Journal of
Sensor and Actuator Networks

MDPI

Article

Comparison of Management and Orchestration Solutions for the 5G Era

Panagiotis Trakadas [1], Panagiotis Karkazis [1,*], Helen C. Leligou [1], Theodore Zahariadis [1], Felipe Vicens [2], Arturo Zurita [2], Pol Alemany [3], Thomas Soenen [4], Carlos Parada [5], Jose Bonnet [5], Eleni Fotopoulou [6], Anastasios Zafeiropoulos [6], Evgenia Kapassa [7], Marios Touloupou [7] and Dimosthenis Kyriazis [7]

1 Synelixis Solutions, 341 00 Chalkida, Greece; ptrak@synelixis.com (P.T.); nleligou@synelixis.com (H.C.L.); zahariad@synelixis.com (T.Z.)
2 Atos, 28037 Madrid, Spain; felipe.vicens@atos.net (F.V.); arturo.zurita@atos.net (A.Z.)
3 Centre Tecnològic de Telecomunicacions de Catalunya, 08860 Castelldefels, Barcelona, Spain; palemany@cttc.es
4 IMEC, University of Ghent, 9000 Gent, Belgium; thomas.soenen@ugent.be
5 Altice Labs, 4540-225 Aveiro, Portugal; carlos-f-parada@alticelabs.com (C.P.); jbonnet@alticelabs.com (J.B.)
6 Ubitech, 15231 Chalandri Athens, Greece; efotopoulou@ubitech.eu (E.F.); azafeiropoulos@ubitech.eu (A.Z.)
7 University of Piraeus, 185 34 Pireas, Greece; ekapassa@unipi.gr (E.K.); MTouloup@unipi.gr (M.T.); dimos@unipi.gr (D.K.)
* Correspondence: pkarkazis@synelixis.com

Received: 21 November 2019; Accepted: 1 January 2020; Published: 4 January 2020

Abstract: 5G is considered to be the technology that will accommodate the development and management of innovative services with stringent and diverse requirements from end users, calling for new business models from the industry. In this context, the development and efficient management of Network Services (NS) serving specific vertical industries and spanning across multiple administrative domains and heterogeneous infrastructures is challenging. The main challenges regard the efficient provision of NSs considering the Quality of Service (QoS) requirements per vertical industry along with the optimal usage of the allocated resources. Towards addressing these challenges, this paper details an innovative approach that we have developed for managing and orchestrating such NSs, called SONATA, and compare it with OSM and Cloudify, which are two of the most known open-source Management and Orchestration (MANO) frameworks. In addition to examining the supported orchestration mechanisms per MANO framework, an evaluation of main operational and functional KPIs is provided based on experimentation using a real testbed. The final aim is the identification of their strong and weak points, and the assessment of their suitability for serving diverse vertical industry needs, including of course the Internet of Things (IoT) service ecosystem.

Keywords: 5G; Management and Orchestration framework; MANO; SDN/NFV

1. Introduction

During the last decade, the tectonic setting of the telecommunications industry has been under heavy transformation. Consequently, the traditional market players' roles are changing dramatically. Infrastructures and platforms that consist of dedicated hardware and proprietary software are moving towards open hardware and software solutions. Infrastructure operators are becoming totally separated from service providers, while the life cycle of each NS is becoming more and more demanding in terms of network dynamicity, computational capabilities, and flexibility. The aforementioned transformation comes as a natural consequence of pushing network performance and cloud-based

functionalities to their extremes due to the exponentially increasing service requirements [1]. In this fluid landscape, the evolution of mobile network technology, especially in the 4G/Long-Term Evolution (LTE) era, has allowed end users to enjoy innovative services with highly demanding requirements in terms of increased bandwidth and reduced latency. Additionally, the support of data-intensive and user-centric applications based on massive and geographically dense deployment and connectivity of smart devices, is becoming also an opportunity. Although the 4G/LTE era is far from being considered as technologically "mature", the telecommunication industry has already taken a step towards the establishment of a radical 5G ecosystem, to ensure that the industry continues to meet the market demand for mobile services as they evolve, as well as further stimulate economic interest and address societal needs. While the 5G concept is still rather unclear in certain dimensions [1], the evolved 5G network will be, for sure, characterized by agile, resilient and converged network realization. These characteristics are based on one hand on the separation between the core and radio access technologies and on the other hand on softwarization context brought by Network Function Virtualization (NFV) and Software Defined Networking (SDN) technologies. In the network softwarization frontier, a handful of commercial as well as open-source Management and Orchestration (MANO) frameworks have emerged, dealing with issues related to the assurance of proper operation of the NFV Infrastructure (NFVI) and Virtual Network Functions (VNF), deployed across administrative domains and heterogeneous infrastructures. In this perspective, MANO frameworks can be considered to be a management and orchestration suite for physical and virtual resources related to the life cycle of the deployed NS.

This paper describes the capabilities of SONATA MANO , as we have developed it within the framework of the 5GTANGO 5G PPP project [2], comparing it with two other well-established open-source solutions. Finally, a sandbox environment has been used for the comparison of the three considered MANO implementations against functional Key Performance Indicators (KPIs) for different NS deployments. The main contributions of this paper are summarized below:

- Providing a comprehensive survey of three representative widely accepted and established open-source MANO frameworks.
- Comparing the functionalities provided by each of the major components comprising the MANO frameworks under discussion.
- Defining the appropriate test procedures based on well-known functional and operational KPIs defined in the literature.
- Defining a sandbox environment supporting automatic test execution, common to all MANO frameworks to ensure fairness among them.
- Providing an in-depth comparison of MANO frameworks, based on sandbox environment testing, highlighting advantages and disadvantages of each solution, to guide further development.

The remaining of the paper is organized as follows. Section 2 presents the three MANO frameworks implementations, while Section 3 provides an extensive comparison between the components of the SONATA, OSM and Cloudify MANO frameworks. Section 4 presents the performance evaluation of the compared MANO frameworks, on operational and functional KPIs. Finally, in Section 5, we close with ideas for future work and current study capabilities.

2. Related Work

2.1. Open-Source MANO Frameworks Overview

The design and development of MANO frameworks has attracted the attention of large companies, universities and operators, which build their own solutions, following the respective ETSI-NFV standards. The ETSI MANO framework is an open-source framework developed by ETSI ISG NFV [3]. The architecture of the NFV framework is shown in Figure 1. The lower left part depicts the NFVI, which consists of hardware resources that are virtualized in order to be used by several

network functions. These VNFs are presented on top of the NFVI, followed by the respective Element Management System (EMS) that performs the typical management functionality for one or several VNFs. Moreover, on the upper left part of Figure 1, the component named "Service, VNF and Infrastructure descriptions" is depicted; this component provides information regarding the VNF deployment template, the VNF Forwarding Graph (FG), as well as other infrastructure and service-related functionalities. Finally, on top of Figure 1, the Operations Support Systems (OSS)/Business Support Systems (BSS) of the operator is shown. The MANO framework is depicted at the right part of the figure and consists of three main building blocks:

- The Virtualized Infrastructure Manager (VIM): Responsible for coordinating the functionalities that are used to control and manage the interaction of a VNF with computing, storage and network resources, as well as their virtualization. Multiple VIMs instances may be deployed, one per different type of NFVI technology.
- The VNF Manager: Responsible for the Life-Cycle Management (LCM) of one or multiple VNFs. Several VNF Managers may be deployed.
- The NFV Orchestrator (NFVO): Responsible for the orchestration and management of the NS deployed on the NFVIs.

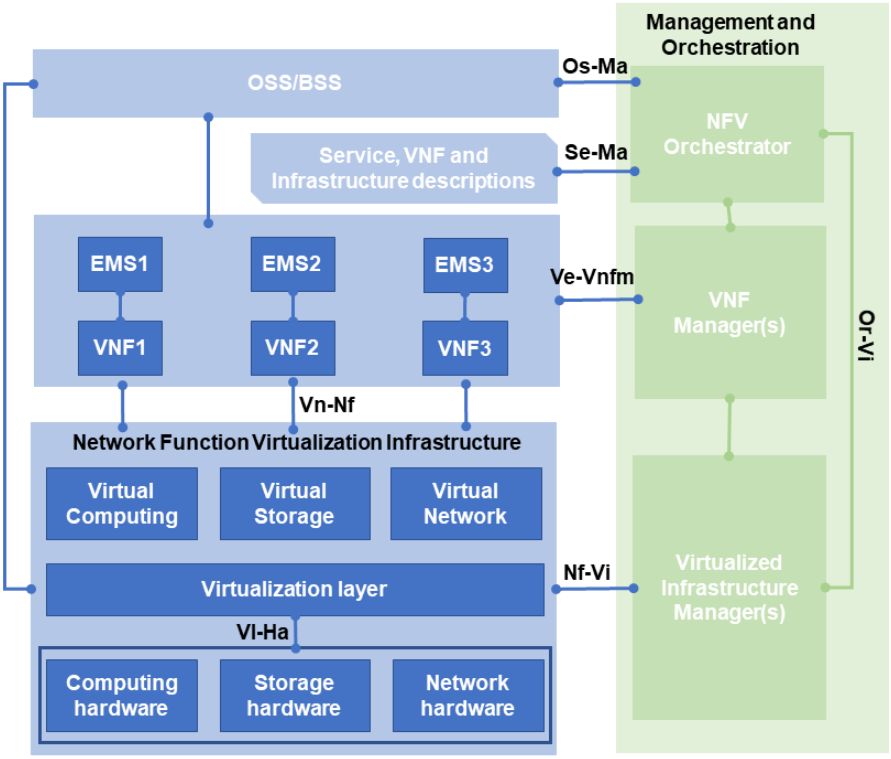

Figure 1. ETSI-NFV architecture [3].

In the sequel, we briefly describe, SONATA, OSM and Cloudify implementations (an extensive list of open-source and commercial MANO frameworks is presented in refs. [4–6]). The current work focuses on the above-mentioned open-source implementations. The selection of the specific MANO frameworks was based on the fact that they require the same level of hardware resources (as we will discuss in section 4.2), making their comparison more accurate and fair.

Open-Source MANO (OSM) [7] is a MANO implementation based on ETSI-NFV Information Models. OSM provides a high-quality open-source MANO stack that fulfills the requirements of commercial NFV applications. OSM is based on the principles of Resource Orchestration (RO) and Service Orchestration (SO) that is built on the VNF Manager (VNFM) and NFVO functional blocks.

Also, OSM enhances the NFVO role with service orchestration capabilities that include the ability to manage NS instances during run-time.

Cloudify [8] is also an open-source cloud orchestration platform, which provides a commercial release targeting vendors. Cloudify orchestrator is stable and currently used in many production environments. The vast adoption of Cloudify is due to its powerful core engine that manages the complete life cycle of the services across many cloud environments. Also, there are many plugins available that make integration with other platforms easier. Cloudify designs and deploys NSs based on a descriptive language that can be considered to be a NFVO as well as a VNFM under the perspective of the ETSI-NFV architecture.

Finally, SONATA (as enhanced and implemented by the 5GTANGO project [2]) is an ETSI-compliant solution which is also offered as open-source. Its feature set includes resource and service orchestration, as well as VNF management. SONATA is customizable by design, including swappable modular plugins, such as life-cycle management, service monitoring, conflict resolution, network slice management, policies enforcement and run-time Service-Level Agreement (SLA) contracts. Additionally, it provides open interfaces for multi-vendor support, independent of the supporting orchestration stack(s). Last but not least, it offers compatibility with leading cloud computing solutions (especially OpenStack and Kubernetes).

2.2. Motivation and State of the Art

In the upcoming 5G era a wide range of new business opportunities are anticipated [9–11]. These opportunities are tightly coupled with the rising popularity of sensor networks and IoT [12]. Sensor networks in combination with SDN, facilitate the development of a wide range of novel applications in various fields, including smart cities, smart healthcare, smart transport, industrial automation and disaster response to name a few. SDN can provide better network management efficiency, reliability and latency of the system, as has been proved by refs. [13,14], where the authors investigate Fog computing in relation to the SDN paradigms. However, it is not investigated how the QoS will be guaranteed in such environments. In ref. [15], the reported research focused on this direction, giving emphasis on how efficient management of resources will potentially improve the QoS provisioning in Software Defined Wireless Sensor Networks.

In the aforementioned studies though, the underlying management and orchestration platforms are not taken into consideration. In ref. [16], the authors provide a basic analysis and comparison of ONAP and OSM, but limited to the scope of installation requirements, documentation maturity and simplicity of installation process, without dealing with functional or operational characteristics of these MANO frameworks. Thus, the conclusions of this comparison are that: (1) ONAP requires much more resources to be installed (2) OSM offers better documentation, and (3) ONAP provides more functionality that spans across all topics of MANO scope. However, no results supporting the latter conclusion are provided. The design and development of 5TONIC, another NFV MANO implementation is presented in ref. [17]. This is an open research and innovation laboratory on 5G technologies, providing trials and experiments with access to a functional production-like NFV environment, based on OSM. Although this paper does not compare characteristics and performance between MANO frameworks, it provides validation results of 5TONIC platform performance. In particlar, the first experiment presents results for the average time of the deployment of an NS consisting of several VNFs. The second experiment shows how the deployment of secton an NS is affected by existing deployments, measured from the successive deployment of 16 instances of the same NS, consisting of a single VNF. Finally, the authors in ref. [18] follow a comprehensive methodology to provide a set of well-defined KPIs that allow for the comparison and benchmarking of MANO frameworks. In this respect, the authors split the KPIs into functional ones that describe non-run-time characteristics of the MANO system, and operational ones that characterize the run-time operations covering the life-cycle management of the NS and VNFs. In order to support this categorization, the paper includes results from the comparison of OSM and ONAP. With respect to the functional

KPIs, the paper presents results regarding the resource footprint of each MANO deployment and the number of supported VIMs. Most importantly, the authors provide results on operational KPIs related to a) the On-boarding Process Delay (OPD), i.e., the time required for an NS to be on-boarded on the MANO, and b)the Deployment Process Delay (DPD); that is the time needed for an NS to be instantiated in the NFVI. Also, the authors defined two additional performance indicators the run-time Orchestration Delay (ROD) and the Quality of Decision (QoD). ROD measures the time difference from the moment an LCM action is executed to the time the action is completed. And the QoD quantifies the performance of a MANO in terms of its effectiveness in carrying out run-time LCM operations, which could not be measured and compared as the versions of OSM and ONAP that have been used at that time were not supporting such actions.

The current paper is the first one that provides a comparative and in-depth analysis of three MANO frameworks implementations using a common test environment and KPIs (i.e., ROD, scaling-in/out action) that was not possible to be measured in the previous works.

3. Comparison of the MANO Components

The purpose of this section is to provide a high-level view of the main components comprising the SONATA MANO (Figure 2) and compare them with the relevant solutions provided by Cloudify and OSM.

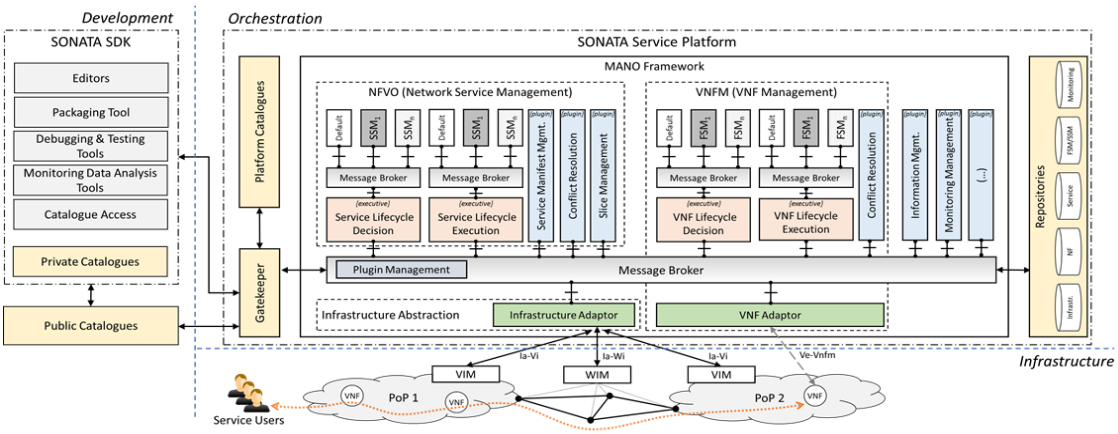

Figure 2. SONATA architecture diagram [2].

3.1. External API

One of the most important features for every MANO framework is to provide a secure and user-friendly set of interfaces to support connectivity and accessibility of external entities to the MANO functionality. Thus, all the MANO frameworks tried to fulfill this requirement from the very beginning of their implementation steps.

In the first releases of OSM, the communication with the external systems was facilitated by the Service Orchestration (SO) and the Resource Orchestration (RO) APIs. From release four on-wards, the Northbound Interface (NBI) that provides this functionality was implemented as an independent component, providing access to internal OSM components through an Apache Kafka message bus. This approach upgraded the NBI to an asynchronous and highly available RESTful API. This API is fully aligned with ETSI SOL-005 and supports also authorization and authentication services based on Keystone server running inside the NBI.

Similarly, Cloudify Manager includes several ways to interact and deploy applications that enable their management and workflows via a REST API that can also be used directly with Cloudify Command Line Interpreter (CLI). In addition to these, Cloudify includes entities, named agents, that are used for deploying executions on application hosts, by listening to task queues and executing tasks

when required. The agents work with specific plugins configured at the descriptor level (blueprint), which means they do not have to be pre-installed in the hosts.

In comparison to the aforementioned frameworks, in SONATA the component that implements this functionality is called Gatekeeper and was introduced in SONATA from its first release. As shown in Figure 2, the Gatekeeper is the single access point to the SONATA MANO and it is responsible for exposing the SONATA's APIs to the outside world, ensuring that every access to the SONATA Service Platform (SP) is performed by authenticated and authorized users. The process includes the filtering out of invalid requests, before they are processed by the relevant internal component (micro-service). The Gatekeeper interacts internally with all NFVO and VNFM components using SONATA's message broker and it is responsible for uploading and retrieving descriptors from the Catalogue, instantiate and delete NSs (through life-cycle management), exposing monitoring metrics (through Monitoring Manager) and enforcing licensing and policy models (through the SLA and Policy Manager respectively).

3.2. Ns Life-Cycle Management

The most important role of any MANO framework is the life-cycle management of every NS. From instantiation to termination, the MANO executes all tasks related to placement calculations, healing, reconfiguration, etc.

In OSM, these actions are performed by the RO module. In particular, regarding the run-time reconfiguration of the VNFs the RO module in collaboration with the Lightweight Management uses Juju [19] charms to manage the complete life cycle of the VNF, including software installation, configuration, clustering, and scaling. There are two types of charms; proxy and machine. Proxy charms are operating remotely from a VNF (running on a VM or physical device) using SSH. Machine charms are written to inside the VNF and handles the complete life cycle of the VNF from installation to removal.

On the other hand, Cloudify claims that it provides a simple embedded VNFM solution for VNF vendors for full life-cycle automation and multi-cloud support. The VNFM is implemented in the Embedded Orchestration Engine (EOE) and continuously monitors the behavior of the VNF and takes all the necessary actions (reconfiguration, scaling etc.) automatically.

SONATA uses the ETSI architecture, but extends it with a unique service-specific management mechanism enabling NS developers to customize the MANO behavior with optimized workflows. Such a concept is necessary as in the NFV context, NSs are often provided by independent developers who are the only people aware of the internal composition and behavior of the service. For example, when an NS needs to scale, only the developer knows which VNF instances to add or remove, where to add them, how to re-configure each of the running instances and the networking between them, etc. Through this service-specific management mechanism, developers can overwrite generic MANO behavior with custom logic. This is done through the Service-Specific Managers (SSM) and the Function Specific Managers (FSM) (Figure 2). In the ETSI dialect, SSMs customize NFVO workflows (i.e., service level) such as scaling and migration. On the other hand, SONATA FSM is the equivalent of the ETSI VNFM (i.e., VNF level) that can customize VNF configuration or healing workflows. Even though VNFM customization concepts in OSM are quite similar to FSMs, service-level customization is unique to SONATA. The introduction of this mechanism gives to the developers a fine-grained control over the operational life cycle of their product. This enables SONATA to support any NS, irrespectively of the complexity of its life-cycle events, and therefore a wider variety of SLAs and policies, as they are built on top of this extended set of life-cycle events.

3.3. Infrastructure Abstraction

The Infrastructure Abstraction (IA) component intends to abstract the multiple VIMs and WAN Infrastructure Managers (WIM), providing a unified approach to the MANO component to interact

with the underlying infrastructure. The IA provides a unique northbound interface, regardless of the plugin that is going to be used.

In the current SONATA implementation, the IA (Figure 2) is responsible to interact with the multiple VIMs and WIMs, providing a unified and abstracted NBI API to components that require the management of resources (mainly the MANO Framework, but also the Slice Manager). This way, the management of different types of technologies (e.g., Virtual Machines (VMs), containers etc.) become unified even if the original resource management APIs are significantly different. By using the IA, other components can be agnostic to the details of a particular technology (e.g., Openstack, Kubernetes, etc.), as those details are embedded in IA plugins, leading to an easy use and flexible VIM/WIM extensibility. Currently, the IA supports three VIMs: (a) Openstack (heat), (b)Kubernetes (k8s), and (c) VIM Emulator (vim-emu), as well as two WIMs: (a) Virtual Tenant Network (VTN) (OpenDayLight app) and (b) Transport API (T-API).

Compared to the SONATA IA, OSM and Cloudify follow similar approaches. In OSM case, deployments in Openstack-based cloud environments, VMware and Amazon Web services (AWS) are supported. In Cloudify, micro-services can be deployed in a variety to different types of VIMs, from Openstack to Kubernetes but in some cases the usage of special plugins is required.

3.4. Monitoring Framework

The monitoring framework is one of the vital functionalities that any MANO must include, since it allows control of the optimization of infrastructure resources as well as the performance of the NSs [20]. However, MANO frameworks adopted different approaches and tools for the realization of an efficient way to monitor resources and services.

To begin with, OSM introduced monitoring in an experimental mode in release 3.0 and keeps enhancing its functionality from that time on-wards. The philosophy that OSM community adopted for its Monitoring Module is that it is not intended to replicate or compete with existing solutions, but rather to provide plugins to incorporate them within OSM architecture and further use it as a conduit for steering actionable events into the SO. One of the most powerful functions that OSM is delivering, as part of the Monitoring Module, is the ability to correlate monitoring information of the VNFs to the relevant NSs. The OSM Monitoring Module is designed to support a flexible plugin method to integrate with the monitoring tool of choice. In the latest release, OpenStack Aodh, OpenStack Gnocchi, Amazon CloudWatch and VMware vRealize Operations Manager are supported, offering a clear advantage against other MANO frameworks that are not able to directly support monitoring tools for the aforementioned cloud platforms.

Cloudify also provides monitoring features starting from a statistics page that enables monitoring of the deployments with regards to system measurements and resources. The Cloudify monitoring statistics appear in form of widgets that can be configured to visualize other metrics. Cloudify incorporates a monitoring plugin called Diamond that is used for installing and configuring monitoring agents in deployed hosts. The plugin collects system resource metrics to publish them in selected destinations, including also an API for the development of custom metric collectors.

On the other hand, SONATA (Figure 2) was the first MANO framework to embrace and integrate a complete monitoring solution based on Prometheus [21] monitoring tool, well before it became a Graduated Project of Cloud Native Computing Foundation (CNCF). Furthermore, since SONATA included a monitoring framework from its first version, the consortium realized that in order to accommodate deployment of innovative NSs, the monitoring framework must address the following requirements: (a) to provide monitoring information related to specific performance metrics of the running NS, (b) to provide monitoring information coming from the VMs or containers hosting the VNFs or Cloud-native Network Functions (CNFs) respectively, and (c) to provide a run-time reconfiguration mechanism for monitoring rules in order to support other components, like SLA Manager, Policy Manager and FSMs but also NS/VNF developers. In its current version 5.0, SONATA monitoring framework provides monitoring data for hosts, VMs, containers, networking

infrastructure (SDN controllers) and also mechanisms in order to collect application-specific metrics from NS/VNFs based on SNMP protocol. Also, SONATA provides monitoring solutions for CNFs deployed on Kubernetes. Taking under consideration that MANO functions are highly dependent on the information coming from the monitoring framework, it is vital to collect data from physical and virtual resources and also custom monitoring metrics directly from the VNFs, representing the Quality of Experience (QoE)/QoS of the NS. Under this perspective, SONATA provides the most complete monitoring framework, which is ready to accommodate monitoring metrics from many sources including service-specific ones.

3.5. Slice Manager

Another key component within the Service Platform of the SONATA is the Network Slice Manager. The basic idea of Network Slicing is to create dedicated virtual networks using common physical infrastructure for a specific service by using independent logical network functions. This component aims at becoming complementary to the already used SDN/NFV technologies so that, all together, improve the existing network infrastructure resources usage and management.

Multiple standardization organizations and research groups are currently looking towards a common definition and a common model. Indeed, the proof that the network slice concept is still being discussed is the fact that well-recognized standardization organizations recently published their first documents; in December 2017, ETSI published a relevant document [22] and in October 2018, the 3GPP presented to the Global System for Mobile Communications (GSMA) a liaison with its data model. Due to this immaturity of the slicing concept, there are only ongoing efforts within the MANO frameworks to have the Network Slicing feature integrated into their software. The main decision within SONATA Network Slice Manager is to implement a component which will be aligned with the 3GPP specifications [23]. Based on these standardization documents, the Network Slicing component of SONATA consists of two main modules. The first module is the Slice Life-cycle Manager which manages the following two data objects: (a) the Network Slice Template (NST), which is the descriptor that defines a Network Slice and which NS it consists of, and (b) the Network Slice Instances (NSI) that are replicas based on an NST and are used to instantiate the set of NSs defined within the NST. The second module is called Slice2NS Mapper and is responsible to map each set of NSs described within an NST to a set of requests to instantiate these services and manage their corresponding responses by updating the correct NSI object with the data coming from the MANO framework. A key point that should be mentioned is that SONATA is in close collaboration with OSM to define how to integrate SONATA Network Slicing implementation with OSM framework [7].

3.6. SLA Manager

Within an SDN, each vertical use case may potentially require different types of NSs. Various applications, ranging from Virtual Reality (VR) and smart manufacturing to massive deployment of smart devices and e-health, require diverse SLAs reflecting different QoS requirements.

In OSM, unlike Cloudify and SONATA, the resource orchestrator does not provide mechanisms to ensure SLAs are respected.

Cloudify describes SLA requirements at the service level. The SLA can be described in the Network Service Descriptor (NSD) to ensure the needed resources for the service, for example in relation to the number of service instances. The deployments are not considered successful unless the specified resources are available and allocated/reserved. Preliminary work has been done in the framework of the SONATA project, taking into consideration mainly licensing issues and possible solutions [2,24]. In SONATA version 5.0, the SLA management provides a set of tools that connect to external systems which manage these SLAs and allow the implementation of efficient business models. Thus, SONATA is the first MANO framework which provides a complete SLA management framework that provided SLA and Licensing management at the same time. The SONATA SLA Manager is the dedicated component that allows management of the whole life cycle of SLAs, from template creation

to violation identification. Specifically, this component provides mechanisms for SLA management in individual and federated environments, based on WS-Agreement specification [25]. The developed SLA Manager is a multi-platform service, which can be adapted and extended to be used in different service platforms. The workflow consists of four discrete phases: (a) the definition and advertisement of the capabilities of network operators in SLA Template (SLA-T) form, (b) the creation and management of SLA Instances (SLA-I) based on the templates, (c) the monitoring agreement compliance during run-time and d)the management of SLA violations. Moreover, an important feature of the SONATA SLA Manager, is the introduction of licenses into the SLAs. The developed SLA Manager proposes a service-based licensing model, which links a license to a specific customer and an instantiated NS, by specifying the number of allowed NS instances. The provided licensing system within SONATA, will potentially increase the business opportunities, because service providers can connect their services to licensing models to recognize what and when to bill customers.

3.7. Policy Manager

The main objective of the introduction of run-time policies management functionalities is the injection of intelligence in the NS orchestration mechanisms. Suitable orchestration actions (e.g., elasticity/security management actions and/or life-cycle management actions for a VNF etc.) to improve the performance of the NS are defined and suggested/deployed in an automated way. Such performance aspects can be associated with the enforcement of a specific SLA, leading to a direct association between high-level SLA objectives and lower-level run-time policies design. Currently, policies management mechanisms are introduced in SONATA, Cloudify and OSM orchestration ecosystems and follow a similar workflow based on consumption of monitoring metrics from a message broker and triggering of specific alarms in case some conditions are satisfied.

Cloudify provides a real-time analysis mechanism for streaming events that trigger elasticity actions and auto-healing actions. Auto-healing refers to the process of identifying a certain component failure and providing a set of actions for fixing it. The events are published through a RabbitMQ message broker and the analysis is performed using a Riemann event processor. Cloudify includes several built-in policies but also supports the creation of custom policies.

In OSM, the Policy Manager is capable of triggering alarms based on infrastructure and VNF monitoring metrics. The alarms are triggered based on rules that are defined within the VNF descriptors. Each alarm refers to a scaling action (add or remove VNF instances) and is based on the evaluation of a single monitoring metric of the considered VNF.

In SONATA, run-time policies definition follows the design of a policies descriptor template and is realized through a Policies Editor. In this respect, each NS policy consists of a set of rules [26], where each rule is built based on a set of conditions and actions. The conditions may regard resource management metrics and VNF specific metrics, while the set of actions may address horizontal scaling and security management. Based on alarms provided by the monitoring framework, inference is realized leading to triggering and enforcement of orchestration actions by publishing messages to other MANO components through the SONATA's message broker, shown in Figure 2. The policies management approach in SONATA is partially aligned with the relevant approach in OSM, given that in both cases, triggering of alerts is realized upon examination of a set of conditions by the monitoring mechanisms. However, the SONATA Policy Manager supports the definition of multiple policies per NS, since the policy rules are decoupled from the VNF descriptor, compared to OSM where the policy rules are hard coded within the VNF descriptors. Also, SONATA supports inference and conflict resolution mechanisms, since it supports evaluation of multiple monitoring rules. Finally, in SONATA, the policy rules may be updated or deactivated on the fly, during the NS operation.

Concluding the discussion on the differences and the similarities between the open-source MANO frameworks, we summarize the main characteristics of each component per MANO in the following (Table 1). It is clear that even though all three MANO frameworks follow a similar architectural approach, they do not support all the functionalities in the same way.

Table 1. MANO characteristics.

Component	SONATA	OSM	Cloudify
External API	Asynchronous RESTful API/Users and Services authorization		
NS Life-Cycle Management	VNF and NS level (SSM and FSM)	VNF level (Juju charms)	VNF level (EOE)
Infrastructure Abstraction	VMs/Containers/Emulator	VMs	VMs/Containers
Monitoring Framework	Run-time rules reconfiguration VM/container/service metrics	VM metrics	System resources/custom metrics
Slice Manager	Based on 3GPP specs	N/A	N/A
SLA Manager	NS level SLAs Easy user interface licensing support	N/A	NS Level SLAs Definition in NSD
Policy Manager	Easy user interface Multiple Policies per NS Security Actions	Single Policy per NS	Self-Healing Actions Single Policy per NS Inference Engine

4. Performance Evaluation

This section presents the configuration setup of the testbed. After that, we analyze and compare the performance of the three MANO frameworks, namely OSM, SONATA and Cloudify, discussing their advantages and disadvantages. The Key Performance Indicators (KPIs) that have been used for comparison are based on the analysis presented in ref. [27], where the KPIs are categorized in functional ones, describing non-run-time characteristics of a MANO framework, such as resource footprint and number of supported VIMs, and operational ones, dealing with run-time operations, such as the time to on-board an NS package or the time to scale up a VNF. The focus in our work is mainly around the comparison of the operational KPIs of the three MANOs, although a table with the comparison of functional KPIs is also provided.

4.1. Testbed Setup

For the sake of fairness among the MANO frameworks comparison, we created a sandbox test environment, consisting of three physical servers (Table 2), very similar to a real design of an NFV production environment. One server is dedicated to the installation of the MANO framework, while the other two are used to host the NFVI, which has been selected to be OpenStack, Ocata version.

Table 2. Server H/W characteristics.

	PowerEdge T330
CPU	Intel Xeon E3-1240 v6 3.7 GHz, 8 MB cache, 4 Cores/8 Threads, TDP 72 W
RAM	16 GB (1 × 16 GB) 2400 MT/s DDR4 ECC UDIMM
HDD	300 GB 10 K RPM SAS 12 GBps 2.5 in Hot-plug Hard Drive, 3.5 in HYB CARR
NET	Intel Ethernet I350 QP 1 GB Server Adapter

Regarding the physical networking infrastructure, the MANO framework is connected to the VIM via an Aruba 2930F 1 GBbps ToR switch (Figure 3). At this point, it should be noted that the MANO frameworks could be also installed and run in VMs. However, we did not opt for this choice, because of the potential impact of the hypervisor on the performance of the MANO frameworks. Therefore, all frameworks have been tested on the same server, following a three-phase scheme: (a) installation of the Operating System (Table 3), (b) installation of the MANO framework and (c) execution of the performance tests.

Table 3. Operating Systems.

	Operating Systems
SONATA	Ubuntu 18.04.3 LTS
OSM	Ubuntu 18.04.3 LTS
Cloudify	CentOS 7
OpenStack	Ubuntu 18.04.3 LTS

All the MANOs were tested under the same conditions, performing the same actions, using the same NFVI and the same physical network infrastructure. However, in order to further ensure the same test conditions, throughout the execution of the tests, the selected NSs consisted of the same VNFs that implements a simple cirros OS (CirrOS 0.4.0) for all cases. Furthermore, all the performance tests were implemented by automated execution scripts based on the provided interfaces (CLI, RESTful APIs). In order to also test the statistical behavior of the MANO frameworks, each test was executed 50 times, and we collected the min, max and average execution times. In order to facilitate this process and to recreate the same stress conditions to all MANOs, a Jenkins server was used. For our analysis we used SONATA release 5.0, OSM release 6.0 and Cloudify version 5.0.

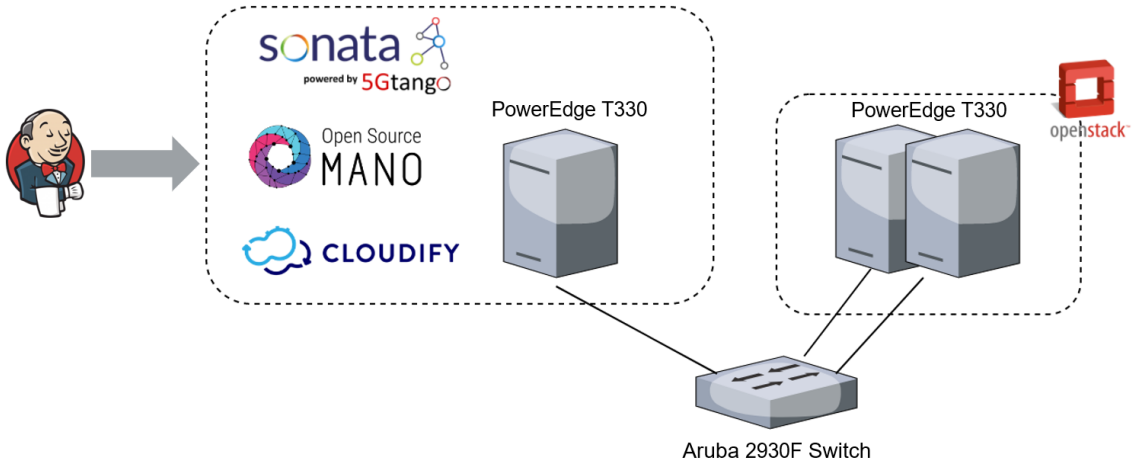

Figure 3. Overview of the testbed setup

4.2. Comparison of Functional KPIs

In this section, we discuss some of the main functional characteristics of the MANO frameworks that have arisen during the installation and testing phases.

With respect to the resource footprint, all three platforms require the same low amount of resources (Table 4), which makes them lightweight and deployable even to a single server. Also, it is worth mentioning that the computational and storage resources of the sandbox environment, which hosts the MANO installations, are more than enough to fulfill their requirements. Regarding the installation times, we observe that for all frameworks are approximately the same (Table 4). In general, it is highly depending on the computing power and the Internet bandwidth connection of the host server. Taking into account that in our case we used the same physical server we can proceed with a comparison between them. Even though in all cases scripts for automated installation are provided, SONATA and OSM scripts are fully automated in contrast to Cloudify in which some manual steps are needed. On the other hand, there are some remarkable differences in the supported VIM types like the AWS in OSM and Kubernetes in SONATA. Also, Cloudify with the usage of the appropriate plugin can support the definition of Kubernetes resources in the blueprints [28]. It is worth mentioning that OSM and SONATA follow the micro-services concept, implementing all their components as stateless services running as containers. This approach provides very fast startup times and sets the bases to

support High Availability (HA) MANO deployments in the future. These features are very important in production environments because NFV vendors need to be sure that in case of any malfunction the MANO will react automatically and quickly fix the problem without any effect in the QoS. Finally, a common feature in all frameworks is the CLI management tool, which provides easy management access and programmability.

Table 4. Comparison of functional characteristics.

Functional Characteristic	SONATA	OSM	CLOUDIFY
Resource footprint	Low (2 vCPU, 8 GB, 40 GB HDD)		
Platform installation time	30 min	25 min	25 min
Supported VIM types	OpenStack OpenVIM Kubernetes	OpenStack VMware OpenVIM AWS	OpenStack
CLI support	Yes	Yes	Yes

4.3. Testing Scenarios for Operational KPIs

In this section, we compare the operational characteristics of the MANO frameworks under test. In particular, we base our comparison in the following four scenarios: First, we measure the package on-boarding time, which is the time needed for an NS package, consisting of the VNF descriptors, the NS descriptor, and the VNF Forwarding Graph (VNFFG) to be on-boarded on the platform. Second, we measure the NS installation time, which is the time required for the same NS package to be deployed in the OpenStack environment. Third, we measure the time needed for the scale-out operation and finally we measure the time required for the scale in execution. It is highlighted that all the above tests have been automatically executed 50 times, using NSs of different size consisting of one, two and three VNF instances. The gathered results (minimum/maximum/average values) are presented in the following figures.

With respect to the on-boarding time, the three frameworks show similar performance. The required average time for all the scenarios was close to 2 s for OSM and Cloudify, while for SONATA it was around 1 s. Each VNF consists of one Virtual Deployment Unit (VDU) and also their images have been pre-loaded in the OpenStack Glance, so the measurements represent the time needed for the management actions of each MANO (e.g., user authentication, NSD/VNFD versioning, etc.). This behavior was expected because all the MANOs under test use similar technologies (e.g., RESTful APIs) for the management of the NS packages.

The results concerning the NS instantiation time revealed interesting aspects of the frameworks. As instantiation time, we defined the time period from the request until the time that the MANO changes the status of the new service to "READY". In all MANO frameworks, a new NS is considered to be finished when all VDUs are running (this information is coming from VIM) and the network configuration is also completed. Of course, the fact that the VDUs are in running state does not necessarily mean that the services that they host are also up and running. SONATA provides an SDK tool that can be used for building and validation of new packages so that the developer can be sure that the package is going to be instantiated correctly. Now, as shown in Figure 4, OSM performed better than the other two in all three deployment scenarios, while SONATA performs better than Cloudify only in case of an NS with 1 VNF. Taking a closer look at the minimum and maximum values, it is evident that maximum values of a certain MANO (e.g., SONATA) may exceed average values of others (e.g., OSM). Another interesting observation is that in all cases the variation between the minimum and the maximum values is pretty large, but for SONATA, in some cases the maximum values are three or more times bigger than of the average value. This means that a small number of NS instantiations took too much time to be completed, which is something that must be further investigated in the future.

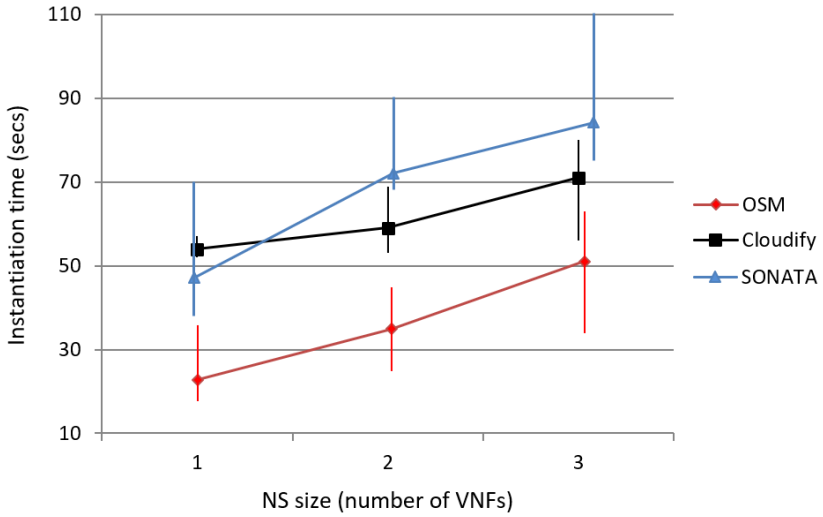

Figure 4. Average NS instantiation time

Regarding the scaling actions, many MANOs claimed in the past that they were capable of performing such actions as part of the LCM operations [27] without big success. In the latest versions, all the under-test MANO performed the requested scaling actions without any problems, giving the impression that the frameworks are mature enough to take their places in production environments. At this point, SONATA has a clear advantage by implementing a fully automated LCM mechanism that includes the definition of SLA contracts and policy enforcement through the FSM, based on monitoring metrics and rules. For each NS, the developer can define several policies, the triggering rules and the corresponding actions in the FSM, so that the user of the service can choose which policy fits better to his/her needs and request it during the instantiation phase.

In the scaling out scenario (Figure 5), Cloudify and SONATA performance are almost identical, while OSM, although performing better for the case of one VNF, linearly increases its required time for the cases of two and three VNFs, ending up with almost double time, compared to the other two platforms, for the case of three VNFs. OSM behavior is noted as a point for future work. If the scaling time is proportional to the number of VNF instances, this will have a negative impact on the LCM actions in case that the NS consists of many VNFs.

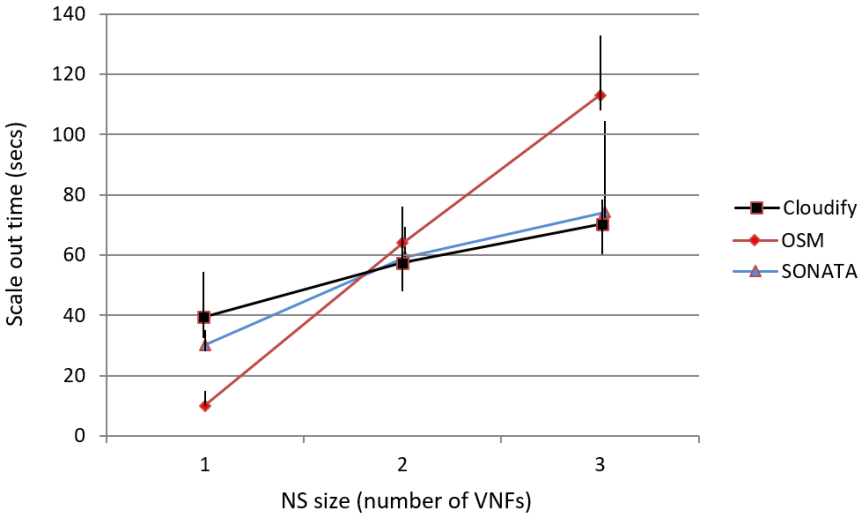

Figure 5. Average NS scaling out time

Finally, with respect to the scaling-in scenarios, the differences are small. However, OSM performs slightly better also in these scenarios (Figure 6). A general observation is that the scale-in actions are performed much faster than the scale-out ones in all cases. This is explained because during the scale-in process the MANO first sets the new network configuration, using the WIM controller and, then deletes the VDUs. In contrast, during the scale-out the MANO must first instantiate the new VDUs, then create the new network graph, attach the VDUs to the new network and finally deliver the NS for usage.

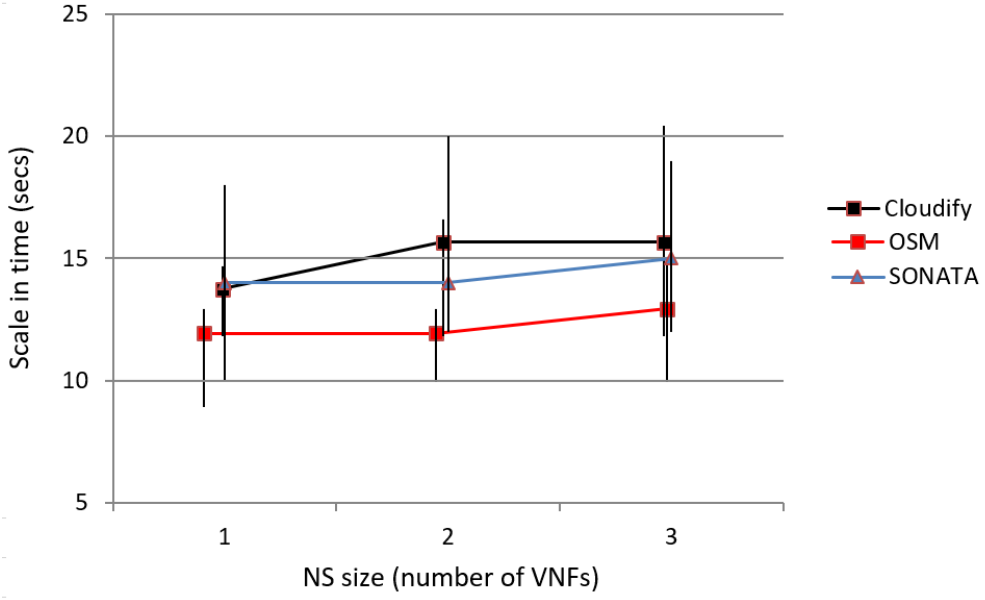

Figure 6. Average scaling-in time

As a general conclusion, all MANOs seem to be functional and perform correctly all the requested tasks within a reasonable time period. No special problems were faced in terms of adoption, usage and stability of the provided mechanisms in all the cases. The software releases were stable and facilitated the realization of experiments and the examination of the targeted performance characteristics. Validation of the proper instantiation and execution of the NSs was realized, taking advantage of the available logging and reporting mechanisms. However, it seems that some limitations exist in terms of security, since there are not in place strong guarantees for tackling targeted cyberattacks. The existence of a wide community for the support of both OSM and Cloudify, along with the continuous expansion of the interested parties adopting SONATA, can act as a guarantee for the continuous evolution of the supported orchestration mechanisms and the support of quality assurance processes in the software releases.

Furthermore, based on the operational and functional KPIs measurements it seems that OSM performs better in most cases but there is a linear increment of the scaling out time that reaches the double value of Cloudify and SONATA in three VNFs. Therefore, there is a need for future research work (a) performing scaling tests based on real NSs consisting of many VNFs (more than 10) over NFVIs with more resources, (b) using different types of virtualization technologies (Linux containers, Kubernetes etc.) and (c) comparing SONATA MANO with other solutions offered by key market vendors.

5. Conclusions

Emerging 5G environments and ecosystems aim at facilitating the diverse needs of composite applications (i.e., applications consisting of micro-services) through a set of technologies and mechanisms. While the underlying 5G technologies provide the required network resources, there are

several challenges on the layers on top that deal with the management and orchestration of physical and virtual resources (as the means for deploying VNFs). In this paper, we presented an innovative approach for managing and orchestrating such resources called SONATA, as we have developed it within the 5GTANGO 5G PPP project [2], in comparison with two of the most known open-source MANO frameworks; namely OSM and Cloudify. The above MANO implementations have been tested and validated in a real testbed against operational and functional KPIs. In all cases, the MANO frameworks deployment and operation has been stable, while no bottlenecks have been identified in terms of performance deterioration due to limited resources allocation.

OSM proved mature and robust and Cloudify proved appropriate for deployments that have no strict requirements like run-time SLA contracts and network slicing, while SONATA provides a complete tool chain for automated NS management in the dynamic 5G context era including innovative features and tools like SDK, monitoring, policies, SLA and networks slicing managers. SONATA uses a unique modular architecture, implemented through a service bus, enabling management and orchestration mechanisms to be "plugged" and triggered as services, including all current and future components, active and passive monitoring, as well as dynamic policy rules based on the obtained data to trigger adaptations not-known in advance.

Our future plans include further investigation based on real applications deployed in production environments using more powerful NFVIs that support different virtualization technologies (e.g., Kubernetes) and comparing SONATA MANO with other open-source and commercial MANO implementations.

Author Contributions: Conceptualization, P.T. and P.K.; methodology, P.T., P.K. and F.V.; software, F.V., P.K. and A.Z.; formal analysis, investigation P.K. and H.C.L.; resources, F.V.; , data curation, P.K.; writing–original draft preparation, P.T., P.K., P.A., T.S., C.P., J.B., E.F., A.Z., E.K., M.T.; writing–review and editing, H.C.L., T.Z., D.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been partially supported by the 5GTANGO (H2020-ICT 761493) and FASTER (H2020-ICT 833507) projects, funded by the European Commission through the Horizon 2020 and 5G-PPP programs.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AWS	Amazon Web services
BSS	Business Support System
CNCF	Cloud-native Computing Foundation
CNFs	Cloud-native Network Functions
CLI	Command Line Interpreter
DPD	Deployment Process Delay
EMS	Element Management System
EOE	Embedded Orchestration Engine
FG	Forwarding Graph
FSM	Function Specific Manager
GSMA	Global System for Mobile Communications Association
HA	High Availability
IoT	Internet of Things
KPI	Key Performance Indicator
LCM	Life-Cycle Management
LTE	Long-Term Evolution
NFV	Network Function Virtualization
NSD	Network Service Descriptor
NS	Network Service
NSI	Network Slice Instance
NST	Network Slice Template

NFVI	NFV Infrastructure
NFVO	NFV Orchestrator
NBI	Northbound Interface
OPD	On-boarding Process Delay
ONAP	Open Networking Automation Platform
MANO	Management and Orchestration
OSS	Operations Support System
OSM	Open-Source MANO
QoD	Quality of Decision
QoE	Quality of Experience
QoS	Quality of Service
RO	Resource Orchestration
ROD	Run-time Orchestration Delay
SLA	Service-Level Agreement
SO	Service Orchestration
SSM	Service-Specific Manager
SLA-I	SLA Instance
SLA-T	SLA Template
SDN	Software Defined Networking
SP	Service Platform
T-API	Transport API
VDU	Virtual Deployment Unit
VM	Virtual Machine
VNF	Virtual Network Function
VTN	Virtual Tenant Network
VIM	Virtualized Infrastructure Manager
VNFFG	VNF Forwarding Graph
VNFM	VNF Manager
WIM	WAN Infrastructure Manager

References

1. Flynn, K. The MobileBroadband Standard. Available online: Https://www.3gpp.org/release-16 (accessed on 19 November 2019).
2. Soenen, T.; Rossem, S.V.; Tavernier, W.; Vicens, F.; Valocchi, D.; Trakadas, P.; Karkazis, P.; Xilouris, G.; Eardley, P.; Kolometsos, S.; et al. Insights from SONATA: Implementing and Integrating a Microservice-Based NFV Service Platform with a DevOps Methodology. In Proceedings of the NOMS 2018—IEEE/IFIP Network Operations and Management Symposium, Taipei, Taiwan, 23–27 April 2018.
3. Network Functions Virtualisation (NFV)—Architectural Framework. Available online: Https://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.02.01_60/gs_NFV002v010201p.pdf (accessed on 10 November 2019).
4. De Sousa, N.F.S.; Perez, D.A.L.; Rosa, R.V.; Santos, M.A.; Rothenberg, C.E. Network Service Orchestration: A Survey. *Comput. Commun.* **2019**, *142–143*, 69–94.
5. Mijumbi, R.; Serrat, J.; Gorricho, J.-L.; Bouten, N.; Turck, F.D.; Boutaba, R. Network Function Virtualization: State-of-the-Art and Research Challenges. *IEEE Commun. Surv. Tutorials* **2016**, *18*, 236–262.
6. Rotsos, C.; King, D.; Farshad, A.; Bird, J.; Fawcett, L.; Georgalas, N.; Gunkel, M.; Shiomoto, K.; Wang, A.; Mauthe, A.; et al. Network Service Orchestration Standardization: A Technology Survey. *Comput. Stand. Interfaces* **2017**, *54*, 203–215
7. OSM Release FIVE Technical Overview. Available online: Https://osm.etsi.org/images/OSM-Whitepaper-TechContent-ReleaseFIVE-FINAL.pdf (accessed on 5 November 2019).
8. Edge Networking & Network Orchestration. Available online: Https://cloudify.co/ (accessed on 29 December 2019).

9. Pol, A.; Vilalta, R.; Munoz, R.; Vicens, F.; Carrillo, S.C.; Roman, A.; Trakadas, P.; Karkazis, P.; Kapassa, E.; Touloupou, M.; et al. Advanced NFV Features Applied to Multimedia Real-Time Communications Use Case. In Proceedings of the 2019 IEEE 2nd 5G World Forum (5GWF), Dresden, Germany, 30 September–2 October 2019.
10. Shekhawat, Y.; Touloupou, M.; Kapassa, E.; Kyriazis, D.; Xilouris, G.; Portabales, A.R.; Piesk, J.; Sprengel, H.; Gomez, I.D.; Vicens, F.; et al. Orchestrating Live Immersive Media Services Over Cloud Native Edge Infrastructure. In Proceedings of the 2019 IEEE 2nd 5G World Forum (5GWF), Dresden, Germany, 30 September–2 October 2019.
11. Caruso, G.; Nucci, F.; Gordo, O.P.; Rizou, S.; Magen, J.; Agapiou, G.; Trakadas, P. Embedding 5G Solutions Enabling New Business Scenarios in Media and Entertainment Industry. In Proceedings of the 2019 IEEE 2nd 5G World Forum (5GWF), Dresden, Germany, 30 September–2 October 2019.
12. Zahariadis, T.; Voulkidis, A.; Karkazis, P.; Trakadas, P. Preventive Maintenance of Critical Infrastructures Using 5G Networks & Drones. In Proceedings of the 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Lecce, Italy, 29 August–1 September 2017.
13. Muthanna, A.; Ateya, A.A.; Khakimov, A.; Gudkova, I.; Abuarqoub, A.; Samouylov, K.; Koucheryavy, A. Secure and Reliable IoT Networks Using Fog Computing with Software-Defined Networking and Blockchain. *J. Sens. Actuator Netw.* **2019**, *8*, 15.
14. Fröhlich, A.A. SmartData: An IoT-Ready API for Sensor Networks. *Int. J. Sens. Netw.* **2018**, *28*, 202.
15. Ahmed, K.; Blech, J.; Gregory, M.; Schmidt, H. Software Defined Networks in Industrial Automation. *J. Sens. Actuator Netw.* **2018**, *7*, 33.
16. Simoes, N. NFV Management and Orchestration: Analysis of OSM and ONAP. In Proceedings of the MAPiS 2019 1st MAP-i Seminar Proceedings, Aveiro, Portugal, 31 January 2019; pp. 1–5.
17. Nogales, B.; Vidal, I.; Lopez, D.R.; Rodriguez, J.; Garcia-Reinoso, J.; Azcorra, A. Design and Deployment of an Open Management and Orchestration Platform for Multi-Site NFV Experimentation. *IEEE Commun. Mag.* **2019**, *57*, 20–27.
18. Peuster, M.; Schneider, S.; Zhao, M.; Xilouris, G.; Trakadas, P.; Vicens, F.; Tavernier, W.; Soenen, T.; Vilalta, R.; Andreou, G.; et al. Introducing Automated Verification and Validation for Virtualized Network Functions and Services. *IEEE Commun. Mag.* **2019**, *57*, 96–102.
19. JAAS: Juju as a Service. Available online: Https://jaas.ai/docs (accessed on 29 November 2019).
20. Trakadas, P.; Karkazis, P.; Leligou, H.C.; Zahariadis, T.; Tavernier, W.; Soenen, T.; Rossem, S.; Murillo, M. Scalable monitoring for multiple virtualized infrastructures for 5G services. In Proceedings of the The International Symposium on Advances in Software Defined Networking and Network Functions Virtualization (SoftNetworking 2018), Athens, Greece, 26 April 2018; pp. 1–4.
21. Prometheus. From Metrics to Insight. Available online: Https://prometheus.io/ (accessed on 3 November 2019).
22. NFV-EVE012 Report on Network Slicing Support with ETSI NFV Architecture Framework. Available online: Https://www.etsi.org/deliver/etsi_gr/NFV-EVE/001_099/012/03.01.01_60/gr_NFV-EVE012v030101p.pdf (accessed on 3 November 2019)
23. Study on Management and Orchestration of Network Slicing for Next Generation Network. Available online: Http://www.3gpp.org/ftp//Specs/archive/28_series/28.801/ (accessed on 3 November 2019)
24. Soenen, T.; Vicens, F.; Bonnet, J.; Parada, C.; Kapassa, E.; Touloupou, M.; Fotopoulou, E.; Zafeiropoulos, A.; Pol, A.; Kolometsos, S.; et al. SLA-controlled Proxy Service Through Customisable MANO Supporting Operator Policies. In Proceedings of the IFIP/IEEE Symposium on Integrated Network and Service Management (IM), Arlington, VA, USA, 8–12 April 2019; pp. 707–708.
25. Web Services Agreement Specification (WS-Agreement). Available online: Https://www.ogf.org/documents/GFD.192.pdf (accessed on 3 November 2019)
26. Gouvas, P.; Fotopoulou, E.; Zafeiropoulos, A.; Vassilakis, C. A Context Model and Policies Management Framework for Reconfigurable-by-Design Distributed Applications. *Procedia Comput. Sci.* **2016**, *97*, 122–125.

27. Yilma, G.M.; Yousaf, F.Z.; Sciancalepore, V.; Costa-Perez, X. Tutorial Paper Benchmarking Open-Source NFV MANO Systems: OSM and ONAP. Available online: Https://www.researchgate.net/publication/332630638_Tutorial_Paper_Benchmarking_Open-Source_NFV_MANO_Systems_OSM_and_ONAP (accessed on 20 November 2019).
28. Cloudify Kubernetes Plugin. Available online: Https://docs.cloudify.co/5.0.0/working_with/official_plugins/configuration/kubernetes (accessed on 20 November 2019).