table>

5.2.2.7 Authentication and User Management

- description: authenticate the identity of the platform is the first step to implement security
- KPIs:
  - [**must**] it should be possible to use the TANGO SSO, when using the human and machine interfaces for V&V
  - [**optional**] an OAuth and V&V Key issuing service should allow users to register and obtain user keys for the system
  - [**Must**] the V&V should ensure that users without admin or Test Suite roles can only submit NS for testing
  - [**Must**] the V&V should ensure that Admin users can add tests, submit NS, and access and change all other user account information
  - [**Must**] the V&V should ensure that users with Test Suites can submit NS for testing and add, remove, edit only the tests that they have approved access to

- related pilots: all
- category: must and optional (indicated in table and in [*italics*] in the KPI descriptions above)
- Target WPs: WP3, WP5

Table 5.27: Low-level non-functional requirements for V&V platform - Authentication and User Management

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Category</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV50</td>
<td>The V&amp;V Authentication</td>
<td>Must</td>
<td>The V&amp;V platform should use the SSO of the TANGO service platform</td>
</tr>
<tr>
<td>VV51</td>
<td>The V&amp;V OAuth</td>
<td>Optional</td>
<td>The V&amp;V, when deployed as a stand alone service should support APIs invoked via a V&amp;V issued OAuth credential</td>
</tr>
</tbody>
</table>
### 5.2.2.8 Confidentiality

- **description:** the confidentiality and data protection is crucial for the users of the V&V platform

- **KPIs:**
  - user 1, shall not be able to obtain information about NS submitted for testing by User 2
  - the admin user shall have access to all user account information, and should be able to edit this, removing users, adding users and replacing them
  - a person fulfilling the role of the GDPR’s Data Protection Officer should be able to obtain all the necessary data about personal user account information stored and processes by the V&V

- **related pilots:** all

- **category:** must

- **target WPs:** WP3, WP5

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Category</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>VV54</td>
<td>EU GDPR (General Data Protection Regulation) Compliance</td>
<td>Must</td>
<td>The system should not record, in logs or otherwise more personal information than necessary about system users</td>
</tr>
<tr>
<td>VV55</td>
<td>User access information</td>
<td>Must</td>
<td>The admin user should have access to all information on the system. This is a privileged role. Users should not be able to see details of tests, NS or other information submitted by other users, unless these users have given permission for this information to be viewable.</td>
</tr>
</tbody>
</table>

### 5.2.3 NFV-enabled software development kit

This sub-section covers the non-functional requirements for the NFV-enabled SDK. As the SDK is mainly a set of tools assisting the developer, some of the requirements below, only apply to particular components of the SDK, such as the emulator.

#### 5.2.3.1 Scalability

- **description:** as long as the underlying hardware on which the SDK is deployed is sufficiently capable to run and test the developed components in the confined setting, the SDK should
not be the limiting factor in developing NFV-based services of a given scale

- **KPIs:** overhead of the SDK tools and environment in terms of used disk, memory and cpu resources compared to bare-metal running of the developed service(s)

- **related pilots:** all

- **category:** optional

- **target WP:** WP4

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDK12</td>
<td>The resource overhead of the SDK in developing and test-running services and components should be negligible compared to the resources required by the service(s) itself.</td>
<td></td>
</tr>
</tbody>
</table>

### 5.2.3.2 Customizability

- **description:** the SDK consists of a range of tools which can be used in different workflows, and possibly on different target platforms, depending on the needs of the developer for the focused service or component. In that sense the SDK should be customisable to the needs of the developer

- **KPIs:** number of supported workflows, number of supported target platforms

- **related pilots:** all

- **category:** optional

- **target WP:** WP4

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDK13</td>
<td>The given set of SDK tools should be used in a range of different workflows, were the use of some tools is mandatory, and others are conditional and up to the preference of the developer.</td>
<td></td>
</tr>
<tr>
<td>SDK14</td>
<td>The SDK should not be limited to a single Service/MANO Platform, and will at least have architectural support to enable other platforms in the future.</td>
<td></td>
</tr>
</tbody>
</table>

### 5.2.3.3 Security

- **description:** the SDK should support a range of security mechanisms enabling authentication, authorization of developers and signing of developed components. This is a common requirement together with the Service Platform and V&V platform

- **KPIs:** authentication/authorization mechanisms supported, signature mechanisms supported
5.2.3.4 Reliability

- **description**: the behaviour of the SDK tools must be reproducible and deterministic. Under the same circumstances, the editing, packaging, emulation, testing and profiling functionality must produce the same or very similar outputs.
- **KPI**: reproducibility of SDK tool output
- **related pilots**: all
- **category**: must
- **target WP**: WP4

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDK19</td>
<td>The SDK must have low variance in execution time and produced output when repeating multiple executions on the same input.</td>
<td>Packaging, validation and profiling tools should be predictable in their execution time.</td>
</tr>
</tbody>
</table>

5.2.3.5 Availability & serviceability

- **description**: as the SDK does not provide an operational service, but is rather a set of tools which can be used in an ad-hoc manner by the developer, they have no strong availability requirements, apart from being usable if the developers computing environment has adequate hardware resources, and fulfils the required software dependencies. Nevertheless, the SDK can be pre-packaged in such a way that dependencies on the local environment are minimal (e.g., providing SDK VM, Vagrant or Docker environment)
- **KPI**: hardware and software requirements of the SDK
- **related pilots**: all
- **category**: must
• target WP: WP4

Table 5.33: Low-level non-functional requirements for NFV-enabled SDK - availability & service-ability

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDK19</td>
<td>The number of dependencies on the execution environment of the SDK must be low in terms of software and hardware.</td>
<td></td>
</tr>
</tbody>
</table>

5.2.4 Extended service platform to support vertical applications

This sub-section covers the non-functional requirements for the Service Platform. These requirements are:

5.2.4.1 Scalability

• description: the Service Platform itself must be able to grow and shrink with its workload, thus minimising OpEx, as well as supporting the scalability of the services and functions on-boarded and instantiated in it. Specially valuable if the capability of starting small, to support quick proof-of-concept deployments

• KPIs:
  – CPU load (considering multiple CPUs as well)
  – Number of requests/sec
  – Number of supporting containers along time

• related pilots: all

• category: must

• target WP: WP5

Table 5.34: Low-level non-functional requirements for service platform - scalability

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP16</td>
<td>The Service Platform must be able to automatically scale out and in, according to load.</td>
<td>This implies using the same mechanisms that the services deployed on it use, based on monitoring data and alarms. Limits must be configurable by the Service Platform’s owner. ‘Starting small’ may imply leaving some modules out of a certain deployment.</td>
</tr>
</tbody>
</table>

5.2.4.2 Customizability

• description: the customizability achieved with the usage of a micro-services based architecture in SONATA may be pushed further by improving the capacity to dynamically switch between different micro-services implementing the same functionality. Furthermore, the ability to be deployed without important but complementary modules in certain scenarios (e.g., User Management or KPIs Management, in a local deployment)
• KPI: number of interchangeable micro-services
• related pilots: all
• category: must
• target WP: WP5

Table 5.35: Low-level non-functional requirements for service platform - customizability

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP17</td>
<td>The Service Platform must be able to have the micro-services it is built upon switched off or exchanged with others providing the same features.</td>
<td>This implies the implementation of this ‘on-boarding’ and ‘feature toggling’ mechanism.</td>
</tr>
</tbody>
</table>

5.2.4.3 Security

• Description: security is a broad scoped non-functional requirement, touching the Service Platform deep and wide. **Authentication**, a special case of Security, is key in guaranteeing that resources be allocated only to those to whom they should be allocated. Knowing every user is today crucial in every opened system, and the Service Platform is no exception. Besides Authentication, security deals with aspects like **authorization** (who can do what), **data protection** and **confidentiality** (see also below), etc.

• KPIs:
  – number of registered users along time
  – number of logged-in users along time
  – number of requests rejected due to rate limitation/user and along time

• related pilots: all
• category: must
• target WP: WP5

Table 5.36: Low-level non-functional requirements for service platform - security

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP18</td>
<td>The Service Platform must be able to support the most up-to-date standards and recommendations in terms of security.</td>
<td>This implies namely the implementation of automatic updates on rules and check list verifications that verify known security breaches.</td>
</tr>
</tbody>
</table>

5.2.4.4 Reliability

• description: to implement a platform with high reliability, giving platform owners a sound base for its business, is key in the success of our project. Guaranteeing a reliable platform as open as ours is a challenge, given that services can be provided by ‘any’ (see above **Security**) developer, no matter how complex and prone to failure it is. This is the non-functional requirement that most relies on the existence of components such as the **Verification&Validation Platform** the 5Gtango project is proposing (see V&V platform)
KPIs:
- availability along time
- number of successful, rejected or in error requests along time

related pilots: all

category: must

target WP: WP5

Table 5.37: Low-level non-functional requirements for service platform - reliability

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP19</td>
<td>The Service Platform must be able to support failures both in features (see Serviceability, below) and in the supporting infrastructure.</td>
<td>This implies automatic scaling out (and in, when and if the triggering cause is not valid anymore) of the platform, load balancing, etc.</td>
</tr>
</tbody>
</table>

5.2.4.5 Availability

- description: a carrier-grade Service Platform must be available whenever it’s (different) users want or need to use it. In particular, for Telecom Operators, availability usually weights more than most of the other characteristics, though the flexibility and openness needed for 5G is quickly getting as important. Balancing these often contradictory non-functional requirements into a highly available service platform will require taking advantage of many of the most recent evolutions in terms of highly available infrastructure tools (e.g., Kubernetes)

KPIs:
- availability along time;

related pilots: all

category: must

target WP: WP5

Table 5.38: Low-level non-functional requirements for service platform - availability

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP20</td>
<td>The Service Platform must be able to support updates and stressful states of functioning without severe lack of features and performance (see also Serviceability).</td>
<td>This implies several adjustments in the platform deployment scripts, testing and documenting the whole process, etc.</td>
</tr>
</tbody>
</table>

5.2.4.6 Serviceability

- description: it’s crucial, for the Service Platform owner, to know how far it can go when submitted to extreme stress or lack of supporting resources. It is our plan to revisit the design of the SONATA Service Platform and improve/extend its ability to deal with this kind of extreme conditions, for example, by temporarily disconnecting some of the features (e.g., service/function on-boarding can be switched off for a limited period of time, so that
scarce resources are devoted to other features such as monitoring). This might include how
the SP scales and places VNFs in the face of limited resources, i.e., trying to deal with few
available resources while minimizing the delay and ensuring good QoS (in the context of smart
manufacturing, we need mechanisms to ensure short delays)

- **KPIs:**
  - Number of successful, rejected or in error requests along time

- **related pilots:** all
- **category:** optional
- **target WP:** WP5

### Table 5.39: Low-level non-functional requirements for service platform - serviceability

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP21</td>
<td>The Service Platform must be able to support different combinations of available features, so that different levels of service can still be provided and extreme stressful scenarios.</td>
<td>This implies several adjustments in the platform configuration and features toggling.</td>
</tr>
</tbody>
</table>

#### 5.2.4.7 Confidentiality

- **description:** specially when supporting **slicing**, confidentiality is a key non-functional requirement within the this list. This is not only about making the owner of a slice completely unaware of the other slices that co-exist in the platform, specially at the level of the impact at resources, when its usage is within pre-defined limits

- **KPIs:**
  - resource availability (CPU, RAM, storage and network) across slices and along time (will show if resource allocation in one slice will impact or not other slice(s))

- **related Pilots:** all
- **category:** optional
- **target WP:** WP5

### Table 5.40: Low-level non-functional requirements for service platform - confidentiality

<table>
<thead>
<tr>
<th>#ID</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP22</td>
<td>The Service Platform must be able to support the strict separation of information access, according to very well defined policies and user roles.</td>
<td>This implies the review of all the user management micro-service, to enable it to control access on a slice-based environment.</td>
</tr>
</tbody>
</table>
6 Conclusion

6.1 Challenges

5GTANGO bridges the gap between SDN and NFV environments by: (i) Elaborating and producing in automatically a template that describes specifications and requirements of network services and necessary SDN applications, in the form of “intents”. Based on these intents, the SDN controller can perform the reconfiguration by translating them to prescriptions for SDN applications. (ii) Compiling the aforementioned template considering not only the cases of VNFs but also the cases of service chains.

5GTANGO will enhance current management and orchestration architectures in two main areas: (i) By integrating DevOps in lifecycle management, which will allow the incremental (and therefore, much more controlled) deployment of network services to production. 5GTANGO will focus on providing service assurance, with QoS guarantees, to satisfy the extremely challenging requirements of emerging application scenarios. (ii) By allowing the dynamic provisioning of network slices and their resource elasticity. 5GTANGO will also introduce vertical-oriented network slices, where 5G systems may support various communication protocols and services such as Internet of Things (IoT) or manufacturing scenarios (“Industry 4.0”). Advanced infrastructure will exploit the capability of data-plane programmability and associated programming interfaces, as vertical extension of slicing.

5GTANGO will go beyond the existing frameworks for VNF allocation and chaining by (i) Providing resource mapping algorithms for placement, scaling, migration, etc. that will consider both network and vertical constraints (e.g., specifics of a manufacturing scenario). (ii) Employing demand-based service templates that enable dynamic allocation of resources. (iii) Supporting multiple optimization goals such as minimizing the number of used servers, maximizing the utilization of means for production, energy consumption, service quality or security.

5GTANGO will provide a holistic breakthrough solution for validation and verification. The current NFV validation propositions (although addressing the validation requirements) are disengaged from the Orchestration Platform itself and are usually pre-active, operating in pre-deployment phases. In addition, they mostly focus on the NFVI validation and VNF performance. 5GTANGO will create a tighter coupling of the validation framework with the MANO stack, adopting methodologies and mechanisms for conducting validation tests from already established solutions. 5GTANGO will allow both in-vitro validation (sandbox environment) and in-situ (in the place of deployment) by offering high-level orchestration capabilities to the developers. The 5GTANGO framework will allow the validation components at the MANO to exploit the pluggable architecture of the Service Platform.

5GTANGO will innovate in autonomicity introducing and extending a complete monitoring framework as well as FSMs and SSMs enabling autonomic management. In greater detail, the monitoring framework will close the virtuous cycle of DevOps paradigm in 5G environments providing to the developers the required visibility upon use of resources and network service operation in-situ (in the actual deployment environment). Autonomicity in the respect of monitoring is defined as the main enabler for closing the loop between network service development, deployment and operation as defined in the Application Lifecycle Management and achieving enriched and advanced self-manageability of cloud environments, nodes and networks by design.
6.2 Pilot scenarios

5GTANGO system will be demonstrated in three vertical pilots: advanced Manufacturing, immersive Media, and real time Communications.

6.2.1 Smart manufacturing

In the context of smart manufacturing, the given advantage is based upon the processing of operational data, machine data, and process data (O/M/P data) generated. Using this data, the process of production, the control of the intralogistics supply chain as well as the control of any kind of component becomes possible. It builds the basis for future digital pervasive supply chains. An occurring challenge is the assessment of the right location for the computation of the data because two challenges occur: (1) any kind of additional equipment or equipment with high performance at the edge of the machine increases costs and reduces sustainability due to waste of computation power. Nevertheless, being at the edge and next to the machine allows real-time feedback to the machine. (2) transferring the amount of measured information from the machine to the cloud requires mechanisms that allow real-time communication and, thereby, online processing of data. Due to missing the availability of network functions on the full path from edge over fog to the cloud, the challenges can hardly be accomplished using existing solutions. 5GTANGO will address these challenges for the smart manufacturing context. With the 5GTANGO platform it will become possible to proactively access functions and services on the path that fulfill the order- and machine-dependent requirements in connectivity and services.

6.2.2 Immerse media pilot

The Immersive media use case is designed to use the capabilities of new technologies such as VR, 360° Video Capturing cameras, 5G networks, vCDNs and ability of new GPUs to render 360 content with low latency. The pilot will capture a live event using a 360° camera at 4K resolution and stream it to users sitting at the leisure of their home on their VR head mounted display (HMD), desktops or smartphones with enrichment content added to it enabling the personalisation of the live stream. New forms of media content such as very high resolution content (4K and above), High Dynamic Range (HDR) content, or immersive VR and 360°, are quickly being adopted by the market. Today, nearly every major player in the video streaming industry has made its first steps into adopting some or all of these technologies, which will soon find their way in our daily life. End-users consume video streaming typically relying on different devices such as smartphones, tablets, PCs, SmartTVs, etc. The content must undergo different network connectivity conditions on the basis of the user location (whether home, office, or on the road), and the time the content is being accessed. To counteract fluctuating network conditions and also to cope with heterogeneous user requests, efficient delivery streaming frameworks such as Dynamic Adaptive Streaming over HTTP (DASH) were recently introduced.

6.2.3 Communications Suite

In real-time unified communication system implementations it is necessary to deal with some difficulties of security (firewall rules setup, routing setup, permissions to access servers and start/stop/modify services, etc.), when monitoring and debugging (setup service and agents to monitor the machines, set the log level on a per service basis, gathering and processing of logs in a centralized storage, etc.), when scaling up the solution (due to human intervention to clone machines and modify them, resource allocation, etc.), and validation and performance tests must be adapted to each environment. The goal of this pilot is to test a scalable real-time communication platform which allows to
exchange real-time media and to use the 5GTANGO platform to provide a scalable implementation solving some of the problems mentioned. The user will access the service using a Unified Communication client which allows to create videoconferences with multiple participants. The solution also supports chat, group chat, file sharing, desktop sharing and other valuable features provided by services which can be easily scaled up like integration with PSTN.

6.3 Requirements

5GTANGO considers four main functional blocks; catalogue (data store, decision support and optimization), verification and validation (V&V) platform, NFV-enabled software development kit (SDK), and extended service platform to support vertical applications. For each of these main functional blocks, the both functional and non-functional requirements (scalability, customizability, security, reliability, availability, authentication and confidentiality) have been analysed. As for the functional requirements, the following high-level requirements have been identified:

6.3.1 Catalogue

- The 5GTANGO Catalogue should be offering a repository for persistently storing the developed VNFs/NS.
- The 5GTANGO Catalogue should be offering support for metadata annotation of stored VNFs/NS.
- The 5GTANGO Catalogue should be offering support for updating the metadata of stored VNFs/NS.
- The 5GTANGO Catalogue should be offering interfaces for interacting with other components.
- By using technologies for the automated analysis of metadata (including V&V metadata, deployment patterns, monitoring information, QoE/QoS parameters, SLAs, policies, etc) the catalogue should enable efficient selection and use of VNFs/NS.
- By analyzing the updated VNFs/NS metadata (updated based on new information from SLAs, policies as well as monitoring – QoS and QoE data) new V&V tests should be triggered (feedback loop towards V&V) and actions towards the provision of QoS guarantees should be proposed.
- Different versions of the same item should be present in the Catalogue tracking every change in its entire lifecycle and who made them.
- It is possible for many instances of the 5GTANGO catalogue to exist at the same time. For the sake of consistency, there should be interfaces for synchronizing the data between those instances.

6.3.2 V&V Platform

- The V&V shall present a set of tools that support test management (including generation, configuration & management), execution, collection and reporting
- The V&V Platform should be able to instrument and collect the resulting metrics from any tests run
- The V&V Platform should offer the following interfaces to other 5GTANGO components
6.3.3 NFV-enabled SDK

- Development of service and component tests
- Development of performance characterisation tools
- Service- and Function Specific Manager development support

6.3.4 Service Platform

- Support the complete DevOps lifecycle for NS/VNFs, including NS/VNF on-boarding, scaling, (re-) configuration, and updating
- Support the virtualization of available vertical resources
- Support of Network Slicing considering resource allocation, service contextualization, and monitoring.
- Support of carrier-grade functionalities for highly reliable and large scale deployments.
7 Acronyms

- 3GPP 3rd Generation Partnership Project:
- 5G Fifth generation of mobile networks
- 5GPPP 5G Infrastructure Public Private Partnership
- API Application Programming Interface
- AR Augmented Reality
- BSS Business Support System
- BW Bandwidth
- CDN Content Delivery Network
- CPU Central Processing Unit
- DASH Dynamic Adaptive Streaming over HTTP
- DE Decision Elements
- DevOps Development and Operations
- DME Decision-Making-Elements
- ECOMP Enhanced Control, Orchestration, Management and Policy
- EPC Evolved Packet Core
- ETSI European Telecommunications Standards Institute
- FSM Function-Specific Manager
- GANA Generic Autonomic Network Architecture
- GPU Graphics Processor Unit
- HDR High Dynamic Range
- HMD Head Mounted Display
- HMI Human-machine interface
- HOT Heat Orchestration Template
- HTML HyperText Markup Language
- IO Input Output
• IoT Internet of Things
• ISG Industry Specification Group
• ISP Internet Service Provider
• JSON JavaScript Object Notation
• MANO NFV Management and Orchestration
• MBT Model-based Testing
• MEC Mobile Edge Computing
• MPC Machine Computer
• NAT Network Address Translation
• NBI northbound interface
• NETCONF Network Configuration Protocol
• NF Network Functions
• NFV Network Function Virtualization
• NFVI NFV Infrastructure
• NFVO NFV orchestrator
• NS Network Service
• NSD NS Descriptor
• NSM Network Service Manager
• OASIS Advancement of Structured Information Standards
• ONAP Open Network Automation Platform
• OpEx Operating Expenditures
• OSS Operations Support Systems
• OTT Over-The-Top
• PC Personal Computer
• PLC Programmable Logic Controller
• PoP Point of Presence
• PSTN Public Switched Telephone Network
• QoE Quality of Experience
• QoS Quality of Service
• QSS Quobis Signalling Service
• RAMI4.0 Reference Architecture Model Industrie 4.0
• REST Representational State Transfer
• RPC remote procedure call
• SBI southbound interface
• SDK Service Development Kit
• SDN Software Defined Networking
• SFU Select Forwarding Unit
• SLA Service Level Agreement
• SP Service Platform
• SPoF Single Points of Failure
• SSM Service-Specific Manager
• SUT System Under Test
• TOSCA Topology and Orchestration Specification for Cloud Applications
• TTCN-3 Testing and Test Control Notation Version 3
• TURN Traversal Using Relays around NAT
• UHD Ultra-High Definition
• URL Uniform Resource Locator
• V&V Verification and Validation
• VIM Virtual Infrastructure Manager
• VLAN Virtual Local Area Networks
• VNF Virtualized Network Functions
• VNFD VNF Descriptor
• VNFFG VNF Forwarding Graph
• VNFM VNF Manager
• VPN Virtual Private Network
• VR Virtual Reality
• WAC WebRTC Application Controller
• WIM WAN Infrastructure Manager
• XML eXtensible Markup Language
A Bibliography


