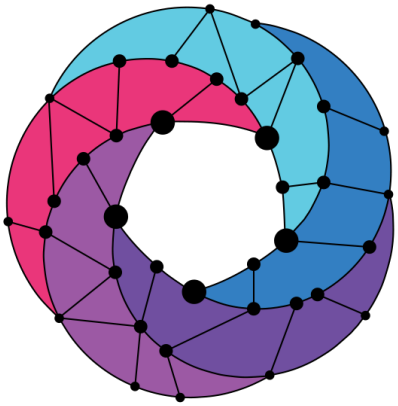


Open Source MANO

**OSM White Paper**

# OSM IN ACTION

A White Paper prepared by the OSM End User Advisory Group

**Issue 1**

**July 2021**

ETSI
06921 Sophia Antipolis CEDEX, France
Tel +33 4 92 94 42 00
info@etsi.org
www.etsi.org

# Contents

# Authors


## Editor:

Antonio Marsico, BT

## Contributors:

Andy Reid, BT

Francisco Javier Ramón, Telefónica

Gerardo García, Telefónica

# Introduction

Telecom operators work in a continuously evolving world and the end-to-end architecture of a network operator's network is never static or complete. The service cloudification promoted by NFV has increased the network flexibility but opened a series of new challenges for network operators, in particular related to the efficient management of networks composed of both mobile and fixed services combining physical and virtual devices that will require high convergence between the different infrastructure and technologies. The orchestration and management of such converged infrastructures requires comprehensive solutions that should bring together diverse vendors and technologies to foster an open infrastructure environment and avoid a series of very specialized vertical integrated solutions, which could result in higher CAPEX/OPEX and wasted computational resources.

4G and 5G mobile services are well known use cases requiring network convergence. 5G enables extensive evolutionary interworking with 4G networks, so most network operators can deploy 5G alongside existing 4G infrastructure and offer the benefits of 5G ahead of complete and fully functional deployment. Initial deployments of 5G have tended to focus on making the improved mobile data bandwidth available from the 5G air interface, where the basic end-user service – enhanced mobile broadband (eMBB) – is a straightforward development of the existing 4G service. Compared to 4G, eMBB does not change the end-user application nor the business models, and so there are fewer barriers to introduction. However, in the meantime, there are new 5G features which are more than a simple evolution of 4G. These are still maturing, notably applications based on mobile edge computing (MEC), and are going to further increase the complexity of network management and the necessity of a converged infrastructure.

The MEC and eMBB services highlight the importance of the evolutionary interworking of 4G and 5G and the need for convergence between those technologies and network infrastructure. This means that orchestration needs to work with a hybrid environment of both 4G and 5G, and network infrastructure as the mobile service will work seamlessly across them. It is important that orchestration is considered important to the mobile technology as well as the infrastructure. Virtualized versions of mobile network functions are available from a variety of sources, both proprietary and open-source, notably the 'core' function of the Evolved Packet Core (EPC).

There has been a general move to exploit containers and related technologies such as Kubernetes, as well as virtual machines in the mobile EPCs. This is leading to a more complex environment for orchestration with hybrid 4G and 5G: some functions virtualized with VMs, some with containers, and some functionality still realized as physical equipment. In addition, some operators wish to place some functionality on public cloud services, such as AWS and Azure, as well as cloud-native environments, such as Google Anthos.

The orchestration of such hybrid and hyper-converged infrastructures, composed of containers and VMs and different coexisting technologies, is one of the main challenges that a network operator faces. An infrastructure that is to be based mainly on software requires solutions to improve the service reliability and delivery time. The Open Source MANO project has already taken over this challenge in addition to demonstrating, through proof of concepts, how it is possible to have a single

orchestration and management layer for such complex infrastructure. We also envisage a possible future application for OSM in the context of a CI/CD DevOps system for NFV.

This white paper highlights real-world applications where OSM orchestrates the deployment of a complex mobile core and how it could be exploited to enable novel NFV paradigms based on DevOps. We discuss the practical realization and how OSM works 'in action'. This white paper is organized as follows: Section II provides an overview of the challenges that network operators face in a converged cloud-native infrastructure and how OSM addresses these challenges. The use cases, namely the management of a mobile core and the proposal for an NFV CI/CD DevOps infrastructure with OSM, are presented in Section III. Section IV concludes the white paper.

# The Challenges for Converged Cloud-Native Infrastructure

The introduction of 5G is opening a variety of new challenges to network operators. The virtualization technologies that underpin 5G are going to change the nature and composition of networks. However, 5G and virtualization cannot be introduced in a manner completely independent of existing networks and their Physical Network Functions (PNFs). These PNFs will coexist with Virtual Network Functions (VNFs) and Containerized Network Functions (CNFs), and need to be managed as a single converged telco network. This scenario can be defined as a *Converged Cloud-Native Infrastructure*, and, in order to exploit the benefits of novel technologies, this infrastructure will require more functions to be close to the customer edge; consequently, specific workloads may be highly distributed.

This new network environment will have increased management complexity, which was already a challenge. The BSS/OSS systems need to be adapted to interact with both the legacy and new environment. In the first instance, this complexity can be tackled by breaking the task down into several hierarchical management domains through increased abstraction with respect to the detailed control of a network function. These hierarchical management domains should then be integrated to build a single, complete network management plane (OSM EUAG Integration White Paper) in which both core and edge workloads (amongst others) are managed together.

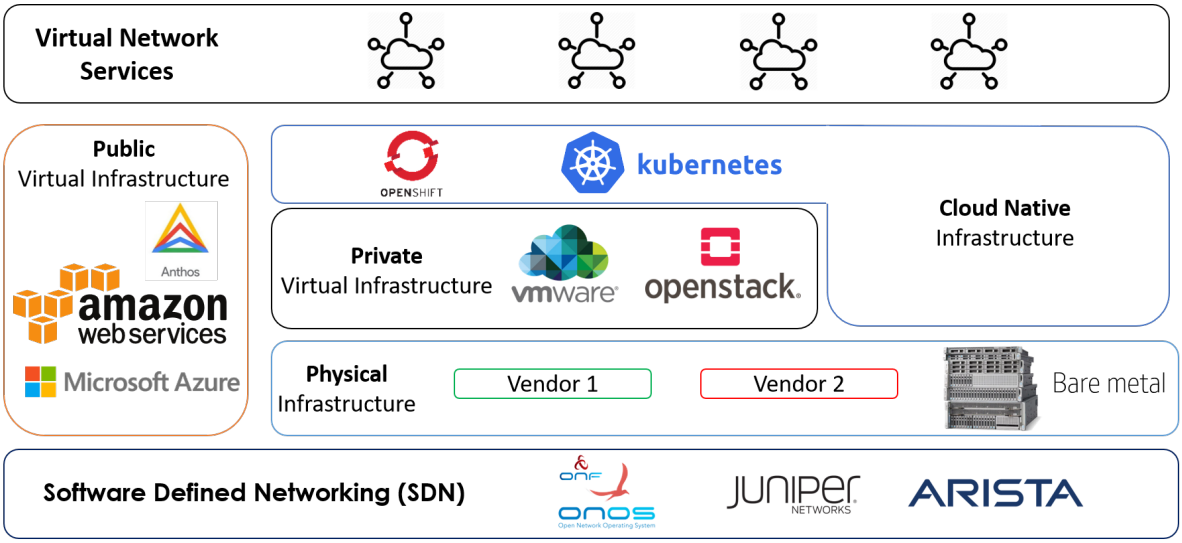
An NFV Management and Orchestration (MANO) system is the central part of this complex network architecture. It should be able to cope with several specific challenges that a network operator will face:

- **The hybrid infrastructure challenge**: how to manage a general network that comprises a mixture of infrastructure and network function architectures, such as CNFs, VNFs, and PNFs
- **The management of multi-vendor NFV infrastructure**: how to cope with supply chain diversification requiring the management of multiple vendors
- **Moving from vertically integrated to open-source solutions**: how to simplify the management of a next-generation converged core, and facilitate moving from vertically integrated and closed-source solutions to open-source ones

## The Hybrid Infrastructure Challenge

The NFV infrastructure of a network operator is continuously evolving together with the architectures of Network Functions. At the beginning of the NFV journey, network operators tended to create only *Private Virtual Infrastructures* based mainly on Virtualized Infrastructure Managers (VIM) running the network functions on VMs or dedicated servers hosting them. Nowadays, in a network operator environment the number of use cases for NFV has increased and the demand for more infrastructure flexibility is a strict requirement. The network functions are no longer just firewalls or routers based on VMs but can instead be highly distributed monitoring systems based on CNF that could involve intense data analytics. As such, the efficient and cost-effective management of such infrastructure and network function architectures, which we define as *the hybrid infrastructure challenge*, is a crucial challenge that network operators are already facing today and will increase in complexity in the near future.

The figure below defines and depicts several types of NFV infrastructure: the Physical Infrastructure, Virtual Public/Private Infrastructure, and the Cloud-Native Infrastructure.




NFV Hybrid Infrastructure

The Physical infrastructure is the “bedrock” of the whole NFV infrastructure. It is composed of bare metal servers and physical network devices. The management of a physical infrastructure is outside the scope of ETSI MANO architecture but must be considered when an NS is deployed, e.g. the creation of an overlay network intra- and inter-datacentres. The physical infrastructure could take advantage of the flexibility introduced by Software-Defined Networking (SDN) to manage physical network devices. This can be used to interconnect not only the private part of the infrastructure but also the public one, e.g. exploiting an SD-WAN solution.

The Virtual infrastructure is represented by the private and public cloud providers. They are managed via a VIM in the ETSI NFV MANO architecture, and provide abstraction and quota management with respect to the physical computational resources of the infrastructure. A network operator can use a mixture of private and public clouds, and should be able to manage them

efficiently, e.g. by moving part of the NSs to a public cloud if more resources are required than available in the private cloud.

Finally, there is the Cloud-Native infrastructure. It is represented by container-based orchestration solutions that can run both on a Private Virtual Infrastructure by means of VMs or directly on bare metal. Container-based solutions may be a target solution for many networking problems as they can simplify scalability and improve resilience. In reality, they will probably not be able to completely replace PNFs and VNFs, or at least not for some time. VNF vendors are more likely to offer hybrid solutions rather than fully cloud-native ones. The management of this infrastructure is more complex and is likely to require the direct support of a hybrid environment.

In such a complex network environment, where multiple types of NS need to be deployed, NFV MANO is one of the most important components in realizing the cost savings and service flexibility that virtualization promises. The MANO component should be capable of instantiating Network Functions (NFs), with different architectures (whether PNF, VNF, or CNF) to a multitude of infrastructures in a seamless way, thus reducing the need for costly ad hoc integration and speeding up the introduction of innovative services for customers. For instance, a network operator should be able to integrate VM and Container-based solutions and control them from a single management plane.

Open Source MANO (OSM) has all the capabilities required to manage this hybrid network infrastructure.

Firstly, it supports interfacing with multiple types of VIMs that can be, for example, private clouds using OpenStack and public clouds like Amazon Web Services (AWS) and Microsoft Azure, and the consumption of resources in these different environments is transparent to the user through OSM. When an NS is deployed on a selected VIM, OSM takes care of all the steps required for deployment. An NS can also be deployed between a private and public cloud based on latency requirements by exploiting the Placement Module (PLA), which was introduced in Release EIGHT.

Secondly, OSM also supports the establishment of overlay connectivity intra- and inter-datacentres by exploiting an SDN-based system. The intra-datacentre connectivity is managed by the SDN-assist feature of OSM and several SDN controllers are supported, in particular ONOS, Juniper Contrail, and Arista. The *WAN Infrastructure Manager* (WIM) is a specialized system that provides the inter-datacentre connectivity via NFVI-POPs, as defined in the ETSI NFV architecture. Additionally, in this case, the WIM relies on the SDN controllers supported by OSM.

Thirdly, OSM natively supports the interaction with Cloud-Native Infrastructures. The support is principally focused on Kubernetes infrastructure, but this can be easily extended to include other environments as required. Importantly, the OSM Information Model (IM) has been fully extended to support CNFs. A new *Deployment Unit*, defined as *Kubernetes Deployment Unit (KDU)*, represents one core component of the CNF to be deployed. The OSM CNF support is based on *Application Managers*. OSM exploits two different *Application Managers*: Helm and Juju.

The former is the de facto standard for container deployment on Cloud-Native Infrastructure and its model; the *chart* describes how container-based applications can be installed and upgraded. OSM uses the *helm repositories* to simplify the deployment of charts. The users can take advantage of

many public chart repositories to deploy a multitude of applications. As well as accessing well-known public chart repositories, often using helm charts, the chart repositories could also be private and hosted within the NFV infrastructure. A network operator can therefore create and deploy custom charts.

The other type of *Application Manager* for CNFs supported by OSM is Juju, which is an automation tool that simplifies the deployment of software applications. It is based on *charms*, software packages that manage the deployment and configuration of applications. Juju is an alternative option to Helm and has the advantage that it can be used to manage the whole lifecycle of the application as well as the relations between available services of different applications. Importantly, Juju charms can be used to modify the configuration of a container-based application at runtime and make these hooks available to the wider OSS environment through OSM NBI. This can greatly reduce the overall integration work needed for a network operator to fully support virtualization.

The support of multiple types of infrastructures and NF architectures is fully available in OSM. This enables telco operators to exploit OSM to simplify the management of their NFV infrastructure. The advantage is that a network operator can reduce both CAPEX and OPEX since it can rely on a single orchestration system that is capable of managing many NSs, whether they be composed of VNF and CNF or hosted between private and public clouds, and container orchestration solutions.

## The management of a multi-vendor NFV infrastructure

In an NFV infrastructure, it is expected that many different vendors will coexist together thanks to the generic hardware architecture that can host many diverse NFs. The management of multiple vendors is challenging because every NF has its own management interface that may not be standardized even if some standardized framework is used, e.g. HTTP RESTful, SSH, NETCONF, etc. For instance, the firewall of *vendor 1* could support HTTP RESTful APIs but the way of changing a configuration management is different from the firewall of *vendor 2*.

The configuration of NFs is the most complex part for MANO. The configuration information model and configuration primitives differ between vendors even if the type of communication interface conforms to a specific standard. The MANO solution should handle all the configuration steps, defined as Day 0, Day 1 and Day 2, and their vendor-specific differences.

The process of ‘standing up’ an NF (the Day 0 configuration) requires a substantial amount of initial configuration. This will normally include at least the configuration of the network interfaces, basic security, and the configuration of a communication framework. This can be achieved by several different solutions, such as cloud-init. However, if a non-standard tool is used by an NF vendor, the complexity increases and some manual configurations are required to make the NF accessible.

Once the communication framework is enabled and the NF is accessible on the management network, it could be set up with the required service configuration. This step is defined as Day 1 and requires configuration via a vendor-specific information model. For instance, a VyOS router could be configured with a static route by using SSH as the communication interface and a set of specific shell commands to edit the configuration.

The NFV requirements in this regard do not stop at Day 1. Once the NF is active, it will still need active management, including further configuration as well as continuous health monitoring. Again,

the interfaces for these ‘in-life’ functions (Day 2 configuration) tend to be vendor-specific. A possible example is the creation of a new firewall rule.

While some orchestration systems have simply ignored the Day 1 and Day 2 configuration problems and only sought to ‘stand up’ a VNF, this cannot meet the orchestration requirements for network operators for a multi-vendor environment. The consequence is that the NFs must interface with proprietary vendor-specific network function managers and integration is left as an ad hoc, higher-level task for somewhere in the OSS. OSM, on the other hand, has full support for Day 1 and Day 2 configuration of NFs from different vendors while presenting a single common abstracted interface for the Day 1 and Day 2 configuration to the wider OSS through the OSM NBI.

OSM uses Juju to manage the VNF configuration through so-called *reactive* charms, i.e. charms that expose *actions* that can be executed on demand. OSM exploits the *actions* to perform Day 1 and Day 2 configurations on a VNF. The charms are effectively a form of script (typically written in Python) with actions that can be common, abstracted and vendor-independent, and translate these actions into the vendor-specific commands for each vendor's NF. Juju automatically and transparently handles all execution, addressing and other message handling. Juju means that charms can be easily customized to support the specific communication protocol of a VNF (or VNFM if that is the way the vendor has designed the VNF). The charms are part of an OSM NF package and they can be easily applied to any OSM instance.

## Moving from vertically integrated to open -source solutions

The management of such a diverse NFV infrastructure requires comprehensive orchestration solutions that reduce the complexity of network management. Vendors usually propose vertically integrated solutions for managing their products which are rarely flexible enough to facilitate integration with other vendors. This gives rise to a set of strategic and technological issues for operators as they have to decide between investing in several isolated management solutions or a single orchestration framework for their NFV infrastructure.

One option would be for operators to run several orchestrators, each one managing a specific part of the NFV infrastructure allocated to a specific vendor. This limits how the resources can be shared with other parts of the infrastructure. These orchestrators are localized and manage their own NFs and a specific quota of hardware resources that was pre-assigned in the provisioning step. This option greatly limits flexibility and limits the way the network can evolve. Such an option would not help to properly realize one of the main benefits of the virtualization, which is the separation of hardware resources for the functional requirements of the network. It is widely observed that suppliers of proprietary solutions often seek to supply as much of the vertical ‘stack’ as possible, running counter to the prime objectives of virtualization.

An alternative to vertically integrated solutions is horizontal orchestration based on open-source solutions. Open source is naturally open, based on standard and open interfaces that can be used in an open architecture. Solutions like OSM move towards this direction since they can provide a single management plane based on standard interfaces and models (e.g. ETSI ISG NFV SOL specifications) for both an NFV infrastructure and NFs. The interaction with NFV infrastructures is achieved by implementing specific adapters (e.g. VIM connectors in OSM), while the interaction with NFs exploits

package artifacts (e.g. charms in OSM) that instruct the orchestrator how to modify the Day 1/2 configurations of NFs.

Software openness gives operators the opportunity to easily test these orchestration solutions and customize the software based on their specific requirements. Moving to open source for a network operator may require a careful evaluation of the different solutions available. There are several important requirements, including software maturity and readiness for production environments (i.e. software stability and scalability), the ease of solution customization, and integration into the network operator's current OSS/BSS systems. Network operators could overcome the open-source entry barriers by increasing their internal software skills and improving collaboration with independent system integrators, which could help with the integration and customization of the solution.

# OSM takes on the challenges of network operators

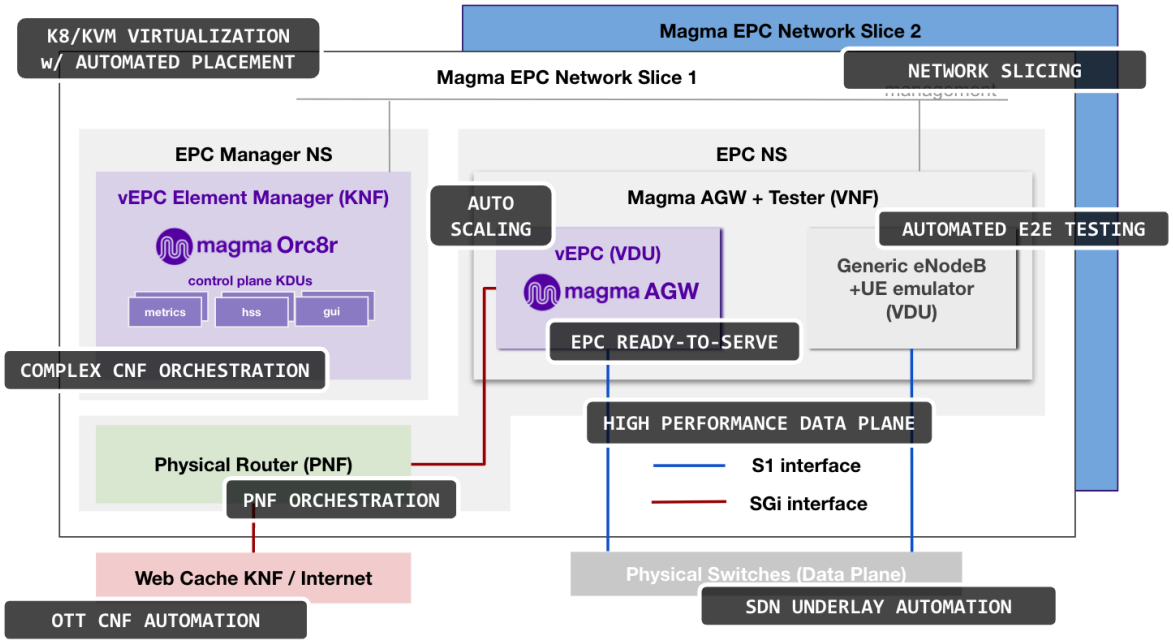
Over the last four years, the Open Source MANO community has always given particular attention to the delivery of a carrier-grade MANO solution that addresses both the new service requirements of telecom operators and support for the smooth evolution and integration of their environments towards a future-ready infrastructure. OSM Release NINE already offers a range of features that can cope with the most demanding NFV production infrastructures, with particular attention given to the challenges of a *Converged Cloud-Native Infrastructure*.

In this section, we describe two different use cases that show the present and the future of OSM, with particular focus on its advanced capabilities. The first use case demonstrates OSM deploying and managing a mobile virtual Evolved Packet Core (vEPC) based on the open-source project MAGMA Core. It offers a broad overview of the supported advanced capabilities of OSM today, and is publicly available. The second use case presents what we envisage as a natural use case of OSM in which it could be part of a CI/CD DevOps system of NFV infrastructure.

## The present: OSM manages a 4G converged network core

The Open Source MANO community Hackfests are events that provide an overview of the OSM capabilities by presenting proof of concepts from real-world use cases. Participants have the possibility to test OSM on a real-world use case and replicate it with a series of hands-on sessions.

One of the most important examples featuring in the Hackfests is the management of a virtual Evolved Packet Core (vEPC) based on the open-source project MAGMA Core. This project aims to provide an open and free EPC to build accessible cellular networks.




NFV Hybrid Infrastructure

The MAGMA Core use case is an important example for network operators, since the management of a mobile network core is one of their main challenges. The continuous evolution of mobile technologies, from 3G to 5G, has brought many architectural changes that should coexist and be managed within the same network environment. From separated communications channels, calls and data, to a converged infrastructure in which all traffic flows through a packet-based network.

Starting from the advent of 4G, the number of vendors offering a virtualized version of their Evolved Packet Core (EPC) solutions has increased. The newest versions of vEPCs are based on a cloud-native architecture, in which both VNFs and CNFs coexist to get full advantage of scalability and resiliency. A virtual EPC with such complexity requires a MANO solution that is able to deploy, configure and manage all the different components in a single management plane to avoid the lock-in with vendor-based vertically integrated solutions and to keep the cost under control.

The MAGMA Core demonstrated many advanced capabilities of OSM: (i) the orchestration of both VNFs and CNFs, (ii) the management of Physical Network Functions (PNFs), (iii) the integration with SDN controllers, (iv) slicing, (v) support of high-performance deployments with SR-IOV interfaces, (vi) VNF autoscaling, and (vii) management of Day 0, Day 1 and Day 2 configurations.

The MAGMA use case requires a VIM and a Kubernetes cluster controlled by OSM to be deployed. The MAGMA network service is composed of several different components:

- **Physical Router (PNF)**: This is a physical device already deployed in the network. In the case of OSM's Hackfest, this element was provided by a standalone VyOS virtual router, but it could be any physical device that acted as a gateway to the external network (e.g. a firewall, border router, etc.). As such, a PNF can be managed and configured by OSM by exploiting the OSM

VCA (VNF Configuration Agent). In this case, a Juju charm performs the Day 1 and Day 2 configurations on the router, in particular the setup of ACLs.

- **Magma AGW**: The Magma Access Gateway implements a VNF version of the 4G EPC; specifically, the *Servicing GW*, the *Mobility Management Entity (MME)* and the *Public Data Network (PDN)* GW. In this case, a specific charm performs all the configurations required to set up a mobile network on the AGW.
- **Magma Orc8r**: This provides a container-based management system for the Magma AGW. The network administrator is able to access the Magma Web GUI, visualize the network metrics from eNodeB to AGW, and modify its configuration.
- **eNodeB and UE emulator**: This VNF is based on the open-source project SRS LTE and emulates the wireless part of the system since there is no hardware component for the Radio Access Network (RAN) in this demo setup. The emulator replicates both the *eNodeB* and the *UE* signalling. From the UE emulator, it is possible to generate traffic and reach the Internet crossing all the mobile network components. A native charm of Juju, a charm that is deployed on the VNF instead of on a separated LXD or Docker container, manages the eNodeB and UE software deployment and configurations.
- **SDN physical switch**: When a VNF requires an SR-IOV port, this is directly attached to the compute host ethernet interface. A VLAN is the simplest means of isolating the NS traffic. This requires a switch that is already configured to allow VLAN traffic on the switch port. However, a more flexible approach could be to use an SDN switch which could enable that connectivity on demand upon NS instantiation (and disable it on NS removal). OSM supports the configuration of SDN networks by interacting with an SDN controller with its *OSM SDN Assist* feature, enabling this scenario quite naturally with no effort for the orchestrator operator.
- **Web cache based on Squid proxy** **Squid proxy**: In this demonstration, both direct or proxy-based Internet access is supported. The community developed a CNF package and a charm for OSM to deploy a Squid proxy cache on a K8s cluster. OSM manages the configuration of the cache; in particular, the addition and removal of authorized URLs to and from the proxy.
- **Magma Slice**: The deployment of NS is managed by a slice template. The OSM slicing provides a system to share NSs between different slices. The Magma slice is composed of the *Magma AGW* NS and the *Magma Orc8r* NS. The Magma AGW NS is flagged as “shared” while the Magma Orc8r is flagged as “not shared”, and as a result the deployment of another slice reuses the *AGW* and deploys another *Magma Orc8r* on a Kubernetes cluster.

### Configurations Performed on Day 1 & Available Configurations for Day 2

After the slice deployment, OSM performs several Day 1 configurations to conclude the setup of the Magma vEPC, where the Magma AGW and the eNodeB/UE emulator are configured by the Juju charms forming part of the NF packages. As such, OSM automates the steps that should have been performed manually by a human operator on the Orc8r Web GUI.

On the Magma AGW, OSM performs the following configurations:

1. Creation of a new virtual network: in the Magma system, it is a name that identifies the new mobile network and groups all the related configurations.
2. Registration of the AGW to the Orc8r: this configures the communication channel between the Orc8r and Magma AGW.
3. Creation of a new mobile subscriber: the charm sets up a new mobile user on the HSS (Home Subscriber Server).

On the eNodeB/UE emulator, OSM configures the IP subnet where the eNodeB/UE is able to find the Magma AGW.

There are a few Day 2 configurations – these are the configurations performed dynamically after the system setup that are supported by OSM on the Magma vEPC, and that can be extended based on the requirements of the mobile network operator:

- *Physical Router Firewall Configuration*: This Day 2 configuration is used to add a new IP address to the ACL that allows traffic from the *SGi* interface to the Internet.
- *Attach a UE to the Magma AGW*: This creates a virtual link and authenticates the UE on the Magma AGW. The inputs are the USIM parameters, such as the USIM-MSI that should already be present on the HSS.
- *Add Allowed URL to Squid*: This adds a new allowed URL to the Squid Cache running on the Kubernetes cluster. It is an important example of how OSM could interact not only with VNFs but CNFs too. The input parameter is the URL to be added.

### Monitoring and VNF Autoscaling with Closed -Loop Automation

From the earliest releases, OSM provides a built-in monitoring system for NS. It relies on Prometheus as a time-series database and Grafana for the data visualization. The information can be collected using connectors for VIM-related metrics and by the so-called *OSM Execution Environments (EE)* that poll the VNFs for application-specific metrics. The VNFD provides instructions about the characteristics of metrics and how they should be collected (VIM or EE).

In addition, there can be specified autoscaling actions for Virtual Deployment Units (VDUs) that form part of the VNFD in the event of a metric exceeding a certain threshold. This enables a Closed-Loop Automation mechanism.

The Magma use case takes full advantage of the monitoring and autoscaling features of OSM. The VNFD of the Magma AGW defines the collection of several parameters: CPU and RAM utilization, and network interface traffic. An autoscaling option is configured when the CPU occupation is over 80%, and a downscaling option when the CPU falls below 10%. In addition, the descriptor offers an alarm option for a particular metric. In this case, a URL GET request is triggered when the CPU exceeds 80%.

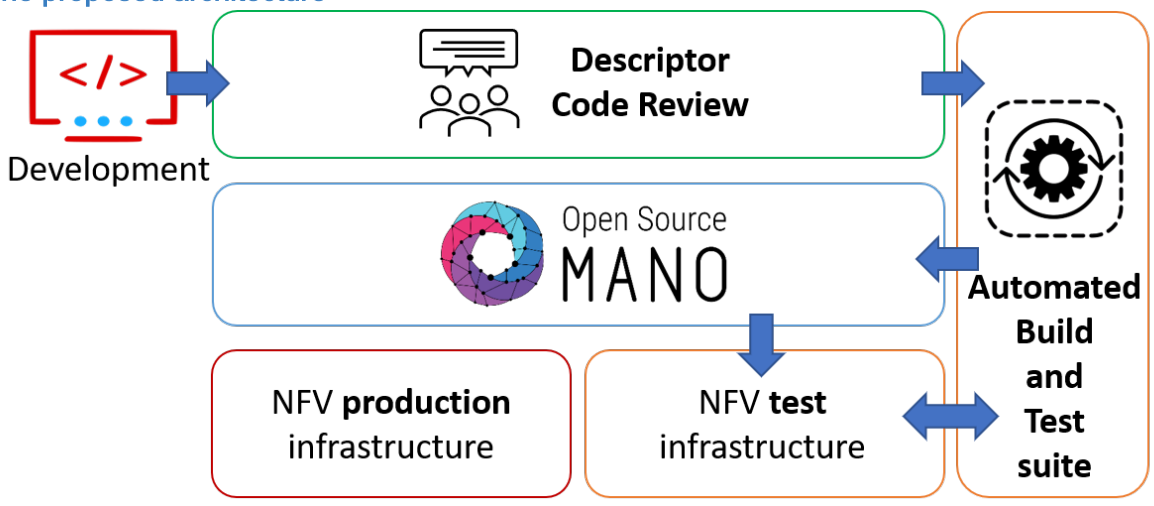
## A key component for agile delivery: CI/CD DevOps in a network operator environment using OSM

Continuous Integration / Continuous Delivery (CI/CD) is becoming a highly recommended practice for offering faster software delivery. The code is continuously developed, tested, and integrated in a master development repository, based on a version control system such as Git. The CI/CD pipeline, if backed by a high-quality and extensive test suite, becomes a key system to grant ready-to-use software that always has the latest features available and, at the same time, proper guarantees of stability.

Network operators are constantly looking at reducing costs while providing new services to their customers. This has led operators to increase software adoption in their networks, in particular exploiting SDN and NFV infrastructures managed by a MANO solution to simplify deployment and management. However, network operators face delays in introducing and upgrading new services. Every service requires extensive testing before going into production to avoid unexpected downtime and to provide customers with service assurance.

How the CI/CD DevOps software development model could be exploited to work in a network environment is currently under careful evaluation by network operators. Open discussions in telecom industry forums and standardization bodies are currently underway to understand if such a model could offer any advantage to service delivery in NFV infrastructure. Based on its extensive use in lab environment by some service providers in OSM EUAG, we believe that OSM can easily become one of the main components of a CI/CD model for NFV, and therefore propose a possible architecture and workflow for this use case.

### The proposed architecture




Proposed DevOps CI/CD pipeline with OSM

The proposal for a CI/CD model for NFV is substantially inspired by the well-known software CI/CD that is usually composed of a collaboration tool for code review, and a build, test and deploy automation tool. In this architecture, we propose an adaptation of the standard CI/CD to a model that could be used in more complex environments, such as an NFV infrastructure to deploy and manage NFs.

In our view, OSM is very well equipped to become a core component of such architecture, since it already has many capabilities that could be exploited and easily integrated with other tools, and provides the highly valuable service of creating network services on demand. OSM could provide the MANO capabilities in an NFV CI/CD pipeline by managing the pre-deployment tests and the entire lifecycle of NFs when the service is deployed in the live infrastructure.

The generic architecture for a CI/CD in NFV could be composed of the following components:

- *Descriptor Development*: Instead of developing software, "NFV" developers typically prepare VNF and NS Descriptors (VNFD and NSD) to be uploaded to a repository and are revised by other developers and repository owners.
- *VNF and NS Descriptor Code Revision*: In the standard CI/CD software development, the code should be reviewed by other developers and tested before being integrated into the master code. In an NFV scenario, we believe that a VNFD/NSD revision is required before creating or changing any NS running in the infrastructure. There are a number of tools that can be used for the review process, from the mainstream Gerrit to the trendy Gitlab.
- *Automated Build and Test suite*: This provides all the automation for managing the VNFD and NSD test and build. However, there is increased complexity with respect to a standard CI/CD pipeline. The VNF and NS packages should be first deployed in an *NFV test infrastructure* and then evaluated against functional and integration tests. If the tests are passed, then the VNF and NS packages can be flagged as "test passed". The tool should provide a modular architecture where multiple plugins can be added to manage the interaction with the MANO layer and manage the functional tests. In the standard software CI/CD, OSM Community uses Jenkins, which is one of the most extensive choices for this purpose, but there are a good number of CI/CD tools that might be equally fit for this purpose.
- *MANO orchestration (OSM)*: This interacts with the *Automated Build and Test suite* to deploy NSs to both the *NFV test infrastructure* and *NFV production infrastructure*. It could be used to collect the performance data and execute Day 1 & 2 configurations during the integration and functional tests, and manage and orchestrate the entire production infrastructure.
- *NFV test infrastructure*: This is the Virtualized Infrastructure Manager (VIM) in which the NS defined by VNFD and NSD could be tested before going live into the production network.
- *NFV production infrastructure*: This is the production VIM that provides NSs to customers. The NSs should be deployed only after the build and test phase is passed.

### Workflow

The CI/CD in NFV presents several challenges and may require a different workflow with respect to the standard DevOps. For instance, the build and test process of an NS is different from that for software: an NS needs to be evaluated against several functional tests on live infrastructure to ensure that there are no misconfigurations affecting a customer service. In addition, each test depends on the type of VNF that you are testing: a router cannot be validated in the same way as a mobile network core.

In general, a source version control, such as GIT, can be the foundation of the NFV CI/CD pipeline. This offers more granular control of every modification to VNFD and NSD, and the possibility to "roll back" a failed configuration. A GIT repository could be associated with an NS for a specific customer, where multiple VNFDs and NSDs could be uploaded to it, and a specific pipeline could be associated with it. Each pipeline could manage a specific tenant on both the *NFV test infrastructure* and *NFV production infrastructure*.

In the next part of this White Paper, we shall try to define what the workflow should be and what the challenges are for the most common operations required by a network operator in its NFV infrastructure.

Firstly, we describe a common workflow for the NFV CI/CD, divided into three different phases: *VNFD/NSD Development and Review*, *Build and Test*, and *Release*. In the last phase, we shall define the processes for three use cases: *new NS deployment*, *NS update*, and *NS rollback*.

### VNFD/NSD Development and Review phase

When the VNFD and NSD are ready, the developer could upload them to the *Descriptor code review*, backed by a GIT repository. The possible working principle is defined as follows:

- *VNFD upload*: A VNFD is uploaded to the *Descriptor code review* tool backed by a GIT repository. This enables the team responsible for the customer NS to collaborate and review the descriptor. Firstly, we expect that the VNFD upload triggers some initial (unit) tests, such as the descriptor validation with respect to a reference model (e.g. ETSI NFV SOL006) to check whether there are descriptor errors. It could trigger other tests if an NSD already exists in the repository and this VNFD forms part of it, such as integration and functional tests of the *NFV test infrastructure*.
- *NSD upload*: The upload of an NSD requires a different workflow and configuration files with respect to the VNFD one. When an NSD is uploaded, it should trigger more complete tests of a new NS composed of one or more VNFDs and their specific Day 0, Day 1 and Day 2 configurations. These tests could begin with basic unit tests (e.g. NSD model validation and availability of all the associated VNFDs) and then move to the more complex integration and functional tests.

### Build and Test phase

The *Build and Test* phase is the core of a CI/CD pipeline. An NS composed of an NSD and several VNFDs is evaluated using a set of tests, starting from unit tests to the more complex integration and functional tests. During this process, if the basic unit tests for the descriptor validation are passed, the VNFD and NSD packages can be built and the next tests can start.

The NFV pipeline, which differs from a software CI/CD, should be flexible and must be appropriate to the type of NS being tested. We expect that the test configurations can be part of a config file that should be linked to the VNFD and NSD, and they should provide some specific parameters to manage the integration and functional tests of the pipeline. For instance, in the case of a firewall, one functional test configuration could be the type of traffic allowed to transit across the firewall and the incoming/outgoing interfaces from/to.

Here we define what the integration and functional tests could do in an NFV CI/CD pipeline:

- *Integration tests*: This is the most challenging part of the entire workflow and we define how the system should work in principle. After the package build, the tool uploads the VNFD/NSD packages to OSM in a project different to the production one or to a different OSM instance used for testing. Then, the test suite triggers the deployment of a new NS with the config parameters of the config file. If the deployment is successful, then the test and build phase for the NS is passed. The test suite requires specific plugins to handle this NFV CI/CD workflow that are currently not available and have to be created from scratch.

- *Functional tests*: This is the most complex test, both for a standard and an NFV DevOps system. In general, after the deployment of an NS, a network operator needs to be sure that the service satisfies customer expectations. A set of functional tests are required to make sure that the NS is stable and provides the correct SLA and right configurations. The test and build suite requires specific plugins to handle the entire functional test lifecycle. These plugins have to each support a specific type of test, i.e. traffic flow, traffic validation, resilience, etc. A vendor of network test systems could take the opportunity to develop and support the plugins that could form part of *Automated Build and Test suite*.

### Release

The *Release* is the last phase of the pipeline and, as with the software CI/CD, it should start when a new merge request is accepted by the repository managers and a new version of the VNFDs and NSDs is integrated into the main working branch (e.g. master) that represents the current NS running in the NFV production infrastructure. This phase is the most difficult one because there is not only a *greenfield* scenario, which is a *New NS deployment*, but there are also *brownfield* scenarios that could occur when an NS is already running, such as *NS update* and *NS rollback*. We describe them as follows:

- *New NS deployment*: The deployment of a new NS is the simplest use case for a network operator. After the test and build phase is passed, a new NS should be ready for deployment in the production infrastructure. As with the software CI/CD, we may expect that the NS developers would “merge” the VNF/NS package into the master repository from the code review tool. Then, manually or automatically, the new NS could be deployed to the production infrastructure.

- *NS update*: Updating an NS is more complex with respect to a greenfield deployment. This is a functionality that we expect should be part of a MANO solution not the CI/CD itself. However, we describe a workflow that would combine features required on the MANO and CI/CD pipeline sides.

Firstly, we have to define different types of updates: *single VNF* or *an entire NS*. In the former, the update could affect either the descriptor or the VNF image on the VIM or both. We consider the simplest case, the VNF package update, since the upload of the VNF image is usually performed directly on the VIM. The descriptor update may involve several components, such as the connection points, the VNF CPU and memory characteristics (e.g. EPA, number of CPUs, etc.), the supported config operations, etc. In general, due to the complexity of the operation, the safest procedure is to

stop the affected VNF and recreate it. In the latter, an entire NS update, the procedure should be the same. We advise that all considerations to upgrade without downtime should be taken into account but we do not discuss them in this proposal.

The CI/CD pipeline should take into account the existence of an NS already running. It means that the NFV test infrastructure should be first deployed with the old VNF/NS package versions and then the update should be performed and tested. On the production infrastructure, we would expect that the MANO solution should take care of the update automatically.

- *NS rollback*: This operation is required when a misconfiguration is found on the NFV infrastructure and it is necessary to restore the previous configuration. The procedure to follow should be the NS update, since an NS is already running. The NS developers could take advantage of the source control under GIT and "revert" the last commit that produced the misconfiguration. Eventually, the previous version of the service is restored by following the NS update procedure.

# Conclusion

Network cloudification has given rise to new Management and Orchestration (MANO) challenges for network operators. The network environment is set to be composed of many diverse technologies, such as Network Functions (NFs) based on VMs, containers or physical devices, and types of infrastructure to which the virtual functions could be deployed, for instance, private clouds, public clouds, and cloud-native. All these technologies and infrastructure types have different requirements and methods of configuration and deployment (e.g. APIs, communication framework, etc.).

A horizontal MANO solution able to control more than one technology and infrastructure type is a potential solution to provide a simplified and more efficient infrastructure management. A network operator could decrease the number of vertically integrated solutions that are usually provided by a single vendor and manage a specific portion of the network infrastructure or technology. A reduced number of vertical solutions could help the transition to a novel network environment that could be composed of multiple vendors and take advantage of open interfaces and MANO solutions. As a consequence, a network operator could hope for more competitiveness in the supply chain and a possible reduction of CAPEX and OPEX.

In this white paper, we have demonstrated that Open Source MANO (OSM) already has all the features to cope with such a diverse network environment and is ready to form part of a future NFV infrastructure based on a CI/CD DevOps paradigm. It can orchestrate a mobile EPC composed of VNFs, CNFs and PNFs, and deploy these NFs to private/public clouds and a Kubernetes cluster. We propose a possible architecture and implementation for a CI/CD pipeline for NFV that integrates OSM as the MANO solution for deploying and configuring the NFs. Orchestration is one of the keys to cost-efficient network management and OSM is ready to take over all the challenges, both today and in the future.


ETSI
06921 Sophia Antipolis CEDEX, France
Tel +33 4 92 94 42 00
info@etsi.org
www.etsi.org