OSM Release FOUR Technical Overview

May 2018

ETSI (European Telecommunications Standards Institute)
06921 Sophia Antipolis CEDEX, France
Tel +33 4 92 94 42 00
info@etsi.org
www.etsi.org

Authors

Adrian Hoban	Intel
Adam Israel	Canonical
Alfonso Tierno Sepúlveda	Telefónica
Claire Boyer	ETSI
Francisco Javier Ramón Salguero	Telefónica
Gerardo García de Blas	Telefónica
Gianpietro Lavado	Whitestack
Mark Shuttleworth	Canonical
Matt Harper	RIFT.io
Michael Marchetti	Sandvine
Silvia Almagia	ETSI
Vanessa Little	VMware

Contents

1 Introduction to Release FOUR

The OSM community has continued diligently on a path of delivering a world-class production ready solution. OSM Release FOUR represents the fifth major release of OSM yet and another significant step on this journey [Ref 1]. It has been engineered, tested and documented to be functionally complete to support Operator RFx processes, and to be a key component for internal/lab and external/field trials as well as interoperability and scalability tests for virtual network functions and services. It allows for rapid installation in VNF vendor, system integrator and operator environments worldwide.

Building on the award winning capabilities[1] developed for the prior releases, Release FOUR brings a large set of new features and enhancements, this version is the most ambitious and innovative OSM Release to date and constitutes a huge leap forward in terms of functionality, user experience and maturity. This new Release brings substantial progress thanks to a number of architectural improvements, which result in a more efficient behaviour and much leaner footprint - up to 75% less RAM consumption. Additionally, its new northbound interface, aligned with ETSI NFV work, and the brand-new cloud-native setup, facilitate OSM's installation and operation, while making OSM more open and simpler to integrate with pluggable modules and external systems, such as the existing OSS.

The thought process that goes into defining the OSM direction continues to remain true to the vision of making NFV easy to use and to provide choice to operators. Release FOUR feature that have progressed that vision include:

1. New northbound interface, aligned with ETSI NFV specification SOL005, providing a single pane of glass to control the OSM system. To facilitate the interoperability with existing and new client applications, the details of the interface are entirely available in OpenAPI format.
2. Cloud-native install that brings a large set on improvements in terms of user experience and optimization. With a reduced footprint, this release is much faster, more responsive and stable than ever, and provides more convenient means for event logging and diagnosis. In addition, a new graphical user interface (GUI) and a set of optional components can be installed on top of the core components to enrich the experience in terms of visualization, VIM sandbox, etc.
3. Monitoring and closed-loop capabilities have also been extended. On-demand and (in an upcoming point release) descriptor-driven setting of alarms and metrics are now much simpler and convenient to configure and consume. Likewise, the support of push notifications and configuration of reactive policies, via the new Policy Manager, opens the door to closed-loop operations.
4. Modelling and networking logic have also been enhanced. The full support of IP profiles, the consistency checking of addressing, the ability to configure specific MAC addresses for non-

[1] OSM won the 2017 Layer123 Network Transformation Award for **BEST OPEN SOURCE DEVELOPMENT.**

cloud-native VNFs, the support of service function chaining assisted by the VIM, or the possibility of using alternative images for public clouds, are only some examples. It also sets a solid path for the support of native charms - so that conventional Juju applications can run natively in OSM - and for the seamless management of Physical Network Functions (PNFs) or hybrid components (physical and virtual), enabling a true end to end service orchestration across virtual and physical network domains

2 Architectural Progression

OSM is innately a modular and model driven architecture (Figure 1) and has evolved steadily to adopt cloud native design principals. The ease with which the community has adapted the modularity to better fit the growing functionality is an important characteristic to how the OSM community continues to operate efficiently. During Release FOUR the community rationalised the Network To VNF Configuration (N2VC) module and the VNF Configuration and Abstraction Task Force to support rapid progress on the VNF configuration functionality. The Information Model and Northbound API were combined under one module ensuring model/interface harmony in the system and to support a growing demand for vendor/user differentiation options, a task force was formed to support the development of greater user interface choice.

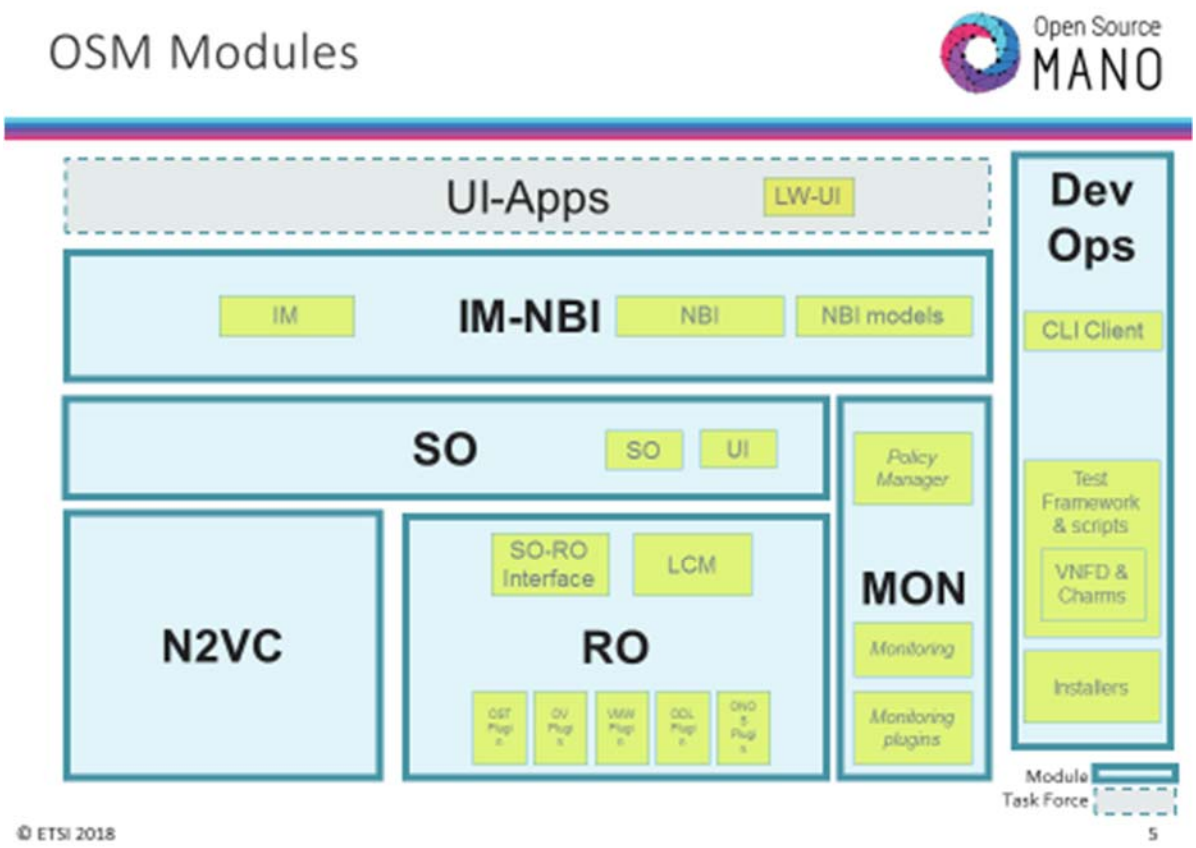

Figure 1: OSM Release FOUR Architecture Progression

The OSM community has also continued to systematically evolve both the architecture within and between modules to meet the needs of the operator community. Figure 2 depicts how OSM Release FOUR has moved to a common message bus provides for a new dedicated channel for asynchronous communication between components. This makes OSM more open and simpler to integrate with new pluggable modules and has facilitated greater centralisation of common services.

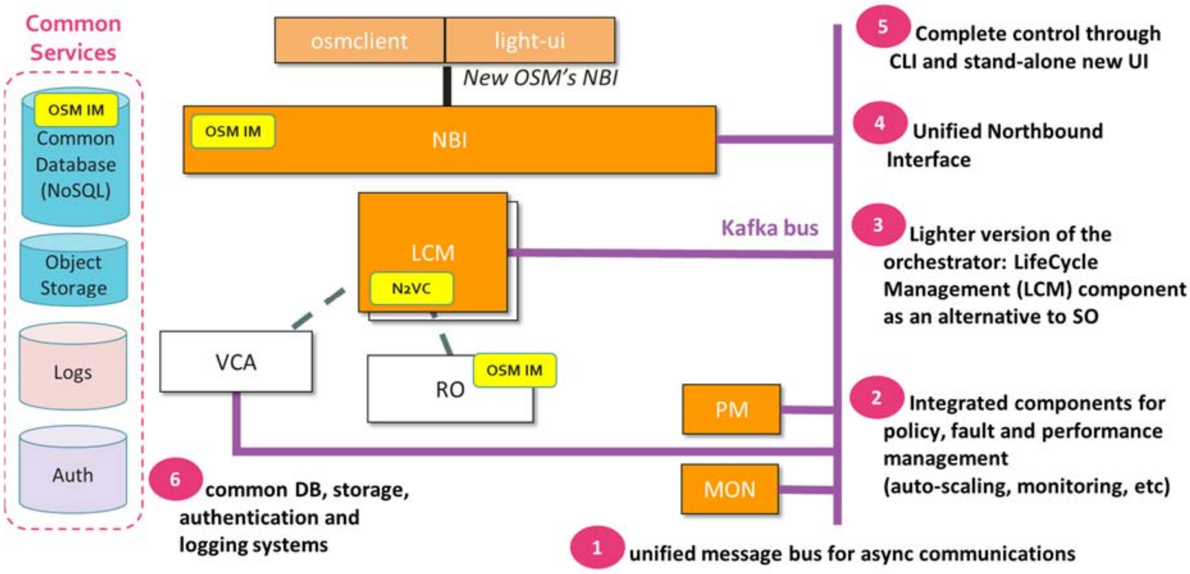

Figure 2: OSM Release FOUR Architecture Progression showing addition of new LifeCycle Management (LCM) component.

3 Release FOUR Feature Review

3.1 Collection of VIM metrics by OSM

One of the guiding principles for the OSM Monitoring Module (MON) is that it is required to interface with and leverage existing or new monitoring systems. The Monitoring Module is not intended to replicate or compete with those systems.

The Monitoring Module should mostly be considered as a tool for driving monitoring configuration updates to the external monitoring tool and as a conduit for steering actionable events into the Service Orchestrator. These actionable events may be either directly triggered by running NS/VNFs or deduced by the external monitoring tools.

One of the most powerful things OSM is delivering as a part of the Monitoring Module is the ability to correlate telemetry related to the VMs and VNFs to the relevant Network Services. Automated correlation is expected to provide a considerable user experience improvement to OSM users and drive up efficiency for operators in a Telecommunications environment.

Apache Kafka was used as the Monitoring Module message bus implementation (and now underpins OSM inter-module communications). It is a fault-tolerant message passing system that supports a publish-subscribe model that aligns with the Monitoring Module's architecture. Messages sent to, or received from the Monitoring Module core will be passed via the message bus for both internal and external components of monitoring. Apache Kafka "topics" and "partitions" are used to segregate messages to MON.

The Monitoring Module is architected to support a flexible plugin method to integrate with the monitoring tool of choice. For Release FOUR, OpenStack Aodh and OpenStack Gnocchi are supported. The community will complete the upgrade of the Amazon CloudWatch and VMware vRealize™ Operations Manager support in an early Release FOUR "point" release.. The Monitoring Module Models sub-component contains the definition of alarms and metrics that OSM can process. The monitoring tool plugin is responsible for translating (aka normalizing) alarms and metrics from the innate format of the monitoring tool into the format that OSM interprets.

Table 1 shows the normalized metrics that OSM uses and their corresponding monitoring tool metrics.

Normalized Metric Name	Unit	VMware vROPs Metric[2]	Amazon CloudWatch Metric[2]	OpenStack Metric
AVERAGE_MEMORY_UTILIZATION	%	mem|usage_average	Not Supported	Memory.total (%)

[2] Feature set to be enabled in a Release FOUR "point" update.

Normalized Metric Name	Unit	VMware vROPs Metric[2]	Amazon CloudWatch Metric[2]	OpenStack Metric
READ_LATENCY_<DISK_NO>	msec	virtualDisk|totalReadLatency_average	Not Supported	Not Supported
WRITE_LATENCY_<DISK_NO>	msec	virtualDisk|totalWriteLatency_average	Not Supported	Not Supported
DISK_READ_OPS	Nos	Not Supported	DiskReadOps	Disk_ops.DISK
DISK_WRITE_OPS	Nos	Not Supported	DiskWriteOps	Disk_ops.DISK
DISK_READ_BYTES	Bytes or bytes/sec	Not Supported	DiskReadBytes	Disk_octets.DISK
DISK_WRITE_BYTES	Bytes or Bytes/sec	Not Supported	DiskWriteBytes	Disk_octets.DISK
PACKETS_DROPPED_<NIC_NO>	Nos	net|dropped	Not Supported	if_dropped.INTERFACE
PACKETS_RECEIVED	Nos	net:Aggregate of all instances|packetsRxPerSec	NetworkPackets In	if_packets.INTERFACE
PACKETS_SENT	Nos	net:Aggregate of all instances|packetsTxPerSec	NetworkPackets Out	If_packets.INTERFACE
CPU_UTILIZATION	%	cpu|usage_average	CPUUtilization	Percent.virt_cpu_total

Table 1: OSM Normalized Metrics

Table 2 shows the normalized alarms that OSM uses and their corresponding monitoring tool metrics that are used to support triggering these alarms.

Normalized Alarm Name	VMware vROPs Metric[2]	Amazon CloudWatch Metric[2]	OpenStack Metric
Average_Memory_Usage_Above_Threshold	Average_Memory_Usage_Above_Threshold	Not Supported	Average_Memory_Usage_Above_Threshold
Read_Latency_Above_Threshold	Read_Latency_Above_Threshold	Not Supported	Not Supported
Write_Latency_Above_Threshold	Write_Latency_Above_Threshold	Not Supported	Not Supported
Disk_read_ops_above_threshold	Not Supported	Disk_read_ops_above_threshold	Disk_ops.DISK
Disk_write_ops_above_threshold	Not Supported	Disk_write_ops_above_threshold	Disk_ops.DISK
Disk_read_bytes_above_threshold	Not Supported	Disk_read_bytes_above_threshold	Disk_octets.DISK
Disk_write_bytes_above_threshold	Not Supported	Disk_write_bytes_above_threshold	Disk_octets.DISK
Net_Packets_Dropped	Net_Packets_Dropped	Not Supported	Net_Packets_Dropped
Packets_in_Above_Threshold	Packets_in_Above_Threshold	Packets_in_Above_Threshold	if_packets.INTERFACE
Packets_out_Above_Threshold	Packets_out_Above_Threshold	Packets_out_Above_Threshold	if_packets.INTERFACE
CPU_Utilization_Above_Threshold	CPU_Utilization_Above_Threshold	CPU_Utilization_Above_Threshold	CPU_Utilization_Above_Threshold

Table 2: OSM Normalized Alarms

In addition to the experimental support shown in Release THREE, there is now the ability to configure on-demand and (in an upcoming point release) descriptor-driven setting of alarms and metrics via the northbound interfaces. This makes the configuration and consumption of pre-correlated telemetry much simpler and convenient to leverage. Likewise, the support of push notifications and configuration of reactive policies, via the new Policy Manager, opens the door to more advanced service assurance via closed-loop operations.

3.2 Fault & Performance Management of NS

Once a Network Service is deployed, it is necessary to have a means to retrieve events notifying of any occurrences that may require the operator action, such as shortage of resources or severe errors. It is also necessary to have a means to retrieve the values for performance indicators that show how the system is behaving. In traditional environments, this is usually achieved by the network element exposing a set of SNMP traps, syslog, and more recently, by Netconf event streams.

The Monitoring module updates for Release FOUR has enabled a path for fault management solutions existing outside of OSM to consume alarm data that can provide the correlation between the NFVI view of faults in the environment with the logical network services view. This in turn facilitates a greater level of autonomy in the overall monitoring solution and enables a path for efficient and effective closed-loop automation. In addition, the optional capabilities that are now enabled with OSM Release FOUR includes the ability to visualise the performance metrics via Grafanna.

3.3 Support of multi-VDU VNFs

Many common use cases involve VNFs that are composed of several types of VDUs that need to be scaled and operated independently but still belong to the same VNF and, hence, need to be completely described in the VNF Package, not at the NS level.

Some common examples are composite VNFs such as IMS or EPC. Simple but decomposed VNFs, such as a vRouter with control and data plane that are intended to run in separate VMs.

Note: VCA relations support is expected to land in a Release FOUR point release.

This functionality now enables a VNF package with a VNF composed of more than one type of VDUs (e.g. 2 types of VDUs: user and data plane) to be successfully on-boarded in the system. The set of primitives available is unified at VNF level. The VNF is properly deployed, with the VMs interconnected as specified in the descriptor and the VNF as a whole can be operated with VNF level primitives.

3.4 VNF SW upgrade

Once a Network Service is running in a production environment it is likely that some VNFs may require a SW upgrade. Via support for Day 2 operations related to actions, OSM has a method to invoke the SW upgrade operation with minimal impact in the running NS.

3.5 Clean-up of OSM's northbound API

OSM Release FOUR delivers on the culmination of multiple activities over multiple releases to harmonise and clean up the northbound REST API. One of the driving factors was to respect the layering architectural principal and by doing so to present the users with a consistent interface experience.

The new northbound interface is aligned with the ETSI NFV specification SOL005. It provides a single pane of glass to control the OSM system. To facilitate the interoperability with existing and new client applications, the details of the interface are entirely available in OpenAPI format

3.6 Allow alternative images for specific VIMs in the descriptor

In a hybrid multi-VIM deployment with private and public cloud sites, it is advisable to allow a different strategy of image management for both types of VIMs. One requirement is that it must not be mandatory to upload images to the public cloud. Reasons for this position include:

1. Public cloud often has a set of pre-loaded images (e.g. AWS's AMIs) that could facilitate their immediate consumption to users.
2. Uploading a big image to public cloud might lead to relevant charges from the cloud provider.

Those public cloud images often have their own identifiers, checksums, etc., different from the ones for an image in public cloud.

Release FOUR has extended the IM for the VNFD so that it is now possible to support:

1. The current image reference is interpreted as the default image reference for all VIMs.
2. Allow, as option, to include references to alternative images for specific types of VIMs that would be used for sites of that VIM type instead of the default one.

3.7 Catalogue Search Functions

Invoking service primitives on a Network Service or getting monitoring parameters requires to provide the Network Service Record ID (nsr-id) or the VNF Record ID (vnfrid), which are returned after instantiation. With Release THREE it was if the system that invokes the primitives later is not the one that launched the instantiation, the nsr-id/vnfr-id was not available.

Release FOUR brings the new northbound API and in the implementation is backed by a common database that is used to store the state of the system. This common database makes searching the system a much more convenient operation and also enables really quick extensions to search capabilities.

3.8 Lightweight build of OSM

The Release THREE installation of OSM required an amount of resources (4 CPUs, 8 GB RAM, 40GB disk) that, although reasonable for production environments, are highly inconvenient for more reduced setups, casual testers or demo environments. This feature delivers on a new build of OSM that:

- Allows to run all operations, preserving OSM's IM and NBI, so VNF/NS packages and OSM_Client are functional.
- Has a minimal footprint (2CPUs, 4 GB RAM).

3.9 Selective support of MAC address selection

Many legacy, non-cloud-native VNFs assume that the VIM can provide specific MAC addresses to specific interfaces of their VDUs/VMs. This feature provides support to model the same behaviour that some of the most popular VIMs already provide for some interface types (paravirtualized interfaces and, optionally, SR-IOV).

As in the case of IP profiles, a coherency check prevent that these MAC addresses created by the VIM have conflicts between different instances of the same NS/VNF or between VNFs interconnected in a NS.

In order to guarantee that coherency, MAC addresses can only be set in a descriptor where they correspond to VLs or CPs that are considered “internal” to the corresponding VNFD or NSD. The implication is that at a VNFD level, a specific MAC address can be set only in internal interfaces/CP.

4 Experimental Features

4.1 VNF Forwarding Graph Descriptor

A service provider needs the ability to offer a service to corporations such as one that consists of a VPN service with access to different types of endpoints (other corporate sites, Internet or public cloud) where each corporation can insert a custom chain of VNFs in the middle. Selected traffic from corporate clients can reach an NFV datacenter. Once in the datacenter, the corporation can select which chains to be used for each traffic flow (i.e. classified) depending on attributes such as the source and destination of the traffic.

The VNF Forwarding Graph Descriptor (VNFFGD) is a well-defined data model from ETSI NFV (http://www.etsi.org/deliver/etsi_gs/NFV-IFA/001_099/014/02.01.01_60/gs_NFV-IFA014v020101p.pdf) which was subsequently adopted by OSM as part of the continuous progression towards ETSI NFV phase 2 models.

Release FOUR introduces experimental support for VNFFGD running in the OpenStack environment, specifically in relation to OpenStack installations that support the networking-sfc capability [Ref 2].

4.2 OSM Platform Resiliency to Single Component Failure

The NFV Orchestrator is a critical component for the operator in a production environment. As such, it must be capable of recovering from unexpected failures of its components, via a combination of techniques. In this case, it should be possible to keep the system alive in case of failure of a single component (e.g. active-standby redundancy).

The move to a container based deployment model has opened the door to innately leveraging some of the resiliency functionality container based models provide.

The community considers that Release FOUR represents another good improvement in terms of resiliency, but a greater level of system testing is required before this feature will be promoted from its experimental status.

4.3 Allow to specify management IP addresses as parameters at instantiation time

Having a management interface ready and reachable right after boot is critical for driving Day 1 and Day 2 operations. With Release THREE OSM assumed that the VIM assigned an IP address to the VNF from the pool in the management network (the VNF can retrieve it via DHCP or metadata service, if available).

However, many legacy OSS assume that the VNFs should have a specific IP address assigned from a pool of addresses that they manage (or that they can learn from other systems). To support this use case, OSM now (experimentally) provides a means to be told, at the time of instantiation, which management IP address should be used. Please note that this address cannot be part of the VNFD or the NSD or, otherwise, several instances of the VNF/NS would have the same IP address, leading to collisions, etc.

4.4 Light UI Features

The Light UI is a new GUI demonstrating use of OSM's SOL005-based NBI API. This is related to the new lightweight build of OSM introduced in section 3.8 above.

5 Preview: Features Planned for a Release FOUR "Point" Update

OSM release regular minor (aka "point") updates in parallel with the development cycle of the next major release. The community are expecting to continue to rapidly progress functionality related to the following:

1. Monitoring of VNF metrics.
2. Completing the upgrade of the VMware vRealize and AWS CloudWatch plugins for MON.
3. Keystone integration is being planned to extend RBAC functionality as well as open up the possibility for support of different back ends for user authentication such as LDAP, Kerberos, etc.
4. Support of physical deployment units in a VNF.
5. Extend scaling support from VNF level only scaling to include VDU scaling.
6. Full charm support with native charms.
7. Full support of IP profiles while descriptor coherency is preserved

6 Release FOUR DevOps Update

Devops has introduced significant improvements to OSM artifact release management utilizing JFrog Artifactory Pro [Ref 3].

Recall the existing OSM CI/CD utilizes a 4 stage pipeline (Figure 3:

1. Stage 1 focuses on Gerrit which supports community collaboration such as code reviews.
2. Stage 2 includes a per-module pipeline where the testing work related to a specific module resides. All of the OSM modules now support packaging in Docker containers, and this stage operates tests within Docker containers and allows for parallel execution of Jenkins pipelines.
3. Stage 3 focuses on system installation and smoke testing.
4. Stage 4 incorporates end-to-end system tests leveraging real NFV infrastructure, VIMs, SDN controllers and VNFs.

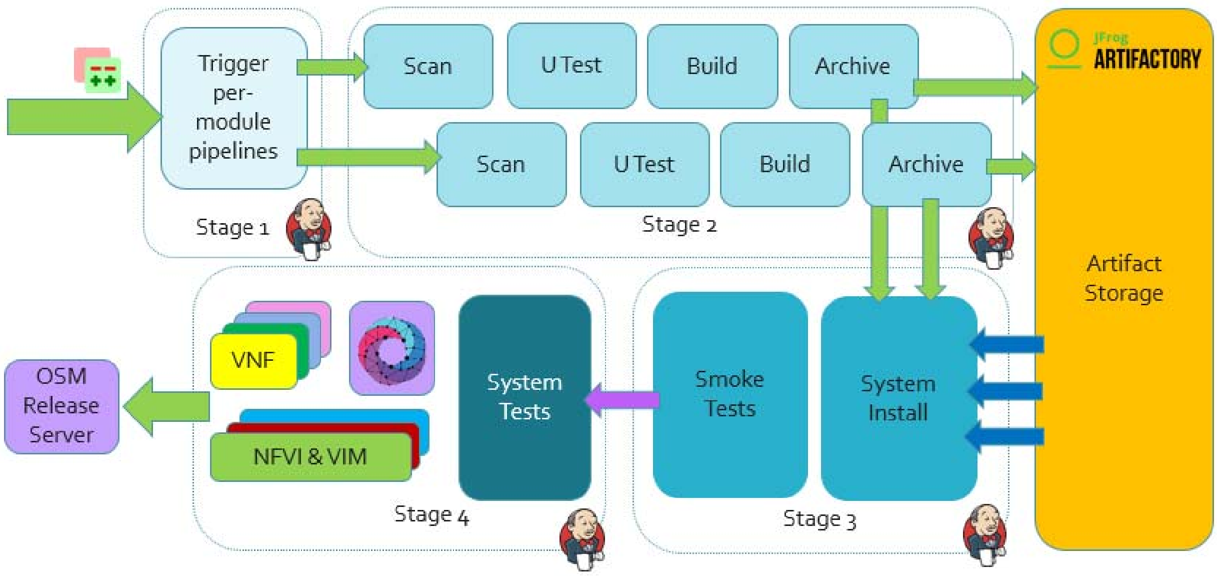

Figure 3: OSM Release FOUR CI/CD Pipeline.

During the stage 2 artifact creation, the builds are stored in the Artifactory repository. Previously in Release THREE, debian package artifacts were utilized. In Release FOUR, OSM now leverages a micro-services architecture where the services are run inside docker containers. Since Artifactory has support for docker image artifacts, the transition to docker was seamless. On success of stage 2, the artifacts are promoted to the stage 3 Artifactory repository.

Stage 3 installs the complete OSM system utilizing the promoted artifacts. A simple type of test called smoke is run that validates the operation of OSM without external VIM interactions. On completion of stage 3, the artifacts are promoted to the stage 4 repository.

Stage 4 re-reruns the OSM via the stage 4 repository while connecting to external VIMs available on the HIVE(*) based OSM Remote Labs network. End-to-end system tests are invoked, and on success, all OSM artifacts are promoted to the release staging repository.

As new artifacts become available in the release staging repository, a Jenkins job is initiated to query the repository and promote all images to the public OSM release site. As OSM utilizes a repository for each MDG component, a common script implements this promotion that leverages Artifactory's open feature-rich REST API.

The net result of optimisations to the CI/CD pipeline is that the developer experience is significantly improved with faster turnaround times on code submissions, and there is now an even more efficient and stable release process that enables the community to release frequent minor releases in parallel with major release development cycles.

(*) HIVE: Hub for Interoperability and Validation at ETSI

7 References

1. OSM Release FOUR Wiki: https://osm.etsi.org/wikipub/index.php/OSM_Release_FOUR
2. VNF Forwarding Graph Wiki: https://osm.etsi.org/wikipub/index.php/OSM_RO_VNFFG_implementation
3. JFrog Artifactory - https://www.jfrog.com/artifactory/

ETSI (European Telecommunications Standards Institute)
06921 Sophia Antipolis CEDEX, France
Tel +33 4 92 94 42 00
info@etsi.org
www.etsi.org