ETSI Plugtests Report V1.0.0 (2020-08)

ETSI NFV&MEC Plugtests
Remote
15-19 June 2020

Keywords

Testing, Interoperability, NFV, MANO, VNF, VIM

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-préfecture de Grasse (06) N° 7803/88

Contents

Executive summary

The NFV&MEC Plugtests 2020 was organised by the ETSI Centre for Testing and Interoperability as part of the NFV Plugtests Programme. The event, originally planned to take place at ETSI 15-19 June 2020, was finally held remotely due to the COVID-19 pandemics travel and hosting restrictions.

The NFV&MEC Plugtests offered NFV and MEC solution providers and open source projects an opportunity to meet on-line and assess the level of interoperability of their NFV and MEC solutions, while validating their implementation of NFV and MEC specifications and APIs.

42 organisations and over 170 engineers were involved in the preparation of this busy event forming an engaged and diverse community of implementers testing together over 65 NFV and MEC solutions, such as:

- Virtual and Containerized Network Functions (VNFs & CNFs), combined in different Network Services (NS)
- Management and Orchestration solutions: NFV Orchestrators and VNF Managers
- NFV Platforms: NFV Infrastructure and Virtual and Infrastructure Managers, posiibly offering Container Infrastructure Services
- MEC Platforms and Applications
- OSS/BSS solutions
- Simulators, test and automation tools implementing or validating NFV and/or MEC APIs.

Figure 1. NFV&MEC Plugtests 2020 Participation

Different participating organisation and Function Under Test (FUTs) were able to interact remotely through the NFV HIVE (Hub for Interoperability and Validation at ETSI) which provides a secure framework to interconnect participants' labs and implementations and is a key element for an efficient Plugtests preparation and successful events.

The main highlights of this NFV&MEC Plugtests were:

- The **experimental cloud-native tests track**, which allowed for the first-time to test interoperability among 5 CNFs, 7 MANO solutions and 7 Containers Infrastructure Services (Kubernetes platforms) from different providers. This testing, which was based on NFV Release 4 specifications currently under development, showed an interoperability rate of 93% and triggered interesting discussions about how the NFV specs will evolve to allow for full automation of CNF onboarding and orchestration. A summary of the discussed point can be found in clause 10.4, and detailed interoperability results are available in clause 9.1.2

- The new **Robot Framework** Test Suite, developed by the Plugtests team to enable **Automated NFV IOP Testing** based on NFV SOL016 specifications, currently under development. The test suite reuses existing NFV API Conformance Robot Test Suites and leverages existing NFV APIs to trigger operations and check finals results in multi-vendor configurations. The interoperability rate for the automated test sessions was above 75% and the experimental test suite will be contributed to ETSI NFV.

- The **growing number of commercial VNFs** are joining the NFV Plugtests Programme, including this year, for the first time some fully functional 5G Cores, which were able to achieve not only NFV interoperability test sessions but also functional testing of **5G use cases**. The list or participating VNFs and other FUTs is available in clause 5 and the details of the 5G testing compiled in Clause 10.4.

- The **extended support of key open source communities** in the NFV and cloud-native ecosystem, and active participation of seven distributions of Kubernetes and Openstack, three of ETSI OSM – Open Source MANO, and one of ONAP, Openslice and StarlingX.

- The **wider interest on MEC technologies** including ETSI specifications and MEC APIs. The **new test plan developed for the event** offered the exploitation of MEC APIs in interoperability testing, with encouraging success rates. **New participants joining the MEC Interoperability testing**, and a notably wider participation of MEC App providers, allowed testing **multi-apps scenarios** and **full MEC Services exploitation workflows**. Also, the newly introduced **MEC API Conformance** track received high interest and permitted the validation of the ETSI MEC API Conformance test suites.

In addition of the above, the Plugtests allowed once more to validate the latest versions of the NFV and MEC Specifications, APIs and Test Suites as well as to identify and fix a number of issues, bugs and inconsistencies, which are compiled in section 10 and will help to increase the quality of future releases.

The following sections describe in detail the preparation of the NFV&MEC Plugtests, the participating implementations, the test plans and testing procedures, the overall results, as well as the lessons learnt, and the feedback collected during the event.

This Plugtests Report is fed back to ETSI NFV and MEC Industry Specification Groups.

1 Introduction

The NFV&MEC Plugtests 2020 aimed at verifying interoperability and API Conformance across different implementations of the main components of the NFV and MEC Architectural Frameworks, including:

- Virtual Network Functions (VNF) and Containerized Network Functions (CNF), combined in different Network Services (NS)
- Management and Orchestration (MANO) solutions, providing integrated or standalone NFV Orchestrators (NFVO) and VNF Managers (VNFM)
- NFV Platforms, including hardware, providing NFV infrastructure (NFVI) and Virtual Infrastructure Manager (VIM) eventually providing Container Infrastructure Services.
- OSS/BSS Solutions
- MEC Apps and MEC Platforms

In addition, test and support functions could be used to simulate certain components, validate the correct behaviour of the Network Services (NS) and/or automate part of the Test Plans.

Remote integration and pre-testing among participants were key for the preparation and a successful Plugtests. For that purpose, the NFV Plugtests HIVE (Hub for Interoperability and Validation at ETSI - a dedicated VPN based network) was used to interconnect local and remote implementations in a reliable and secure way.

All the participating implementations, Functions Under Test or test/support Functions were connected and/or accessible through the HIVE network: most of the NFV, MEC platforms and MANO solutions run remotely on participants' labs. VNF, CNF, MEC Apps and NS Packages and images were made available in a local Repository and Docker Registry hosted at ETSI and uploaded to the different NFV / MEC platforms during the pre-testing phase. Some test and support VNFs were deployed locally on local infrastructure at ETSI, also accessible through HIVE. All the participants had access to the HIVE network, either over their company's site-to-site VPN or through a personal client-to-site VPN.

As in previous events, all the information required to organise, coordinate and manage the NFV&MEC Plugtests 2020 was compiled and shared with participants in the private NFV Plugtests Programme WIKI. Part of the information presented in this document has been extracted from there: https://wiki.plugtests.net/NFV-PLUGTESTS (login required).

.

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long-term validity.

[NFV002] ETSI GS NFV 002: "Network Functions Virtualisation (NFV); Architectural Framework".

[NFV003] ETSI GS NFV 003: "Network Functions Virtualisation (NFV); Terminology for main concepts in NFV".

[TST002] ETSI GS NFV-TST 002: “Network Functions Virtualisation (NFV); Testing Methodology; Report on NFV Interoperability Testing Methodology”

[TST007] ETSI GR NFV-TST 007: “Network Functions Virtualisation (NFV); Testing; Guidelines on Interoperability Testing for MANO”

[TST010] ETSI GR NFV-TST 010: “Network Functions Virtualisation (NFV); API Conformance Testing Specification”

[SOL002] ETSI GS NFV-SOL 002 V2.4.1: “Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; RESTful protocols specification for the Ve-Vnfm Reference Point”

[SOL003] ETSI GS NFV-SOL 003 V2.4.1: “Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; RESTful protocols specification for the Or-Vnfm Reference Point”

[SOL005] ETSI GS NFV-SOL 005 V2.4.1: “Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; RESTful protocols specification for the Os-Ma-nfvo Reference Point”

[SOL016] ETSI GS NFV-SOL 016 V0.5.0 (draft): “Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; NFV-MANO Procedures”

[MEC003] ETSI GS MEC 003 V2.1.1: “Multi-access Edge Computing (MEC); Framework and Reference Architecture”

[MEC009] ETSI GS MEC 009 V1.1.1: Mobile Edge Computing (MEC); General principles for Mobile Edge Service APIs

[MEC011] ETSI GS MEC 011 V2.1.1: “Mobile Edge Computing (MEC); Mobile Edge Platform Application Enablement”

[MEC-012] ETSI GS MEC 012 V2.1.1: “Multi-access Edge Computing (MEC); Radio Network Information API”

[MEC-013] ETSI GS MEC 013 V2.1.1: “Multi-access Edge Computing (MEC); Location API”

[MEC017] ETSI GR MEC 017 V1.1.1: “Mobile Edge Computing (MEC); Deployment of Mobile Edge Computing in an NFV environment”

[MEC025] ETSI GS MEC-DEC 025 V2.1.1: “Multi-access Edge Computing (MEC); MEC Testing Framework”

[DEC032] ETSI GS MEC-DEC032: "Multi-access Edge Computing (MEC); API Conformance Test Specification"

[MEC-IOP-TP] NFV&MEC Plugtests 2020 - Test Plan for MEC Interoperability: https://portal.etsi.org/Portals/0/TBpages/CTI/Docs/ETSI_NFV_MEC_Plugtests_2020_-_MEC_IOP_Test_Plan_v1_0_0.pdf

[4NFVPLU-R] 4th ETSI NFV Plugtests Report: https://portal.etsi.org/Portals/0/TBpages/CTI/Docs/4th_ETSI_NFV_Plugtests_Report_v1.0.0.pdf

[FORGE] ETSI Forge https://forge.etsi.org

[NFV-ROBOT-TS] Robot Test Suite for NFV API Conformance https://forge.etsi.org/gitlab/nfv/api-tests

[MEC-ROBOT-TS] Robot Test Suite for MEC API Conformance https://forge.etsi.org/rep/mec/gs032p3-robot-test-suite

[MEC-TTCN3-TS] TTCN-3 Test Suite for MEC API Conformance https://forge.etsi.org/rep/mec/gs032p3-ttcn-test-suite

[NFV-ISSUE-TR] Issue Tracker for the Robot Test Suite for NFV API Conformance https://forge.etsi.org/gitlab/nfv/api-tests/issues

[NFV-AUTO-IOP] Robot Test Suite for NFV Automated Inteoperability https://forge.etsi.org/rep/plugtests/nfv/automated-interop

[MEC-ROBOT-ISSUE] Issue Tracker for the Robot Test Suite for MEC API Conformance https://forge.etsi.org/rep/mec/gs032p3-robot-test-suite/issues

[MEC-TTCN3-ISSUE] Issue Tracker for the TTCN-3 Test Suite for MEC API Conformance https://forge.etsi.org/rep/mec/gs032p3-ttcn-test-suite/issues

[NFV-FEAT17-WIKI] Wiki page of NFV FEAT17 Cloud Native VNFs and Container Infrastructure Management https://nfvwiki.etsi.org/index.php?title=Feature_Tracking#FEAT17:_Cloud-Native_VNFs_and_Container_Infrastructure_management

[NFV-IOP-TR] NFV Interoperability Test Results per Test Case https://nfvwiki.etsi.org/images/NFV_Interoperability_Results_per_Test_Case.pdf

[NFV-API-TR] NFV API Conformance Test Results per Test Case https://nfvwiki.etsi.org/images/NFV_API_Conformance_Results_per_Test_Case.pdf

3 Abbreviations

For the purposes of the present document, the terms and definitions given in [NFV003] and [TST002] apply.

4 Technical and Project Management

4.1 Scope

The NFV&MEC Plugtests focused on testing several areas

- Multi-vendor NFV Interoperability
    - Including Cloud-Native configurations
    - Including OSS/BSS solutions
    - Including Automated IOP Testing
- NFV API Conformance
- Multi-vendor MEC Interoperability
- MEC API Conformance

4.1.1 NFV Interoperability

The main goal of the multi-vendor NFV interoperability test sessions were to validate ETSI NFV end-to-end capabilities such as onboarding, instantiation, manual and automatic scaling, updates, fault and performance management and termination.

During these sessions, the Systems Under Test (SUTs) were made of different combinations of the following Functions Under Test (FUTs):

- One or several NFV Platforms, including hardware and providing pre-integrated VIM and NFVI functionality, and eventually Containerized Infrastructure Services
- One MANO solution, providing pre-integrated or standalone NFVO and VNFM functionality
- One Network Service, composed of VNFs/CNFs from one or several providers,
- Optionally, an OSS / BSS solution or a Test System

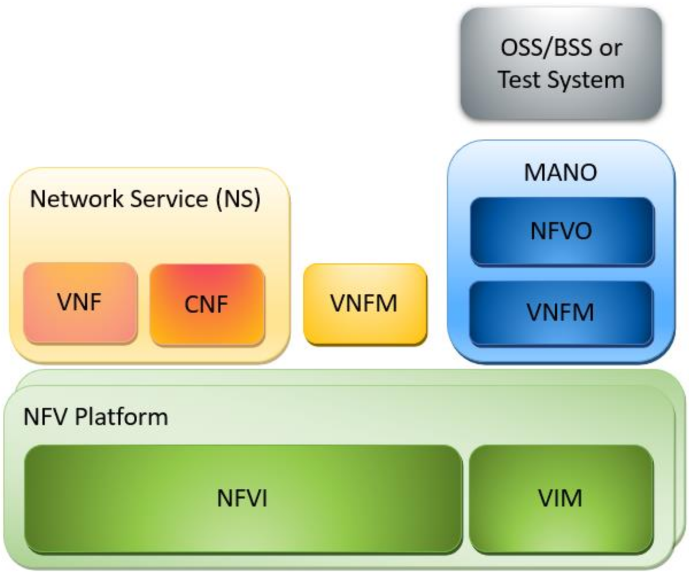

Figure 2. NFV Interoperability System Under Test

4.1.2 NFV API Conformance

The main goal of the NFV API Conformance test sessions was to run individual Test Sessions between participating Functions Under Test and a Test Automation Platform (TAP) provided by the Plugtests Team. These sessions allowed to:

- validate the Robot Test Suites (run by the Test Automation Platform) for the NFV API Conformance Test Specification: NFV TST010
- asses the level of conformance of participating Functions Under Test of VNFs, VNFMs, and NFVOs (operated by participants) with NFV SOL002, SOL003, and SOL005 APIs and OpenAPIs (v.2.41. and v2.6.1)

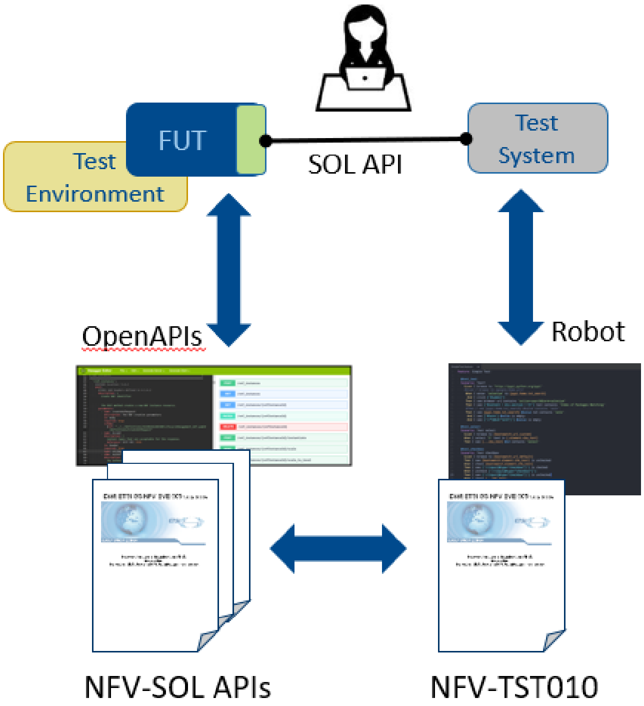

Figure 3. NFV API Conformance testing

4.1.3 MEC Interoperability

The MEC Interoperability testing was featured for the first time as a standalone track dedicated to Multi-Access Edge Computing in the NFV Plugtests Programme. The main goal of these sessions was to test interoperability of MEC Applications execution on different MEC Platform and in different deployment types (i.e. MEC standalone and MEC in NFV). The MEC Interoperability Track proposed 3 groups of interoperability tests covering application lifecycle management, traffic and DNS management, and MEC Service management. The main interoperable interfaces where between MEC Applications and MEC Platforms (the Mp1 reference point). The test configurations were derived from the generic Interoperability testing architecture reported in [MEC025].

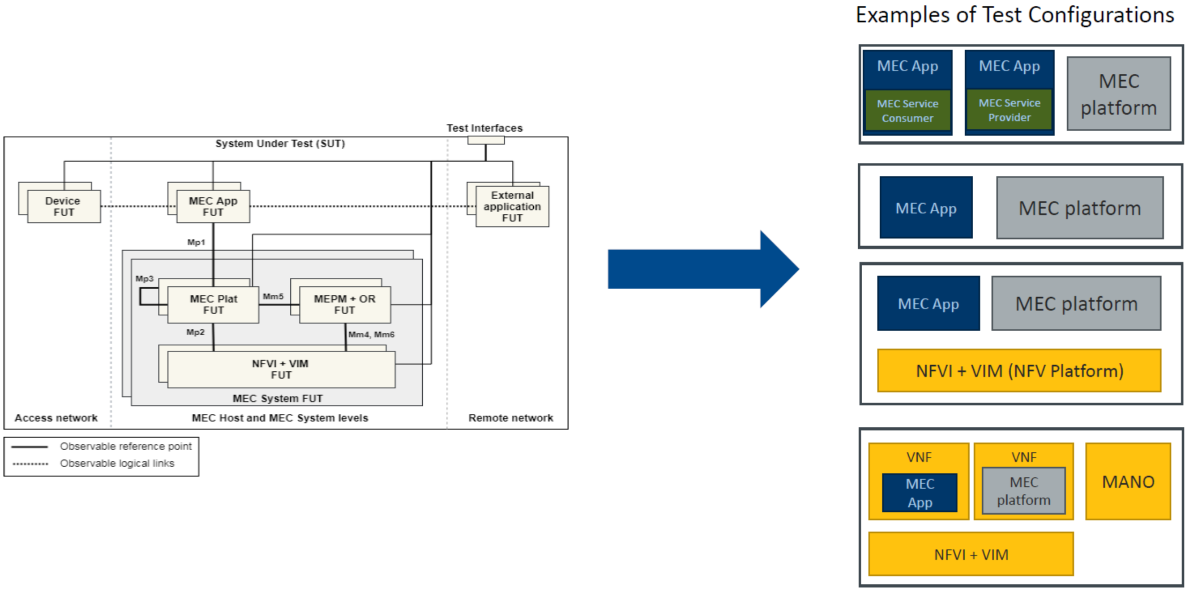

Figure 4. MEC Interoperability Testing

4.1.4 MEC API Conformance

The MEC Conformance Test sessions focussed on API Conformance Testing for MEC APIs. The test plan was based on the ETSI GS MEC-DEC032 [DEC032] specifications, and targeted MEC Platforms as API providers for MEC management APIs and MEC service APIs.

MEC API server providers were invited to test their API implementations. For this activity two test suites were made available, one implemented in TTCN-3 language (and executed via the Titan TTCN-3 test system), one implemented in the Robot Framework language (executed via the Python executor of Robot Framework).

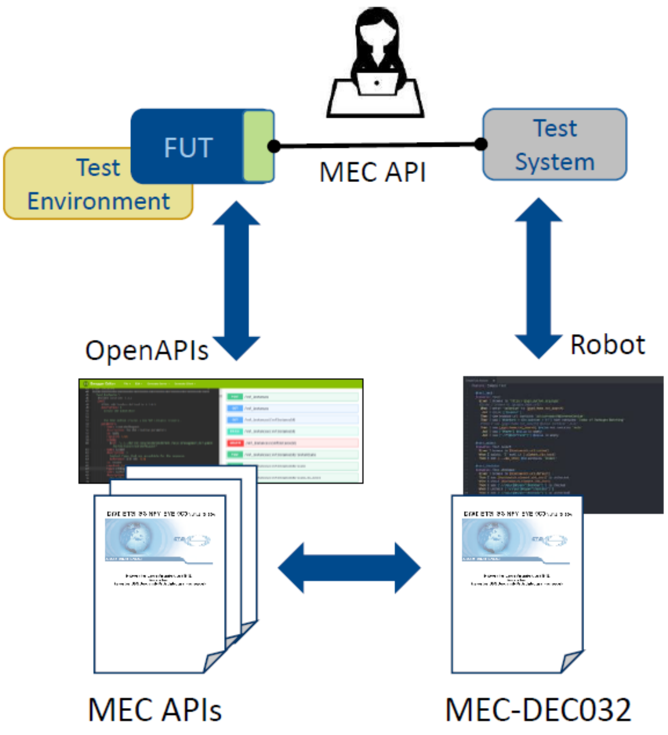

Figure 5. MEC API Conformance testing overview

4.2 Timeline

The NFV&MEC Plugtests preparation run through different phases as described in the figure below. The phases where updated when the decision of running the Plugtests as a fully remote event was taken, to accommodate the API Test Sessions before the Plugtests week (instead of during it, as originally planned for the face to face event).

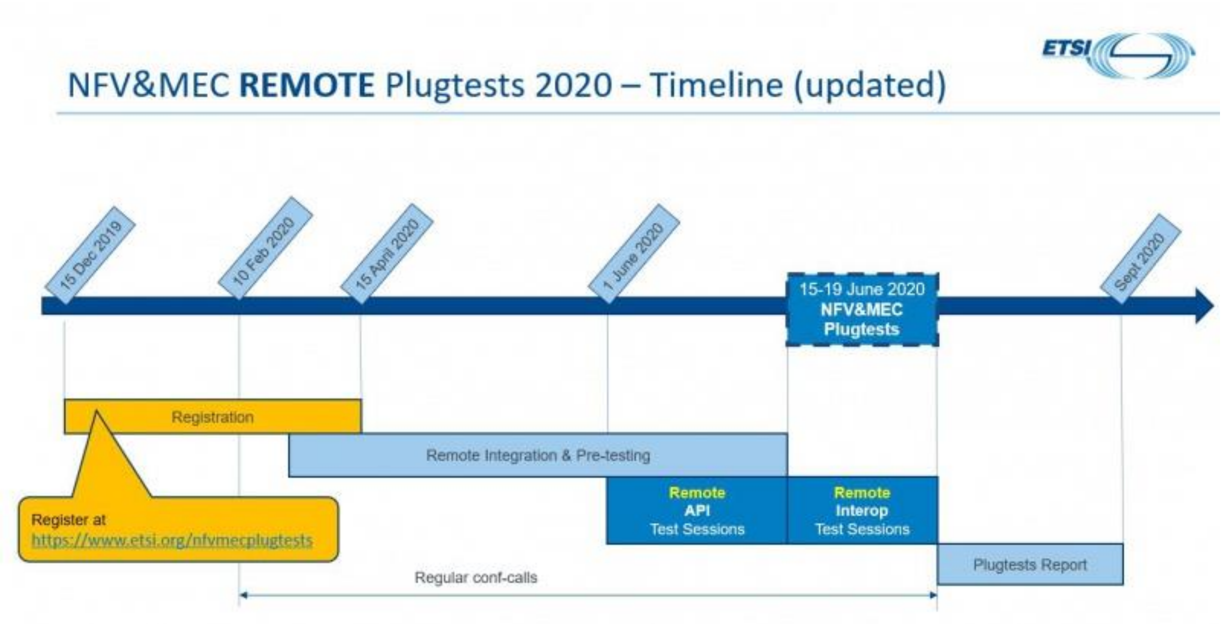

Figure 6. NFV&MEC Plugtests 2020 timeline

Registration to Plugtests was open from December 2019 to mid-April 2020 to any organisation willing to participate with a Function Under Test, or test tool. A number of observers (network operators, research, and academia) and supporting Open Source communities participated to the test plan development and review. Overall, over 170 people were involved in the Plugtests preparation and test sessions.

The following sections describe the different phases of the Plugtests preparation. During the preparation phases, weekly conf-calls were run among organisers and participants to discuss and track the remote-integration progress, anticipate and solve technical issues, review and discuss the test plans, and prepare for the different types of test sessions.

4.2.1 Remote Integration & Pre-Testing

During the Remote Integration phase, the following activities were run in parallel:

1) FUT Documentation

Participants documented their Functions Under Test (FUTs), by filling in or updating one form compiling the simplified Interoperability Features Statements (IFS) for each of their implementations. The simplified IFS Templates for each type of NFV FUT used for this Plugtests can be found in Annex A. The IFS Template for the MEC FUTs can be found in the MEC IOP Test Plan [MEC-IOP-TP]

All participants provided general details about their FUTs (description, diagrams, links to documentation.), as well as about the team(s) operating the FUT and representing the company during the preparation phase and the Test Sessions (email, Slack Id, time-zone…).

Participants providing VNFs complemented their documentation with diagrams and resource requirements.

Participants providing NFV Platforms created and documented projects and credentials for each participating MANO solution, and exposed and documented the North Bound Interfaces (NBI) of their VIM or K8s Platform.

All the information described above was made available in the Plugtests WIKI, so that it could be easily maintained and consulted by participants.

2) Test Plan Development

While this Plugtests event relayed in several existing Test Specifications, guides and test suites published by ETSI NFV and MEC, such as NFV TST007 for NFV Interoperability, NFV TST010 for NFV Conformance and, MEC DEC032 for MEC Conformance, a new Test Plan was developed for the MEC Interoperability Test Sessions. This new Test Plan [MEC-IOP-TP] builds on the experience and learnings of the experimental MEC track run during the 4th NFV Plugtests in 2019 [4NFVPLU-R] and was discussed with participants during the preparation of the Plugtests. It will be provided as input to the ETSI MEC ISG.

3) Connection to HIVE

The interconnection of different FUTs involved in the testing relies on HIVE: Hub for Interoperability and Validation at ETSI. NFV Plugtests Programme participants connect their labs to HIVE when they register for an event, so that they can interact with other participants FUTs to configure complex multi-vendor Systems Under Test.

Participants are invited to maintain their connection to HIVE in between events, and use it to run additional testing, PoCs, demos and showcases. New participants that joined this Plugtests for the first time, were invited and guided by the Plugtests team to get their implementations available on HIVE.

At the end of this phase, over 20 remote sites were connected to HIVE and each of them was allocated a dedicated network for each FUT they hosted. The interconnection of remote labs allowed to run integration and pre-testing tasks remotely among any combination of participating FUTs and helped to ensure an efficient use of the scarce time during the Plugtests week and a smoother run of the remote test sessions.

A site-to-site connection to HIVE was mandatory for participants providing NFV platforms and MANO solutions, and highly recommended for participants providing VNF and VIM software. The latter could also rely on client-to-site connection to HIVE, as long as they had no software (i.e. support function) running locally in their labs and only required access to remote labs for trouble shooting and infrastructure access purposes

Additional details on the remote test infrastructure are provided in Clause 6.

Once the above steps were completed, FUTs could start cross-FUT remote integration, see 7.1 for details on the procedures.

Once remote integration was completed, participants had the opportunity to run remote pre-testing among different combinations of VNF, MANO and NFV Platforms as well as MEC Apps and MEC Platforms.

Additional details on the pre-testing plan and procedures are provided in Clause 7.

4.2.3 Remote API Conformance Test Sessions

From 1 to 15 June, during the two weeks that preceded the Plugtests week, participants were invited to run API Conformance Test sessions to assess the level of conformance of their implementations with NFV SOL and MEC APIs.

Upon request of the participants, availability of the API Test Platform was extended for further two weeks after the Plugtests.

4.2.4 Remote Interoperability Test Sessions

From 15-19 June, participants were invited to run collaboratively multi-vendor Interoperability test sessions, according to a test schedule provided by the Plugtests team. The on-site Plugtests week was organised as follows:

Time (CET)	Monday	Tuesday	Wednesday	Thursday	Friday
8:00 - 10:00	Test Sessions				
10:00-11:00	BREAK				
11:00 - 13:00	Test Sessions				
13:00 - 14:00	BREAK				
14:00 - 16:00	Test Sessions				
16:00 - 17:00	DAILY SYNC				
17:00 -19:00	Test Sessions				

Figure 7. Plugtests week plan

The test sessions were run in parallel, so that all the participants were busy all the time. At any moment in time, there were up to 11 active test sessions: 10 on NFV IOP and 1 on MEC IOP. Participants in remote time-zones were invited to schedule additional test session outside the above plan.

4.3 Tools

4.3.1 Plugtests Wiki

The NFV Plugtests Wiki is the main entry point for all the information concerning NFV&MEC Plugtests, from logistics aspects to testing procedures. Access to this Wiki is restricted to companies participating to the NFV Plugtests Programme.

4.3.2 Test Session Scheduler

The Test Session Scheduler allowed the Plugtests organisers to produce a daily schedule during the Plugtests Week. This tool has the following objectives:

- maximise the number of test sessions
- balance the amount of test sessions among participants
- consider supported features of the participating FUTs
- consider participants' time-zones and availability
- minimise the number of participants not involved in a test session anytime.

The picture below shows a partial view of a daily schedule. Each yellow box corresponds to a specific Test Session, which depending on the targeted configuration included different components. In the NFV tracks, sessions included a Network Service with one or more VNFs from one or different providers, one MANO solution, one or more VIM&NFVI, one or more VNFMs, and one OSS/BSS solution or simulator. In the MEC track, each session included one MEC Platform and one or several MEC Apps. For each of these sessions a Test Session Report (TSR) was filed (see next clause). In addition to the pre-scheduled test sessions, participants were invited to request, run and report results for additional test sessions.

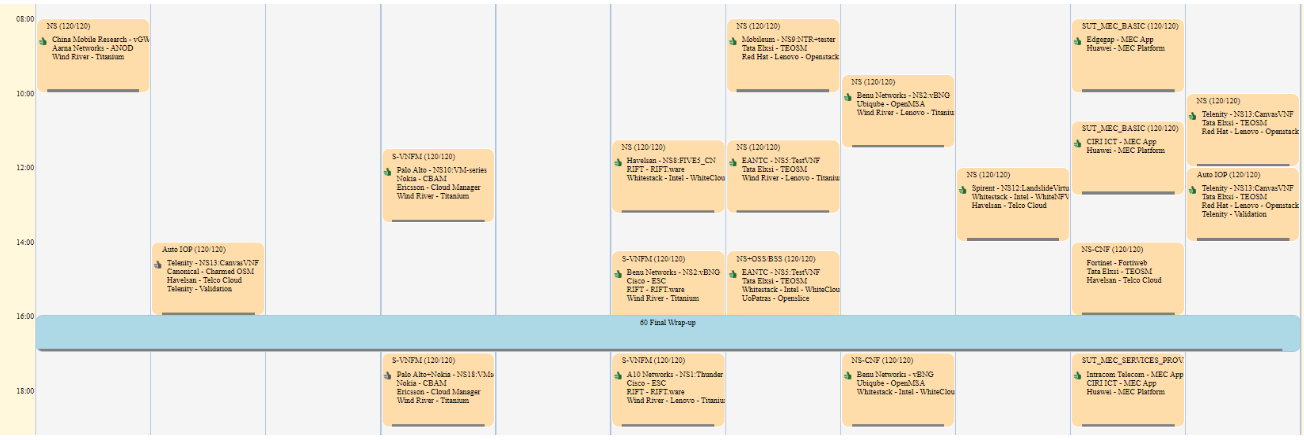

Figure 8. Daily Schedule example

4.3.3 Test Reporting Tool

The Test Reporting Tool guides participants through the Test Plan during the Test Sessions and allows them to create Test Session Reports compiling detailed results for each test case in scope. It allows reporting on pre-scheduled Test Sessions, but also on Test Sessions organised on the fly among participants to prepare, complete or complement the scheduled testing (freestyle sessions).

Only the companies providing the FUTs or Test Functions (when applicable) for each specific Test Session have access to the Test Session Reports (TSR) contents and specific results. All the companies providing the FUTs for a Test Session, i.e. VNF provider(s), MANO provider and NFV Platform provider(s) are required to verify and approve the reported results at the end of the session.

4660	2020-06-18 08:00	120	Tata Elxsi (OSM)	NS-CNF	Benu Networks - vBNG Tata Elxsi - TEOSM Havelsan - Telco Cloud
4661	2020-06-18 11:00	120	MEC	SUT_MEC_BASIC	Intracom Telecom - MEC App Huawei - MEC Platform
4662	2020-06-15 08:00	120	Luxoft	NS-CNF	Fortinet - Fortiweb Luxoft - SDL Wind River - QCT - Cloud Pltfrm (K8s)
4663	2020-06-18 13:15	120	MEC	SUT_MEC_SERVICES_CONSUMER	Intracom Telecom - MEC App Huawei - MEC Platform
4664	2020-06-19 14:15	120	Tata Elxsi (OSM)	NS+OSS/BSS	EANTC - NS5:TestVNF Tata Elxsi - TEOSM Whitestack - Intel - WhiteCloud UoPatras - Openslice
4665	2020-06-18 08:45	120	MEC	SUT_MEC_SERVICES_PROVIDER	Intracom Telecom - MEC App Huawei - MEC App Huawei - MEC Platform
4666	2020-06-18 12:00	120	Luxoft	NS	Palo Alto - NS10:VM-series Luxoft - SDL Red Hat - Lenovo - Openstack
4667	2020-06-16 17:00	120	Canonical (OSM)	NS	Spirent - NS12:LandslideVirtual Canonical - Charmed OSM Whitestack - Intel - WhiteCloud
4668	2020-06-18 11:00	120	Ubiqube	Auto IOP	EANTC - NS5:TestVNF Ubiqube - OpenMSA Wind River - Lenovo - Titanium Nextworks - SOL016 Robot
4669	2020-06-15 14:00	120	Whitestack (OSM)	NS-CNF	Mobileum - NTR Whitestack - Intel - WhiteNFV Red Hat - Lenovo - Openshift
4670	2020-06-18 14:00	120	Whitestack (OSM)	NS	Havelsan - NS8:FIVE5_CN Whitestack - Intel - WhiteNFV Wind River - Lenovo - Titanium
4671	2020-06-18 14:00	120	Tata Elxsi (OSM)	NS-CNF	Mobileum - NTR Tata Elxsi - TEOSM Red Hat - Lenovo - Openshift
4672	2020-06-18 14:00	120	Ad-hoc	Multi-Site	Ulak+Spirent - NS16:Cinar_5GC+Landslide Tata Elxsi - TEOSM Havelsan - Telco Cloud Wind River - Titanium
4673	2020-06-17 14:00	120	RIFT	NS	Ulak Haberlesme - NS14:Cinar_5GC RIFT - RIFT.ware Whitestack - Intel - WhiteCloud
4674	2020-06-17 08:00	120	RIFT	S-VNFM	EANTC - NS5:TestVNF Cisco - ESC RIFT - RIFT.ware Havelsan - Telco Cloud
4675	2020-06-16 11:00	120	Whitestack (OSM)	NS	Fortinet - NS6:Fortigate Whitestack - Intel - WhiteNFV Wind River - QCT - Titanium

Figure 9. Test Reporting Tool (extract of the TSR list)

Another interesting feature of this tool is the ability to generate real-time statistics (aggregated data) of the reported results, per test case, test group, test session or overall results. These stats are available in real time for all participants and organisers and allow tracking the progress of the testing with different levels of granularity, which is extremely useful to analyse the results.

5 Participation

5.1 Functions Under Test

The tables below summarise the different Functions Under Test provided by the Plugtests participants, and the location from where they were provided / supported or connected to the HIVE network:

5.1.1 VNFs / CNFs / MEC App

Organisation	Solution Name	VNF	CNF	MEC App	Team Location(s)	Short Description
A10 Networks	Thunder CGN/FW	Y	Y	Y	Germany	Full functional CGN and FW application
Benu Networks	Benu vBNG	Y	Y	Y	USA	Virtual Broadband Network Gateway
China Mobile	vGW	Y	N	N	Beijing, China	Open source virtual gate way
CIRI ICT - Uni. Bologna	Unibo Test MEC	N	N	Y	Bologna, Italy	Testing application for MEC infrastructure with support of the MEC 011 API.
Cisco	Cisco VPC	Y	N	N	UK	Cisco's Virtual Packet Core mobile traffic solution
Cisco	Cisco Simple VNF	Y	N	N	UK	Simple VNF, minimal resource configuration, useful for end-to-end testing
EANTC	TestVNF	Y	N	N	Germany	Test VNF
Edgegap	Arbitrium	N	N	Y	Canada	Video game server with edge computing enablers use as MEC App
Fortinet	Fortigate	Y	N	Y	Europe	Full NGFW SDWAN, IPsec, Firewall L7, DLP, Antivirus
Fortinet	Fortiweb	Y	Y	Y	Europe	WAF, AI enabled, API protections, VM or Container
Havelsan	FIVEG_CN_NRF_MANAGEMENT_VNF	Y	N	N	Istanbul, Turkey	5G Core Network Repository Function - Management Service
Havelsan	FIVEG_CN_NRF_SERVICE_DISCOVERY_VNF	Y	N	N	Istanbul, Turkey	5G Core Network Repository Function - Discovery Service
Havelsan	FIVEG_CN_NRF_ACCESS_TOKEN_VNF	Y	N	N	Istanbul, Turkey	5G Core Network Repository Function - Access Token Service
Havelsan	FIVEG_CN_AUSF_VNF	Y	N	N	Istanbul, Turkey	5G Core Access and Mobility Management Function
Havelsan	FIVEG_CN_NSSF_SELECTION_VNF	Y	N	N	Istanbul, Turkey	5G Core Network Slice Selection Function - Selection Service
Havelsan	FIVEG_CN_NSSF_AVAILABILITY_VNF	Y	N	N	Istanbul, Turkey	5G Core Network Slice Selection Function - Availability Service
Havelsan	FIVEG_CN_CACHE_VNF	Y	N	N	Istanbul, Turkey	5G Core Caches
Havelsan	FIVEG_CN_CHECKER_VNF	Y	N	N	Istanbul, Turkey	5G Core Checker
Intracom Telecom	Host IoT GW	N	N	Y	Athens	MEC App for IoT devices management in the field
Mobileum	NTR	Y	Y	N	India	Network Traffic Steering (NTR) connects to

						LTE core components like MME, HSS.
Nokia	Simple VNF	Y	N		Hungary	Sample VFN
Palo Alto Networks	VM-series	Y	N	N	Poland / Spain	NGFW with GTP-C+GTP-U awareness and correlation
Spirent	TestCenter	Y	Y	N	US/Germany	Traffic Generator, Traffic Analyzer, Capture, Control Plane emulator
Spirent	Landslide Virtual	Y	N	N	US/France	Mobile core network test and emulation tool
Telenity	Canvas SMSC	Y	N	N	Istanbul, Turkey	Short Messaging Service Center
Telenity	Canvas PROV	Y	N	N	Istanbul, Turkey	Provisioning and Screening for SMSC
Telenity	Canvas OPS	Y	N	N	Istanbul, Turkey	Operational Support and Reporitng for SMSC
Ulak Haberleşme	Cinar_AMF VNF	Y	N	N	Istanbul, Turkey	5G Core Access and Mobility Management Function
Ulak Haberleşme	Cinar_SMF VNF	Y	N	N	Istanbul, Turkey	5G Core Session Management Function
Ulak Haberleşme	Cinar_NRF VNF	Y	N	N	Istanbul, Turkey	5G Core Network Repository Function
Ulak Haberleşme	Cinar_NSSF VNF	Y	N	N	Istanbul, Turkey	5G Core Network Slice Selection Function
Ulak Haberleşme	Cinar_PCF VNF	Y	N	N	Istanbul, Turkey	5G Core Policy Control Function
Ulak Haberleşme	Cinar_AUSF VNF	Y	N	N	Istanbul, Turkey	5G Core Access and Mobility Management Function
Ulak Haberleşme	Cinar_UPF VNF	Y	N	N	Istanbul, Turkey	5G Core User Plane Function
Ulak Haberleşme	Cinar_UDR VNF	Y	N	N	Istanbul, Turkey	5G Core Unified Data Repository
Ulak Haberleşme	Cinar_UDM VNF	Y	N	N	Istanbul, Turkey	5G Core Data Management

Table 1. VNFs / CNFs / MEC Apps Under Test

5.1.2 NS

The VNFs / CNFs under test were combined in different Network Services as follows:

NS#	NS Name	VNFs
NS1	A10	Thunder CGN/FW
NS2	Benu	Benu vBNG
NS3	Cisco VPC	Cisco VPC
NS4	Cisco Simple NS	Cisco Simple VNF
NS5	EANTC	TestVNF
NS6	Fortigate	Fortigate
NS7	Fortiweb	Fortiweb
NS8	Havelsan	FIVEG_CN VNFs
NS9	Mobileum	NTR+ tester
NS10	Palo Alto Networks	VM-series
NS11	Spirent Test Center	TestCenter
NS12	Spirent Landslide	Landslide Virtual
NS13	Telenity Canvas	Canvas VNFs
NS14	Ulak 5G Core	Cinar_5GC VNFs
NS15	Ulak + Spirent	Cinar_5GC + Landslide
NS16	A10 + Spirent	Thunder + STC
NS17	Palo Alto + Nokia	VM-series+simpleVNF
NS18	A10 + Benu	Thunder+vBNG
NS19	Cisco + Palo Alto	SimpleVNF+VM-series
NS20	Cisco + Nokia	SimpleVNF+SimpleVNF

Table 2. NSs Under Test

5.1.3 MANOs

Organisations	Solution Name	NFVO	VNFM	CNFO	Location(s)	Short Description
Aarna Networks – UNH - IOL	Aarna Networks (ONAP)	Y	Y	Y	Bangalore, India	Aarna Networks ONAP Distribution 4.0 (ONAP El Alto)
Canonical	Charmed OSM	Y	Y	Y	HW (US) - SW support (EU)	Canonical's Charmed Open Source MANO 7.0.1
Cisco	NFVO and VNFM (ESC)	Y	Y	N	US	Cisco NFVO and VNFM (Elastic Services Controller)
Cisco	VNFM (ESC)	N	Y	N	UK	VNFM (Elastic Services Controller) only
Ericsson	Cloud Manager	Y	N	N	Hungary/Ireland	NFVO only
Luxoft	SDL	Y	Y	Y	Bucharest	MANO (NFVO+VNFM)
Nokia	CBAM	N	Y	N	Hungary	VNFM only (CloudBand Application Manager)
RIFT	RIFT.ware	Y	Y	Y	USA / India	RIFT NFVO and VNFM
Tata Elxsi	TEOSM (OSM)	Y	Y	Y	India	TEOSM Erai 4.0, an end to end multi domain service orchestrator
Ubiqube	OpenMSA	Y	Y	Y	Ireland/ France / India	Integrated Automation Platform
Whitestack	WhiteNFV (OSM)	Y	Y	Y	USA	Whitestack's Open Source MANO Distribution

Table 3. MANOs Under Test

5.1.4 VIM&NFVIs / MEC Platforms

Organisations	Solution Name	LAB	OS	K8s	MEC Plat	Location	Short Description
Canonical – Supermicro	Charmed Openstack	Supermicro POD	Y	N	N	HW (US) - SW support (EU)	Charmed OpenStack Stein/Train (on 18.04.04 LTS - Kernel 5.3)
Canonical – Supermicro	Charmed Kubernetes	Supermicro POD	N	Y	N	HW (US) - SW support (EU)	Charmed Kubernetes 1.17 (on 18.04.04 LTS - Kernel 5.3)
Havelsan	Havelsan Telco Cloud	Havelsan POD	Y	Y	N	Ankara / TURKEY	OpenStack STEIN + Kubernetes1.16.x
Huawei	Edge Gallery	Huawei POD	N	N	Y	China Beijing	MEC platform (MEAO+MEPM+MEP)
Red Hat - Lenovo	Red Hat Openstack	Lenovo POD ThinkSystem SR630/SR650	Y	N	N	NC, USA	RH OpenStack 16.x
Red Hat - Lenovo	Red Hat Openshift	Lenovo POD ThinkSystem SR630/SR650	N	Y	N	NC, USA	RH Openshift 4.3
Whitestack - Intel	WhiteCloud	Intel/Whitestack POD @OSM Remote Lab	Y	Y	N	USA	OpenStack Stein distribution + Kubernetes 1.16.x
Wind River	Cloud Platform 19.12 (K8s)	Wind River @OSM Remote Lab	N	Y	N	Kista, Sweden	WR Cloud Platform - Kubernetes 1.16.x
Wind River	Titanium Cloud R5	Wind River @OSM Remote Lab	Y	N	N	Alameda, CA, USA	WR Titanium Cloud - OpenStack Pike
Wind River - Lenovo	StarlingX	Lenovo POD ThinkSystem SR630/SR650	N	Y	Y	NC, USA	StarlingX R3 - K8s
Wind River - Lenovo	Titanium Cloud R5	Lenovo POD ThinkSystem SR630/SR650	Y	N	Y	NC, USA	Wind River Titanium Cloud R5 - OpenStack Pike
Wind River - QCT	Cloud Platform 19.12 (K8s)	QCT POD	N	Y	N	Taipei, Taiwan	WR Cloud Platform - Kubernetes 1.16.x
Wind River - QCT	Titanium Cloud R5	QCT POD	Y	N	N	Taipei, Taiwan	WR Titanium Cloud - OpenStack Pike
ZTE	CommonEdge	ZTE POD	N	N	Y	Shangai, China	Edge Platform (Conformance only)

Table 4. VIM&NFVIs & MEC Platforms Under Test

5.1.5 OSS/BSS

For the first time, this year's event included an OSS/BSS solution as (optional) part of the system under test.

Organisation	Name	Team Location	Short Description
University of Patras	Openslice	Patras, Greece	Open Source OSS/BSS solution https://openslice.readthedocs.io/en/stable/

Table 5. OSS/BSS Solutions

5.3 Test Functions

In addition to the Test VNFs included in the VNF section, the following Test Functions were available during the Plugtests to support or complement the scope of the Test Sessions. Their use was optional.

Organisation	Name	Team Location	Short Description
Telenity	Test Script	Turkey	Onboarding, Launching, Manual Scaling and Day2 Configuration test script for OSM based solutions
ETSI	Test System	ETSI, France	NFV SOL016 Based Robot Test Suite for NFV IOP Automated testing

Table 6. Test Functions

5.3 Technical Support

The organisations below provided technical support and expertise to the Plugtests Team and contributed actively to the test plan development and technical arrangements to prepare and run the Plugtests.

Organisation	Role
EdgeGap	Technical Support
FScom	Technical Support
Nextworks	Technical Support
Sismondi	Technical Support

Table 7. Technical Support

5.4 Observers

The following organisations joined the NFV Plugtests as observers and contributed with technical advice and test plan review:

Organisation	Role
DOCOMO	Telecommunications service provider
DT	Telecommunications service provider
Indian Institute of Technology (IIT)	Academia
National Chiao Tung University (NCTU)	Academia
NTT	Telecommunications service provider
Orange	Telecommunications service provider
STC	Telecommunications service provider
University of Oulu	Academia

Table 8. Observers

5.5 Open Source Communities

The Open Source communities listed below were actively involved in the Plugtests preparation and contributed to the Test Plan review. Their solutions were widely present in the Test Sessions, sometimes through multiple distributions:

Community / Project	Role	Details
ETSI OSM - Open Source MANO	MANO	https://osm.etsi.org
Kubernetes	VIM&NFVI	https://kubernetes.io/
ONAP	MANO	https://www.onap.org
Openslice	OSS/BSS	https://openslice.io
Open Stack	VIM&NFVI	https://www.openstack.org
StarlingX	VIM&NFVI	https://www.starlingx.io

Table 9. Supporting Open Source communities

6 Test Infrastructure

6.1 HIVE

The remote integration, pre-testing, API conformance and multi-vendor interoperability testing were enabled by the NFV Plugtests Programme's HIVE network

Figure 10. NFV Plugtests HIVE network

The NFV HIVE (Hub for Interoperability and Validation at ETSI) network securely interconnects participants' remote labs and Functions under Test and allows for remote multi-party interoperability testing and validation activities. Over 20 remote locations including several OSM Remote Labs participating to the Plugtests leveraged the HIVE network to make their Functions Under Test available for the test sessions.

The HIVE network was a key asset for a successful fully remote event. The figure below describes how the NFV and OSM HIVE networks are interconnected to support the NFV Plugtests activities.

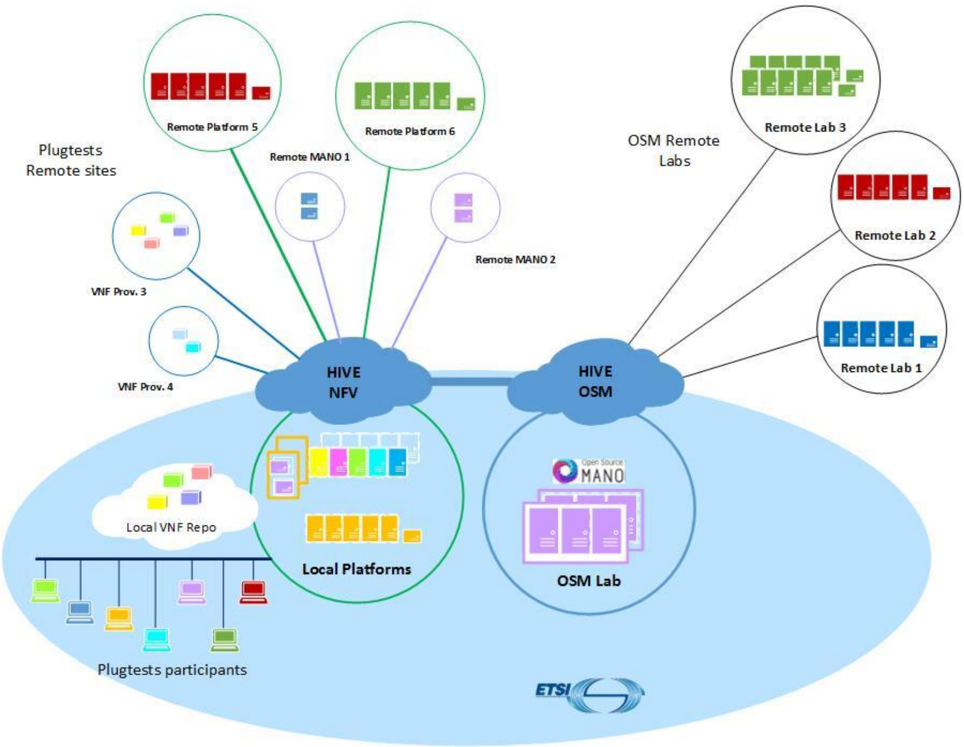

Figure 11. Remote Test Infrastructure

6.2 Test Automation Platform

The API Conformance Test Sessions relied on a Test System acting as API consumer for the NFV and MEC APIs exposed by different NFV and MEC components over different reference points of the respective architectural frameworks. The capabilities offered by the test system were:

- Sending configurable HTTP(S) requests
- Allowing custom payloads to be exchanged
- Automatically applying headers validation on the response payloads
- Automatically applying schema validation on the response payloads
- Receiving notifications

The test system was deployed as a set of Testing Tools in the new HIVE Test Automation Platform (TAP), able to run the Robot Framework and TTCN-3 Test Suites developed for [TST010] and [MEC032]. The Platform orchestrates test session executions in all the required steps, including:

- Configuration of the test system w.r.t Implementation Conformance Statements and implementation details,
- Test selection and execution, with detailed interaction with the user, and
- Test reporting and logging, to enable results collection and issues resolution.

The execution of the tests was triggered on demand by the participants, in a self-service fashion. A demo and a detailed presentation of the new platform was provided to participants during the preparation of the event.

The HIVE Test Automation Platform acts as a generic test orchestrator, able to transparently execute Test Suites implemented with different frameworks and tailored for different technologies. Ad-hoc components have been developed for NFV and MEC conformance testing, which can be reused in future activities and will be actively developed.

The usage of the HIVE TAP (with respect to the execution of sessions manually operated by a member of the Plugtests Team), led to a higher automation of the test execution which enabled a deeper learning of the Test Suites and the workflow of their execution. The learnings will be contributed back to ISG NFV and MEC (respectively in the TST and DECODE WGs) and a set of fixes has been made available in the code base during the event itself.

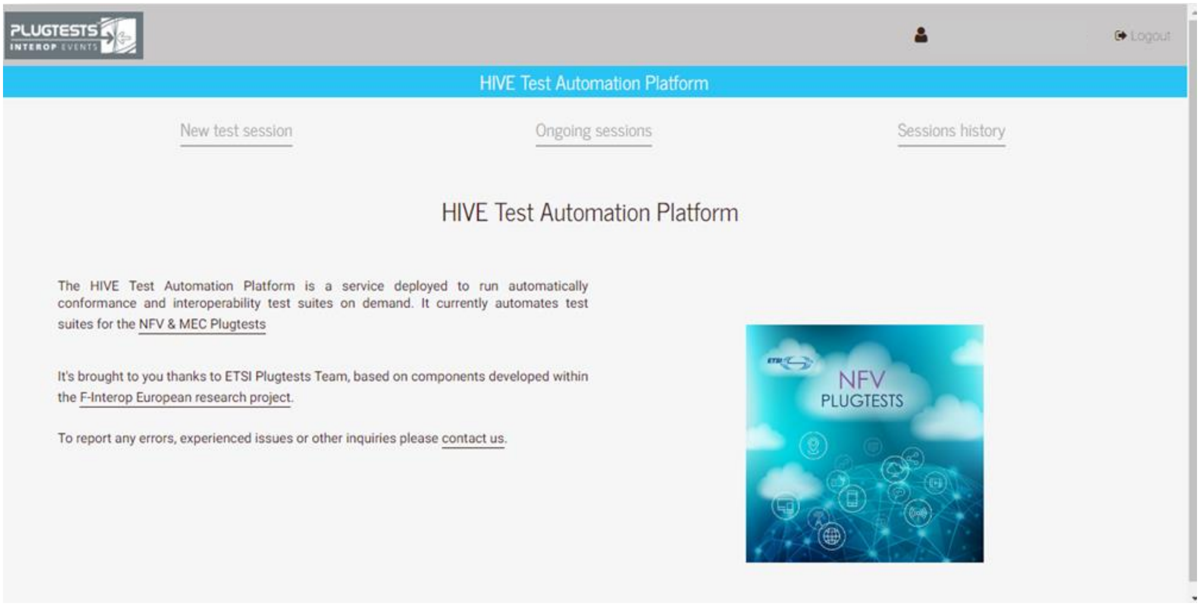

Figure 12. HIVE TAP - Test Automation Platform

7 Test Procedures

7.1 Remote Integration Procedures

Remote integration procedures for different FUTs and FUT combinations were documented in the WIKI (see next chapters) and the progress captured in a multi-dimensional tracking matrix which was reviewed regularly during the preparation calls. The tracking matrix was like the one shown below.

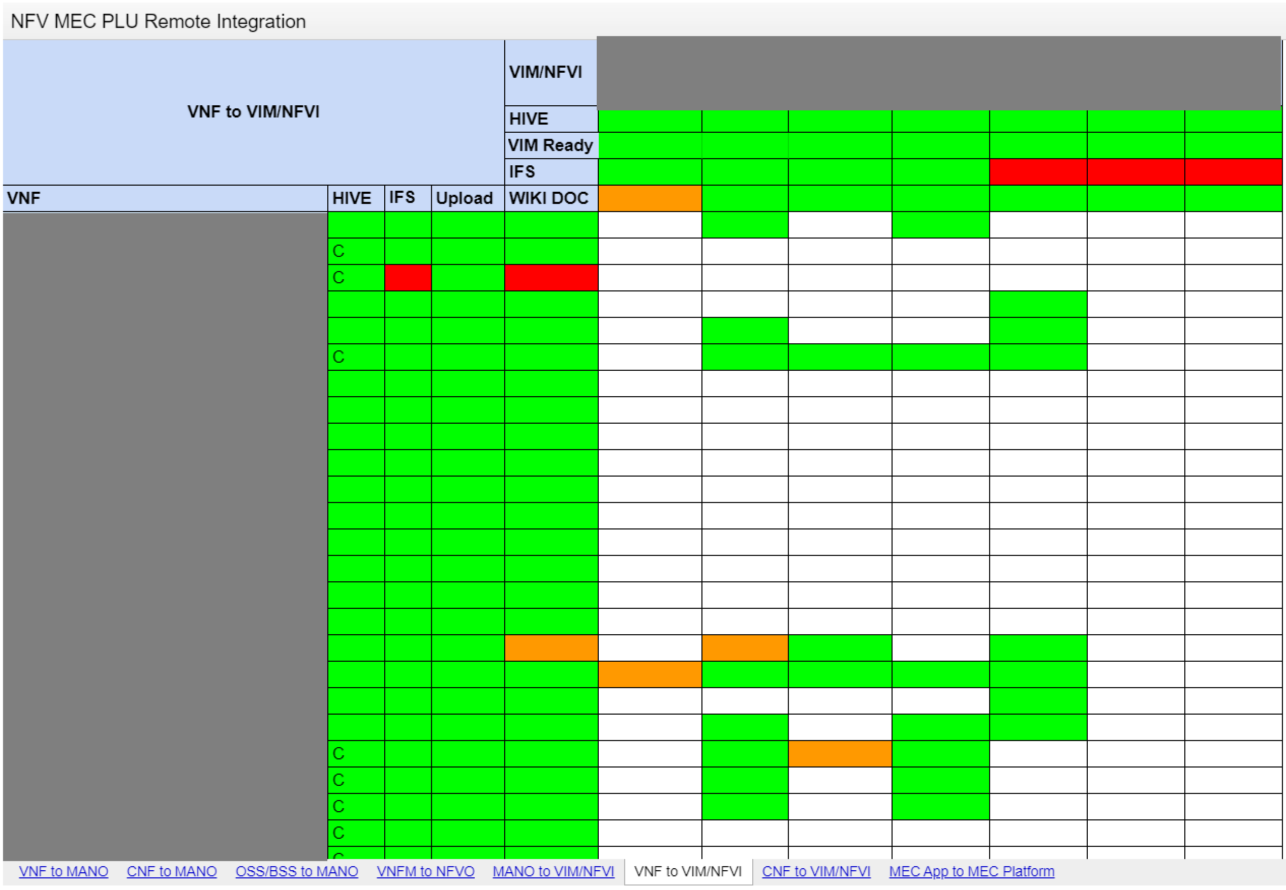

Figure 13. Pre-testing tracking matrix

The progress on each of the dimensions of the remote integration activities (VNF to MANO, MANO to VIM&NFVI VNF to VIM&NFVI and NS to MANO) was tracked following the colour code described below:

Pre-testing Matrix Colour Code	
	Not Started / Not Reported
	Not Applicable
	Pending
	Ongoing
	Completed

Table 10. Remote integration matrix colour code

7.1.1 Per-FUT Procedures

The next sections describe the independent procedures for each type of FUT:

7.1.1.1 VNF / CNF / MEC App

1. **HIVE:** Connect to **HIVE** (site to site or client to site)
2. **IFS:** Fill in the VNF Interoperability Feature Statement form
3. **Upload**: Upload VNF / CNF /MEC App Package(s), descriptors, artifacts, and SW image(s) to the central VNF Repository / Docker Registry
4. **WIKI DOC**: Provide a schema describing your VNF / NS

7.1.1.2 MANO

1. **HIVE:** Connect to **HIVE**. site-to-site VPN connectivity is required for MANO solutions
2. **IFS:** Fill in the MANO Interoperability Feature Statements form
3. **WIKI DOC**: provide VNF and NS Descriptor examples.

7.1.1.3 VIM&NFVI / MEC Platform

1. **HIVE:** Connect to **HIVE** with site-to-site VPN
2. **VIM Ready**: Install, start and configure VIM
3. **IFS:** Fill in the VIM&NFVI Interoperability Feature Statements form
4. **WIKI DOC**: provide additional information required for MANO solutions to integrate: NBI IP@, tenants, credentials, VNF Management IP pool, etc...

7.1.2 Cross-FUT Procedures

7.1.2.1 MANO to VIM&NFVI

1. Test connectivity from MANO to VIM&NFVI & from VIM&NFVI to MANO
2. Connect MANO to VIM (get NBI IP, project and credentials on VIM&NFVI wiki page)
3. Verify VIM resources can be accessed from MANO
4. Specify the reference VNF in the pre-testing table
5. Upload Ref VNF to VIM
6. On-board, instantiate and terminate Reference VNF from MANO

7.1.2.2 VNF / CNF to VIM&NFVI

1. Upload VNF / CNF image to VIM
2. Verify physical network connectivity in NFVI will allow for VNF/NS deployment
3. Manual creation of VM's / K8s clusters and network infrastructure required for this VNF / CNF (to prepare MANO On-board automatic execution)
4. Manual execution Instantiate VNF /CNF steps (to prepare MANO automatic execution)

5. Manual execution Terminate VNF / CNF steps (to prepare MANO automatic execution)

7.1.2.3 VNF / CNF to MANO

1. Identify the reference VIM&NFVI(s) that will be used for pre-testing
2. Create VNFD/NSD for MANO, and upload to VNF Repository
3. On-board VNFD/NSD to MANO
4. Upload VNF /CNF image(s) to Ref VIM
5. Instantiate VNF/VNF/NS from MANO
6. Scale in/out
7. Terminate VNF/VNF/NS

7.1.2.4 MEC App to MEC Platform

8. On-board MEC App to MEC Plat
9. Upload MEC App image(s) to MEC Plat
10. Instantiate MEC App from MEC Plat
11. Execute tests
12. Terminate MEC App

7.2 NFV IOP Testing Procedure

During the Plugtests week, a daily Test Session Schedule was produced with the Plugtests Scheduler. Test Sessions were organised in several parallel tracks, ensuring that all participants had at least one pre-scheduled Test Session every day. Pre-scheduled test sessions could be completed with additional sessions addressing other test configurations.

Participants could choose to run the above-mentioned additional testing as “freestyle” test sessions, and create a test report on the fly, or to ask the Plugtests team to schedule the additional sessions for them.

During each test session the procedure for interoperability testing was as follows:

1) The MANO, VIM&NFVI(s), and VNFs/CNFs representatives would create and join a dedicated Slack channel.

2) One representative of the team opened the Test Session Report and the Test Plan.

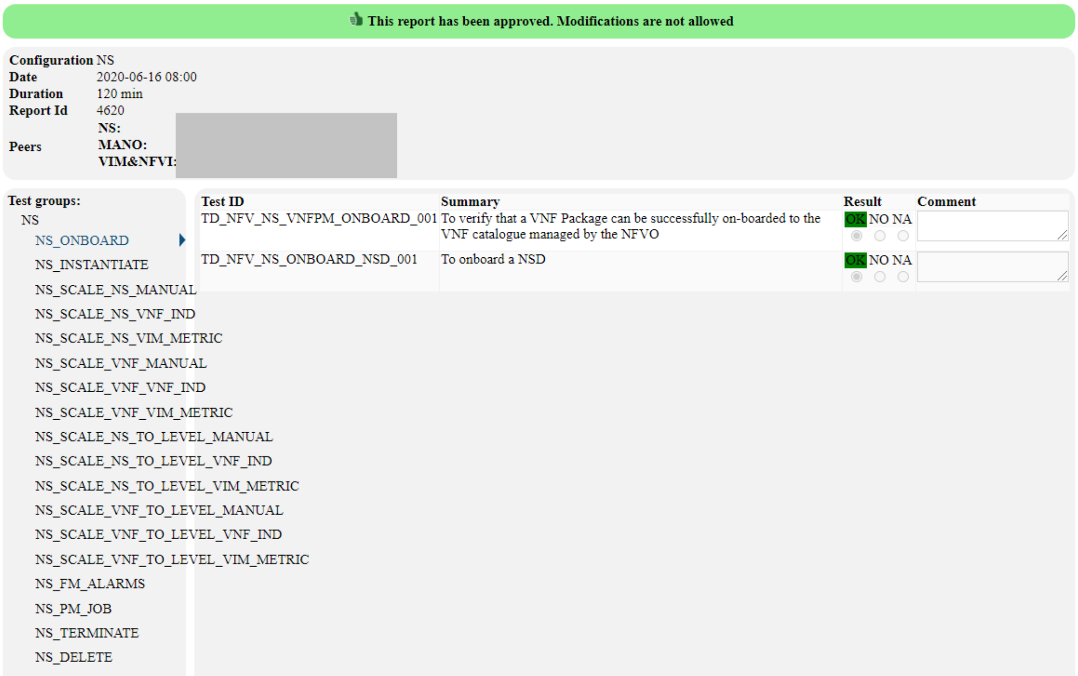

Figure 14. Test Session Report

3) For each Test in each group of the Test Plan:

a. The corresponding Test Description was applied to the target SUT Configuration:

SUT_NS

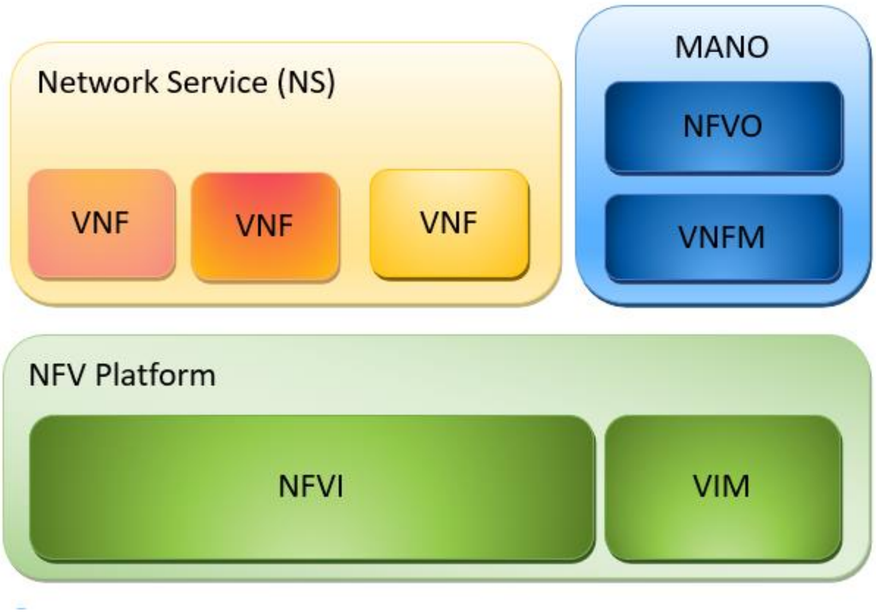

Figure 15. SUT Configuration example.

b. VNFs, MANO and VIM&NFVI providers jointly executed the different steps specified in the Test Description and evaluated interoperability through the different IOP Checks prescribed in it.

Interoperability Test Description				
Identifier	TD_NFV_NS_LCM_SCALE_OUT_001			
Test Purpose	To verify that a NS can be successfully scaled out (Scale_NS) by an operator			
Configuration	SUT_SINGLE-VENDOR_NS SUT_MULTI-VENDOR_NS SUT_S-VNFM SUT_AUTO-LCM-VALIDATION			
References	[IFA013] Clause 7.3.3 [IFA007] Clause 7.2.4 [IFA005] Clause 7.3, 7.4, 7.5 [IFA006] Clause 7.3, 7.4, 7.5 [SOL005] Clause 6.3 [SOL003] Clause 5.4.4			
Applicability	* [IFS_NFV_NFVO_5] NFVO supports NS scaling by adding/removing VNF instances * [IFS_NFV_NS_10] NS can scale out/in by adding/removing VNF instances (Scale_NS)			
Pre-test conditions	* NS is instantiated (TD_NFV_NS_LCM_INSTANTIATE_001)			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1	Stimulus	Trigger NS scale out (Scale_NS) in NFVO with an operator request	
	2	IOP Check	Verify that the VNFM receives instantiation request for the additional VNF(s) to be deployed for the given NS	
	3	IOP Check	If VNFM is in direct mode: • Verify that the VNFM is granted by the NFVO to allocate the virtualised resources required for the additional VNFs in the VIM If VNFM is in indirect mode: • Verify that the VNFM is granted by the NFVO to manage the instantiation of the additional VNFs	
	4	IOP Check	Verify that the additional resources have been allocated in the VIM according to the descriptors	
	5	IOP Check	Verify that the additional VNF instance(s) are running and reachable from the management network	
	6	IOP Check	Verify that the initial configuration for the additional VNF(s) has been successfully applied	
	7	IOP Check	Verify that the additional VNF instance(s) in the NS are considered INSTANTIATED by the VNFM	
	8	IOP Check	Verify in the NFVO that the NS has been scaled out (i.e. query or display the NS instance resource)	
	9	IOP Check	Verify that NS is functional by running the end-to-end functional test	
IOP Verdict				

Table 11. Test Description example

c. The Test Result was reported to the Test Session Report, as follows:

i. OK: all IOP Checks were successful

ii. NOK: at least one IOP Check failed. A comment was requested.

iii. NA: the feature was not supported by at least 1 of the involved FUTs. A comment was requested to clarify the missing feature.

4) Once all the tests in the Test Session Report were executed and results recorded, all the involved participants reviewed the Report and approved it.

7.3 NFV API Testing Procedure

The NFV API Conformance Test Sessions aimed at validating the conformance of the participants FUTs to the SOL002, SOL003 and SOL005 API specifications, while validating the API Conformance Robot Test Suite. The Test System was run on the HIVE TAP, which provided the connectivity to the participating FUTs. The Test System executed the Robot Framework Test Suites developed for the NFV API Conformance Test Specification [TST010] - in both the published 2.4.1 version and in the stable draft of version 2.6.1 - and made available via the ETSI Forge [NFV-ROBOT-TS].

Each test session was executed on-demand and in a self-service fashion, to allow maximum flexibility and scability of execution resources (e.g. w.r.t. timezone differences, number of parallel sessions).

Within each test session, the user would – via the workflow implemented in HIVE TAP – execute the following steps:

1. Log into the HIVE TAP, with credentials created for each participating team;

2. Select the API (i.e. NFV Interface) to be tested, e.g. NS LifecycleManagement over SOL 005.

3. Fill in the configuration of the test system, i.e. providing values for the variables defined in the Robot resource files. The variables were automatically collected from the Robot test suite and presented in a form for the user, who could fill them in individually or as a JSON data structure (enabling reuse of configuration settings);

4. After activation of the test session (which comprised initialization of a dedicated test environment and instantiation of the specifically configured test system), the user could execute and skip individual groups of tests within the Test Suite for the selected API. Groups were defined by the individual Robot files in the test suite.

5. After the execution of all tests in the Robot File, the user was presented with the possibility to download the detailed test reports as produced by the Robot executor, in both human readable (HTML) and machine readable (XML) formats.

6. After execution of all Robot files, the user was presented with the possibility to restart or terminate the test sessions.

Results where automatically collected by the platform to allow the generation of aggregated statistics.

7.4 MEC IOP Testing Procedure

The MEC Interoperability test sessions were organized among participants according to the capability of their FUTs and the test cases selected by the participants. The test session run in parallel with session for other tracks.

Organized sessions were allocated in the Plugtests Scheduler for the coordination of participants. For each test session, a specific test configuration was assigned, and a set of multi-vendor components selected for the participation.

Test sessions were executed remotely and facilitated by use of communication channels such as live chats (with dedicated private spaces for individual sessions) and audio/video conferencing tools (provided either by the Plugtests Team or the participants). Given the novelty of the MEC Track, the Plugtests team participated in almost all the test sessions to provide support and coordinate the actions.

For each session, the same methodology of NFV Interoperability track were executed, i.e:

- The test plan and the test descriptions applicable to the test session were reviewed,
- Each test description was jointly executed by the participants, by executing stimuli and verifications,
- Outcome of each test descriptions were reported and validated by the participants.

7.5 MEC API Testing Procedure

The MEC API Conformance Test Sessions aimed at validating the conformance of the participants FUTs to the MEC011, MEC012 and MEC013 API specifications, while validating the API Conformance Robot and TTCN-3 Test Suites. As for the NFV API track, the Test System was run on the HIVE TAP, which provided the connectivity to the participating FUTs. The Test System executed the Robot Framework Test Suites developed for the MEC API Conformance Tests [MEC-DEC032-3] and made available via the ETSI Forge ([MEC-ROBOT-TS] and [MEC-TTCN3-TS].

The testing procedure followed the same methodology and process as descripted in clause 7.3 of the present document.

8 Test Plans Overview

8.1 NFV Interoperability

The NFV Interoperability test sessions were based on [TST007]. The following clauses summarise the different configurations and interoperability test cases in scope for this Plugtests, and how they were grouped to optimise test session scheduling, duration and results collection and analysis.

8.1.1 NS

The NS Configuration was based on the “SUT Configuration 1” in [TST007]. It involves one MANO solution, one VIM&NFVI and one Network Service (NS), including just one on several VNFs.

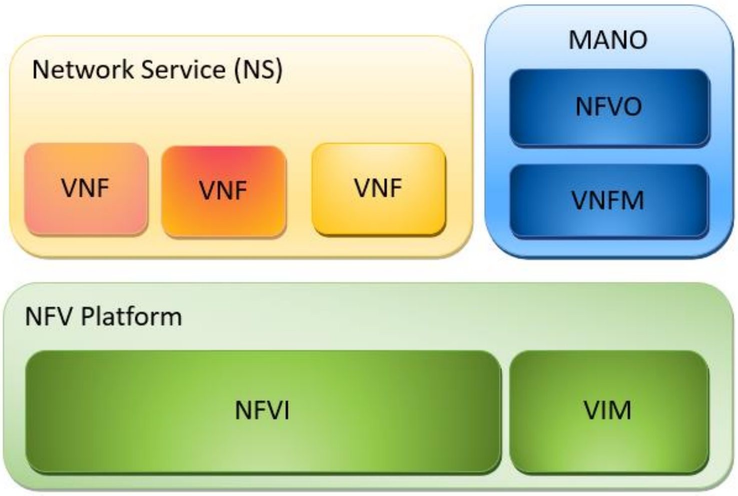

Figure 16: NS SUT Configurations

For this configuration, the table below lists the groups and TDs that apply:

Group	Test IDs	# TDs
NS_ONBOARD	TD_NFV_NS_VNFPM_ONBOARD_001 TD_NFV_NS_ONBOARD_NSD_001	2
NS_INSTANTIATE	TD_NFV_NS_NSLCM_INSTANTIATE_001	1
NS_SCALE_NS_MANUAL **NS_SCALE_NS_VNF_IND** **NS_SCALE_NS_VIM_METRIC**	TD_NFV_NS_NSLCM_SCALE_OUT_xxx TD_NFV_NS_NSLCM_SCALE_IN_xxx	8
NS_SCALE_VNF_MANUAL **NS_SCALE_VNF_VNF_IND** **NS_SCALE_VNF_VIM_METRIC**	TD_NFV_NS_NSLCM_SCALE_OUT_VNF_xxx TD_NFV_NS_NSLCM_SCALE_IN_VNF_xxx	8
NS_SCALE_NS_TO_LEVEL_MANUAL	TD_NFV_NS_NSLCM_SCALE_TO_LEVEL_001	1

NS_SCALE_VNF_TO_LEVEL_MANUAL **NS_SCALE_VNF_TO_LEVEL_VIM_METRIC**	TD_NFV_NS_NSLCM_SCALE_VNF_TO_LEVEL_xxx	2
NS_UPDATE_VNF	TD_NFV_NS_NSLCM_UPDATE_xxx	4
NS_PM_JOB	TD_NFV_NS_PM_NS_xxx	5
NS_TERMINATE	TD_NFV_NS_NSLCM_TERMINATE_001	1
NS_DELETE	TD_NFV_NS_VNFPM_DELETE_001 TD_NFV_NS_DELETE_NSD_001	2

Table 12. NS Test Groups

The full list and detailed steps of these NFV IOP Test Descriptions can be found in [TST007].

8.1.2 NS CNF

The NS-CNF group was based on the “SUT Configuration 1” in [TST007]. It involves one MANO solution, one VIM&NFVI and one Network Service (NS), including one or several CNFs.

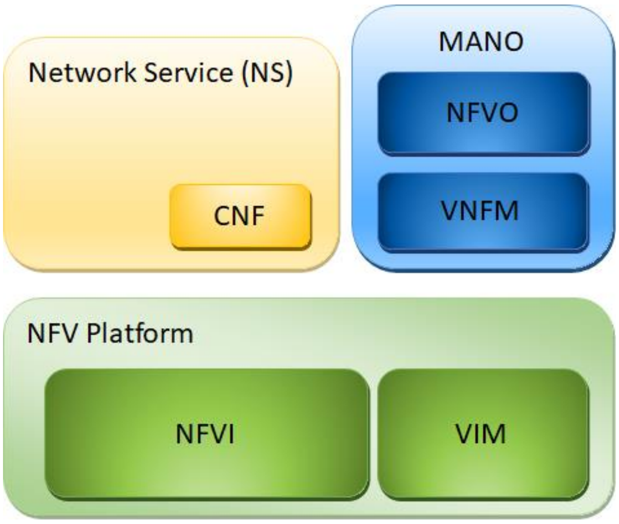

Figure 17: NS-CNF SUT Configurations

For each of these 2 configurations, the table below lists the sub-groups and TDs that apply:

Group	Test IDs	# TDs
NS-CNF_ONBOARD	TD_NFV_NS-CNF_VNFPM_ONBOARD_001 TD_NFV_NS-CNF_ONBOARD_NSD_001	2
NS-CNF_INSTANTIATE	TD_NFV_NS-CNF_NSLCM_INSTANTIATE_001	1
NS-CNF_SCALE_NS_MANUAL	TD_NFV_NS-CNF_NSLCM_SCALE_OUT_xxx TD_NFV_NS-CNF_NSLCM_SCALE_IN_xxx	2
NS-CNF_SCALE_VNF_TO_LEVEL_MANUAL	TD_NFV_NS-CNF_NSLCM_SCALE_VNF_TO_LEVEL_001	1

NS-CNF_UPDATE_VNF	TD_NFV_NS-CNF_NSLCM_UPDATE_xxx	3
NS-CNF_NS_TERMINATE	TD_NFV_NS_NSLCM_TERMINATE_001	1
NS-CNF_NS_DELETE	TD_NFV_NS_VNFPM_DELETE_001 TD_NFV_NS_DELETE_NSD_001	2

Table 13. NS-CNF Test Groups

The full list and detailed steps of these NFV IOP Test Descriptions can be found in [TST007].

8.1.3 Specific VNFM

The Specific VNFM group leverages the "SUT Configuration 1" as described in [TST007].

This configuration involved one MANO solution, one VIM&NFVI, one VNF and a standalone VNF Manager. The Specific VNFM and the NFVO in the MANO solutions were requested to both support the same mode (direct or indirect) for resource management

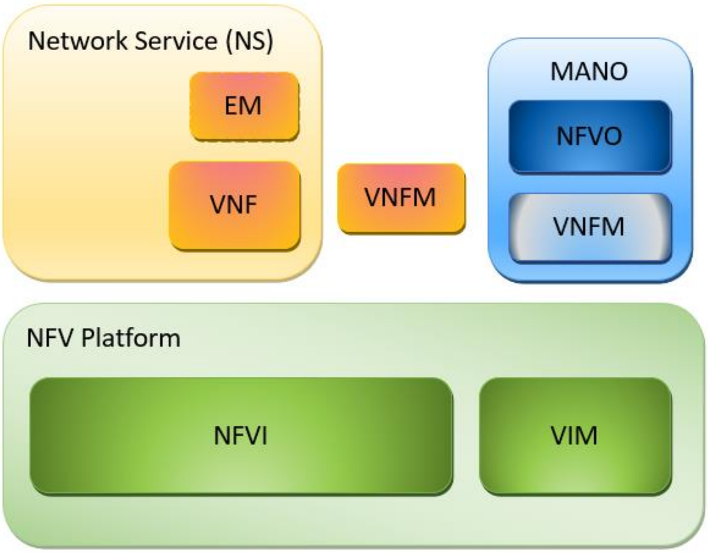

Figure 18: Specific VNFM SUT Configuration

For this configuration, the table below lists the groups and TDs in scope:

Group	Test IDs	# TDs
S-VNFM_ONBOARD	TD_NFV_S-VNFM_VNFPM_ONBOARD_001 TD_NFV_S-VNFM_ONBOARD_NSD_001	2
S-VNFM_INSTANTIATE	TD_NFV_S-VNFM_NSLCM_INSTANTIATE_001	1
S-VNFM_SCALE_NS_MANUAL **S-VNFM_SCALE_NS_VNF_IND** **S-VNFM_SCALE_NS_VIM_METRIC**	TD_NFV_S-VNFM_NSLCM_SCALE_OUT_xxx TD_NFV_S-VNFM_NSLCM_SCALE_IN_xxx	8
S-VNFM_SCALE_VNF_MANUAL **S-VNFM_SCALE_VNF_VIM_METRIC**	TD_NFV_S-VNFM_NSLCM_SCALE_OUT_VNF_xxx TD_NFV_S-VNFM_NSLCM_SCALE_IN_VNF_xxx	5

S-VNFM_SCALE_VNF_TO_LEVEL_MANUAL	TD_NFV_S-VNFM_NSLCM_SCALE_VNF_TO_LEVEL_001	1
S-VNFM_UPDATE_VNF	TD_NFV_S-VNFM_NSLCM_UPDATE_xxx	3
S-VNFM_FM_SUBSCRIPTION **S-VNFM_FM_ALARMS**	TD_NFV_S-VNFM_FM_NS_ALARM_xxx	5
S-VNFM_NS_TERMINATE	TD_NFV_S-VNFM_NSLCM_TERMINATE_001	1
S-VNFM_NS_DELETE	TD_NFV_S-VNFM_VNFPM_DELETE_001 TD_NFV_S-VNFM_DELETE_NSD_001	2

Table 14. Specific VNFM Test Groups

The full list and detailed steps of these NFV IOP Test Descriptions can be found in [TST007].

8.1.4 OSS/BSS

This configuration extends the “SUT Configuration 1” described in [TST007] by adding an OSS/BSS solution or simulator interacting with the MANO over SOL005.

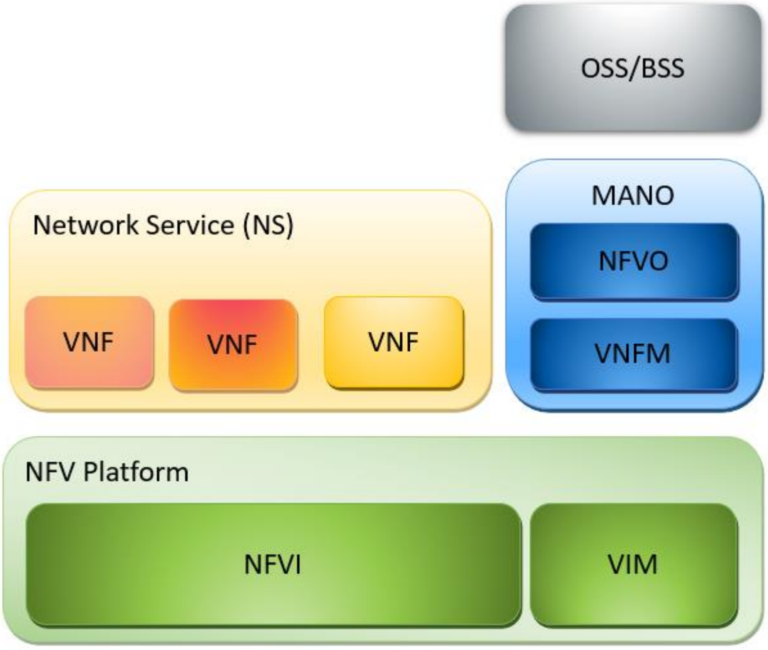

Figure 19. OSS/BSS SUT Configuration

For this configuration, the table below lists the groups and TDs in scope:

Group	Test IDs	# TDs
NS_OSS-BSS_ONBOARD	TD_NFV_OSS-BSS_VNFPM_ONBOARD_001 TD_NFV_OSS-BSS_ONBOARD_NSD_001	2
NS_OSS-BSS_INSTANTIATE	TD_NFV_OSS-BSS_NSLCM_INSTANTIATE_001	1

NS_OSS-BSS_NS_TERMINATE	TD_NFV_OSS-BSS_NSLCM_TERMINATE_001	1
NS_OSS-BSS_NS_DELETE	TD_NFV_OSS-BSS_VNFPM_DELETE_001 TD_NFV_OSS-BSS_DELETE_NSD_001	2

Table 15. OSS/BSS Test Groups

The full list and detailed steps of these NFV IOP Test Descriptions can be found in [TST007].

8.1.5 Multi-site

This configuration was based on the “SUT Configuration 4” described in [TST007]. It involves 1 MANO solution, 2 NFVI&VIMs and 1 NS with several VNFs.

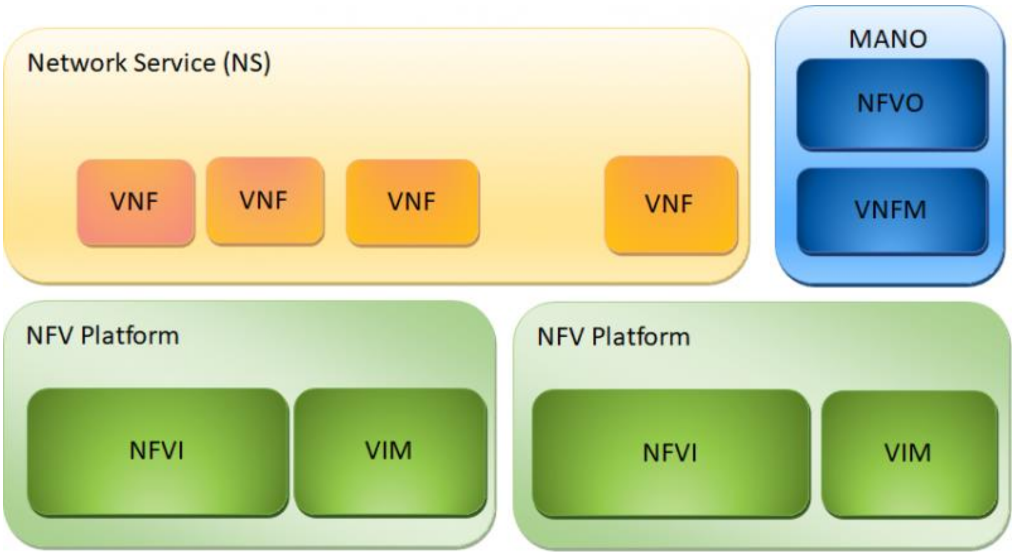

Figure 20. Multi-site SUT Configuration

For this configuration, the table below lists the groups and TDs in scope:

Group	Test IDs	# TDs
MULTISITE_ONBOARD	TD_NFV_MULTISITE_VNFPM_ONBOARD_001 TD_NFV_MULTISITE_ONBOARD_NSD_001	2
MULTISITE_INSTANTIATE	TD_NFV_MULTISITE_NSLCM_INSTANTIATE_001	1
MULTISITE_NS_TERMINATE	TD_NFV_MULTISITE_NSLCM_TERMINATE_001	1
MULTISITE_NS_DELETE	TD_NFV_MULTISITE_VNFPM_DELETE_001 TD_NFV_MULTISITE_DELETE_NSD_001	2

Table 16. Multi-site Test Groups

The full list and detailed steps of these NFV IOP Test Descriptions can be found in [TST007].

8.1.6 Multi-VNFM

This was a variant of the Multi-Site configuration, where stand-alone VNFMs were also deployed on each site.

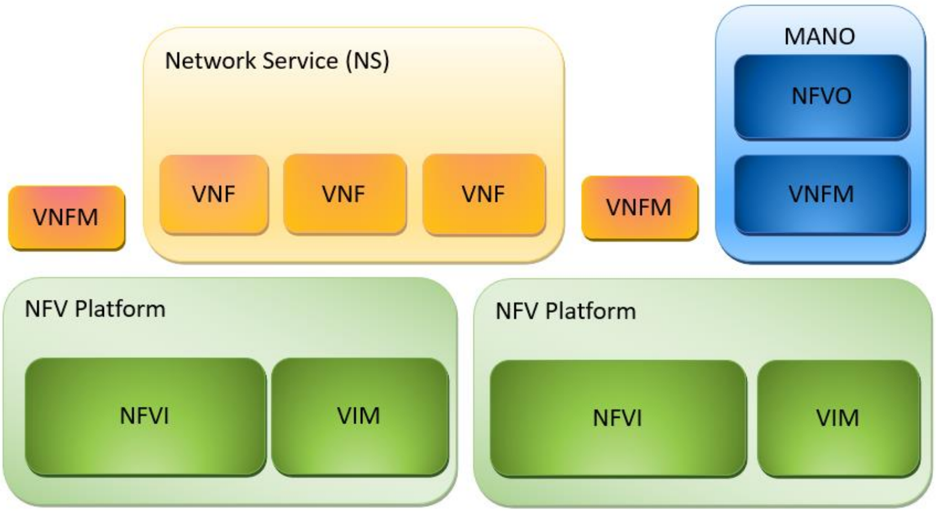

Figure 21. Multi-VNFM SUT Configuration

For this configuration, the table below lists the groups and TDs in scope:

Group	Test IDs	# TDs
MULTI_S-VNFM_ONBOARD	TD_NFV_MULTI_S-VNFM_VNFPM_ONBOARD_001 TD_NFV_MULTI_S-VNFM_ONBOARD_NSD_001	2
MULTI_S-VNFM_INSTANTIATE	TD_NFV_MULTI_S-VNFM_NSLCM_INSTANTIATE_001	1
MULTI_S-VNFM_SCALE_VNF_MANUAL	TD_NFV_MULTI_S-VNFM_NSLCM_SCALE_OUT_VNF_001 TD_NFV_MULTI_S-VNFM_NSLCM_SCALE_IN_VNF_001	2
MULTI_S-VNFM_TERMINATE	TD_NFV_MULTI_S-VNFM_NSLCM_TERMINATE_001	1
MULTI_S-VNFM_DELETE	TD_NFV_MULTI_S-VNFM_VNFPM_DELETE_001 TD_NFV_MULTI_S-VNFM_DELETE_NSD_001	2

Table 17. Multi-VNFM Test Groups

The full list and detailed steps of these NFV IOP Test Descriptions can be found in [TST007].

8.1.7 Automated IOP Testing

During the NFV&MEC Plugtests 2020 preparation, a new Robot Framework Test Suite, leveraging existing API Conformance Test Cases, was developed to automate IOP testing according to [SOL016] workflow specifications.

To test the flows, a test configuration was defined so that the Robot Framework test system acted as the OSS/BSS (SOL005 client) to trigger initial actions and evaluate final results on the NFVO over [SOL005], and interacted with VNFM an VNF over [SOL003] and [SOL002] to perform intermediate checks.

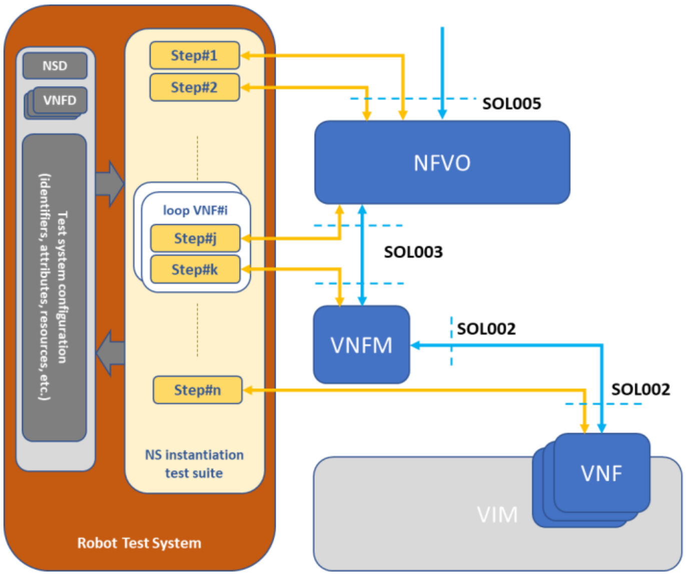

Figure 22: SOL016 based Automated IOP.

The Automated IOP test configuration had 2 variants:

1. In the first one the full MANO (NFVO and VNFM) was provided by the same participant

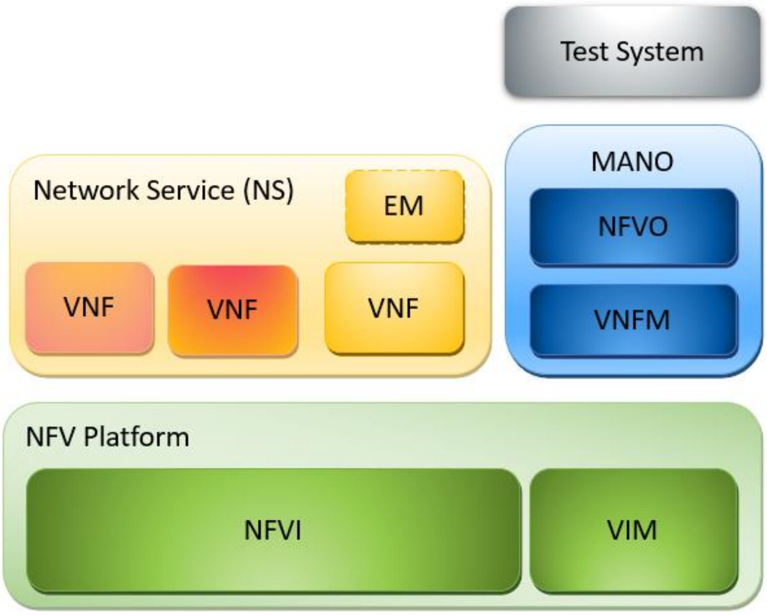

Figure 23: Automated IOP Testing SUT Configuration

For this test configuration, with combined MANO and VNFM, the table below lists the groups and TDs in scope

Group	Test IDs	# TDs
AUTO_IOP_ONBOARD	TD_NFV_AUTO_IOP_VNFPM_ONBOARD_001 TD_NFV_AUTO_IOP_ONBOARD_NSD_001	2
AUTO_IOP_INSTANTIATE	TD_NFV_AUTO_IOP_NSLCM_INSTANTIATE_001	1
AUTO_IOP_SCALE_VNF_MANUAL	TD_NFV_AUTO_IOP_NSLCM_SCALE_OUT_VNF_001 TD_NFV_AUTO_IOP_NSLCM_SCALE_IN_VNF_001	2
AUTO_IOP_PM_JOB	TD_NFV_AUTO-IOP_PM_NS_xxx	2
AUTO_IOP_NS_TERMINATE	TD_NFV_AUTO_IOP_NSLCM_TERMINATE_001	1
AUTO_IOP_NS_DELETE	TD_NFV_AUTO_IOP_VNFPM_DELETE_001 TD_NFV_AUTO_IOP_DELETE_NSD_001	2

Table 18. AUTO IOP Test Groups

2. In the second one where the NFVO and the VNFM were provided by different participants.

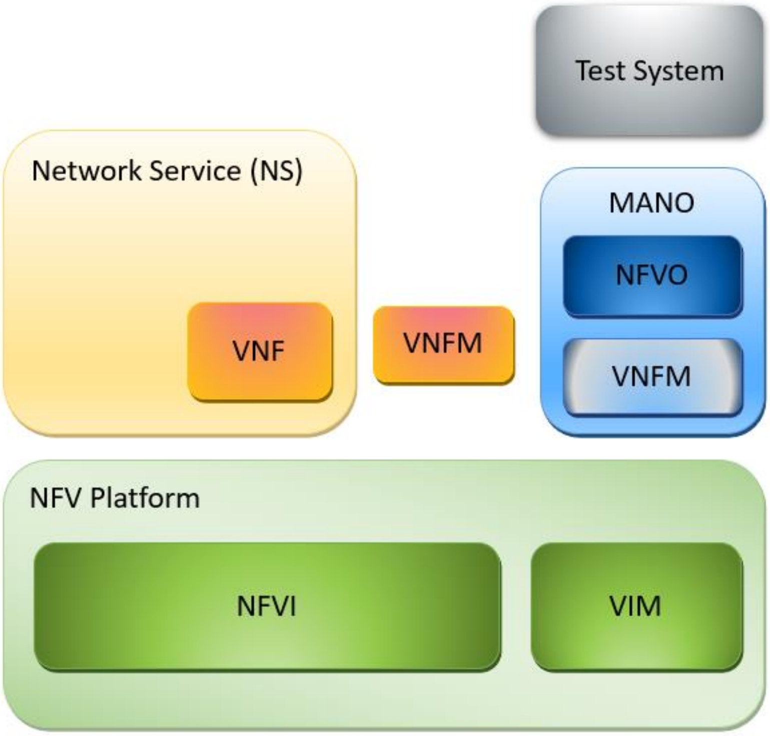

Figure 24: Automated IOP Testing (S-VNFM) SUT Configuration

For this second configuration with Specific-VNFM, the table below lists the groups and TDs in scope

Group	Test IDs	# TDs
AUTO_IOP_S-VNFM_ONBOARD	TD_NFV_AUTO_IOP_S-VNFM_VNFPM_ONBOARD_001 TD_NFV_AUTO_IOP_S-VNFM_ONBOARD_NSD_001	2
AUTO_IOP_S-VNFM_INSTANTIATE	TD_NFV_AUTO_IOP_S-VNFM_NSLCM_INSTANTIATE_001	1
AUTO_IOP_S-VNFM_NS_TERMINATE	TD_NFV_AUTO_IOP_S-VNFM_NSLCM_TERMINATE_001	1
AUTO_IOP_S-VNFM_NS_DELETE	TD_NFV_AUTO_IOP_S-VNFM_VNFPM_DELETE_001 TD_NFV_AUTO_IOP_S-VNFM_DELETE_NSD_001	2

Table 19. AUTO IOP S-VNFM Test Groups

The complete Robot Framework Automated IOP Test Suite can be found in [NFV-AUTO-IOP] repository.

8.2 NFV API Conformance

This NFV API Conformance test plan was based on the Robot Framework Test Cases developed for [TST010] NFV API Conformance Test Specification, addressing FUT API Conformance to [SOL002], [SOL003] and [SOL005] specifications.

In addition to the above-mentioned tests plan, an additional NFV API Conformance test plan based on Robot Framework Test Cases was developed for [PLUGTESTS], addressing FUT API Conformance to [SOL016] workflow specifications.

The test system acted as consumer for the NFV API produced by the FUTs, thus focusing only on testing the server-side of the NFV API under Test.

The following clauses summarise the test cases in scope for this NFV API Plugtests, and how they were grouped to optimise the remote test session executions.

The complete Test Specification can be found in [TST010] and the associated Robot Test Cases are available in the ETSI Forge [NFV-ROBOT-TS].

8.2.1 VNF/EM

The test configuration as described in the figure below was defined to test the interfaces exposed by a VNF/EM towards the VNFM, such as VNF Configuration API and VNF Indicator API. The test system acts as the VNFM (SOL002 API Client)

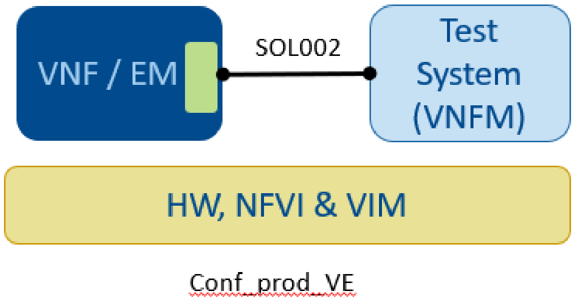

Figure 25: VNF/EM API Test Configuration

8.2.2.1 SOL002

API	Test IDs	# Test Cases
VNF Configuration API	TD_API_VNF_SOL002_VNF_CONF_1 .. 6	6
VNF Indicator API	TD_API_VNF_SOL002_VNF_IND_1 .. 36	36

Table 20. VNF/EM SOL002 Test Groups

The complete list of Test Cases can be found in [TST010] Clause 6.3

8.2.2 VNFM

The test configuration as described below was defined to test the interfaces exposed by a VNFM towards VNF/EM (SOL002) or towards the NFVO (SOL003). The test system is acting either as VNF/EM (SOL002 API client) or as NFVO (SOL003 API client)

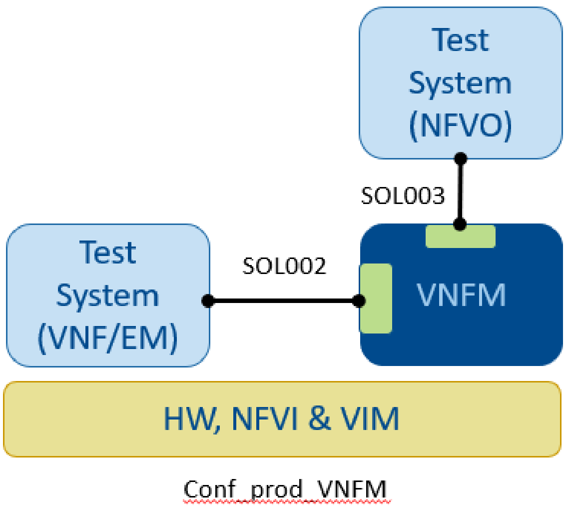

Figure 26: VNFM API Test Configuration

8.2.2.1 SOL002

API	Test IDs	# Test Cases
VNF Fault Management API	TD_API_SOL002_VNF_FM_1..39	39
VNF Life Cycle Management API	TD_API_SOL002_VNF_LCM_1..139	139
VNF Performance Management API	TD_API_SOL002_VNF_PM_1..64	64

Table 21. VNFM SOL002 Test Groups

The complete list of Test Cases can be found in [TST010] Clause 6.3

8.2.2.2 SOL003

API	Test IDs	# Test Cases
VNF Fault Management API	TD_API_SOL003_VNF_FM_1..33	33
VNF Indicator API	TD_API_SOL003_VNF_IND_1..42	42
VNF Performance Management API	TD_API_SOL003_VNF_PM_1..64	64
VNF Life Cycle Management API	TD_API_SOL003_VNF_LCM_1..139	139

Table 22. VNFM SOL003 Test Groups

The complete list of Test Cases can be found in [TST010] Clause 7.3

8.2.3 NFVO

The test configuration as described below was defined to test the interfaces exposed by NFVOs towards the VNFM (SOL003) or OSS/BSS (SOL005). The test system is acting as the VNFM (SOL003 Client) or OSS/BSS (SOL005 client)

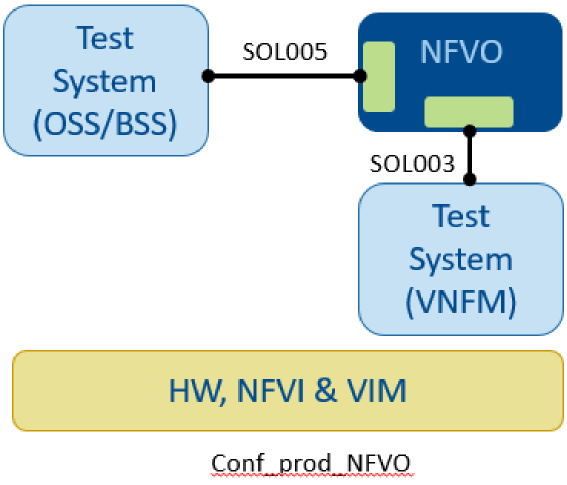

Figure 27: NFVO API Test Configuration

8.2.3.1 SOL003

API	Test IDs	# Test Cases
VNF Package Management API	TD_API_SOL003_VNF_PCKGM_1..75	75
VNF Lifecycle Operation Granting API	TD_API_SOL003_VNF_LCM_GRANT_1..14	14
VNF Virtual Resource Quota Available Notification	TD_API_SOL003_VR_QUOTA_NOTIF_1..19	19

Table 23. NFVO SOL003 Test Groups

The complete list of Test Cases can be found in [TST010] Clause 7.3

8.2.3.2 SOL005

API	Test IDs	# Test Cases
NSD Management API	TD_API_NFVO_SOL005_NSD_MGMT_1..91	91
NS Performance Management API	TD_API_NFVO_SOL005_NS_PM_1..64	64
VNF Package Management API	TD_API_NFVO_SOL005_VNF_PCKGM_1..84	84
NS Fault Management API	TD_API_NFVO_SOL005_NS_FM_1..36	36

Table 24. NFVO SOL005 Test Groups

The complete list of Test Cases can be found in [TST010] Clause 5.3

8.3 MEC Interoperability

The MEC Interoperability test sessions were based on a dedicated test plan developed during the Plugtests preparation [MEC-IOP-TP]. The following clauses summarise the different configurations and interoperability test cases in scope for this Plugtests, and how they were grouped to optimise test session scheduling, duration and results collection and analysis.

8.3.1 MEC Basic

The SUT_MEC_BASIC configuration includes a single MEC application along with a MEC platform. In those tests, the term "MEC Platform" is used to indicate any of the following components: MEC platform, MEC orchestrator or MEC platform manager. The providers of other components of the MEC system such as MEO or MEPM are out of scope. The MEC application runs – together with the MEC Platform - on the MEC host or the NFVI.

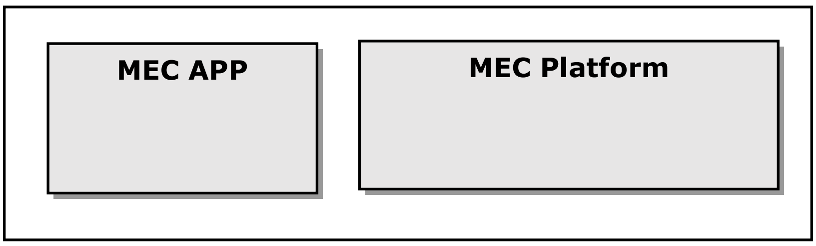

Figure 28: Test configuration SUT_MEC_BASIC

For this configuration, the table below lists the groups and TDs in scope:

Group	Test IDs	# TDs
MEC Application Lifecycle	TD_MEC_APP_ONBOARD TD_MEC_APP_START TD_MEC_APP_STOP TD_MEC_APP_STATUS TD_MEC_APP_CHANGE	5
MEC Services	TD_MEC_SVC_QUERY TD_MEC_SVC_REGISTER TD_MEC_SVC_UPDATE TD_MEC_SVC_DEREGISTER TD_MEC_SVC_TRANSPORTS_QUERY TD_MEC_SVC_QUERYTIME	6
MEC Traffic	TD_MEC_NTW_ACTIVATE TD_MEC_NTW_UPDATE TD_MEC_NTW_DEACTIVATE TD_MEC_NTW_DNS_ACTIVATE TD_MEC_NTW_DNS_DEACTIVATE	5

Table 24. MEC Basic Test Groups

The complete list of Test Cases can be found in [MEC-IOP-TP] clause 8.

8.3.2 MEC Services with Single App

The SUT_MEC_SERVICES_SINGLE_APP test configuration is similar to the configuration SUT_MEC_BASIC, with a difference on the integration between the two elements. In this configuration, one (1) MEC application runs with in the MEC Host alongside the MEC platform. The configuration focuses on the capabilities around MEC Services such as the capability of applications and the platform to provide and register. The service is registered and available for discovery through the service registry in the MEC platform.

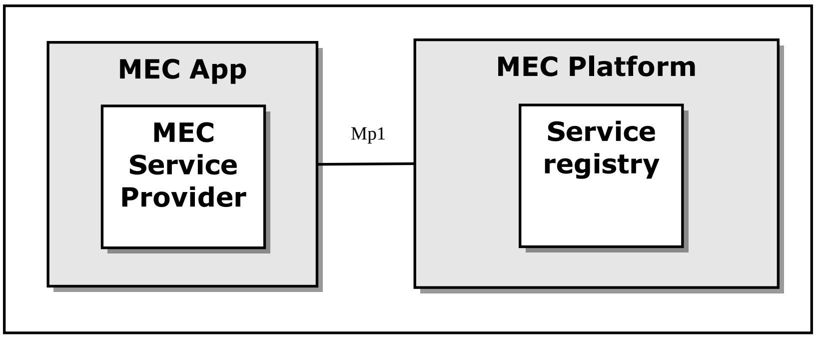

Figure 29: Test configuration SUT_MEC_SERVICE_SINGLE_APP

For this configuration, the table below lists the groups and TDs in scope:

Group	Test IDs	# TDs
MEC Application Lifecycle	TD_MEC_APP_ONBOARD TD_MEC_APP_START TD_MEC_APP_STOP TD_MEC_APP_STATUS TD_MEC_APP_CHANGE	5
MEC Services	TD_MEC_SVC_QUERY TD_MEC_SVC_REGISTER TD_MEC_SVC_UPDATE TD_MEC_SVC_DEREGISTER TD_MEC_SVC_CONSUME	5

Table 25. MEC Services (Single App) Test Groups

The complete list of Test Cases can be found in [MEC-IOP-TP] clause 8.

8.3.3 MEC Services with Multiple Apps

The SUT_MEC_SERVICES_MULTI_APP configuration is similar to the configuration SUT_MEC_SERVICES_SINGLE_APP, with a difference on the integration between both elements. In this configuration, two (2) MEC applications run together alongside the MEC Platform. The configuration focuses on the capabilities around MEC Services such as the capability of applications and the platform to provide, discover or consume MEC services.

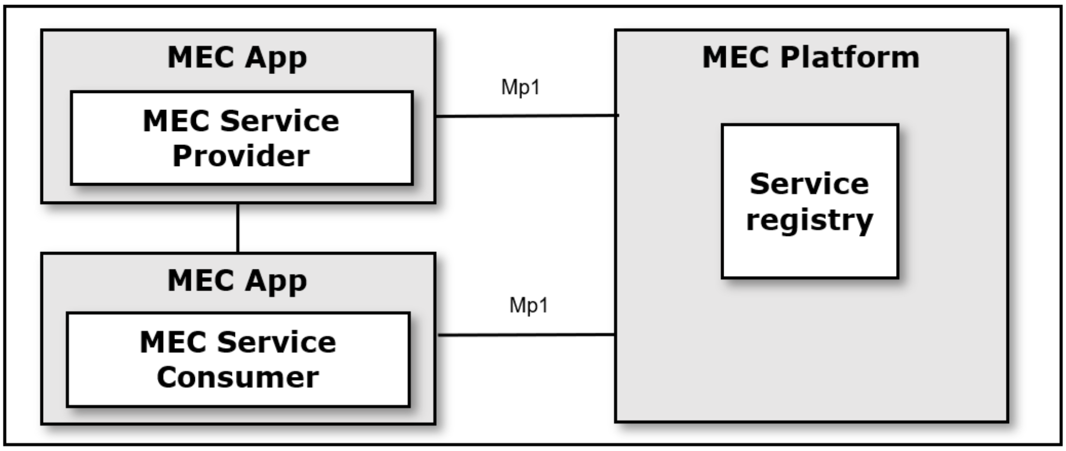

Figure 30: Test configuration SUT_MEC_SERVICE_MULTI_APP

For this configuration, the table below lists the groups and TDs in scope:

Group	Test IDs	# TDs
MEC Application Lifecycle	TD_MEC_APP_ONBOARD TD_MEC_APP_START TD_MEC_APP_STOP TD_MEC_APP_STATUS TD_MEC_APP_CHANGE	5
MEC Services	TD_MEC_SVC_QUERY TD_MEC_SVC_REGISTER TD_MEC_SVC_UPDATE TD_MEC_SVC_DEREGISTER TD_MEC_SVC_CONSUME TD_MEC_SVC_TRANSPORTS_QUERY TD_MEC_SVC_QUERYTIME	7
MEC Traffic	TD_MEC_NTW_ACTIVATE TD_MEC_NTW_UPDATE	5

	TD_MEC_NTW_DEACTIVATE TD_MEC_NTW_DNS_ACTIVATE TD_MEC_NTW_DNS_DEACTIVATE	

Table 26. MEC Services (Multi App) Test Groups

The complete list of Test Cases can be found in [MEC-IOP-TP] clause 8.

8.3.4 MEC in NFV Platforms

The SUT_MEC_NFVI configuration, the MEC platform and the MEC application(s) are hosted and executed on a third party NFV Infrastructure. The focus is on interoperability of virtualization technologies and VIM APIs in a multivendor scenario.

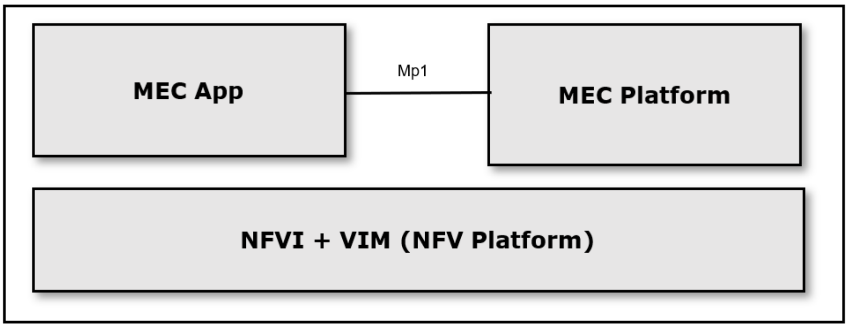

Figure 31: Test configuration SUT_MEC_NFVI

For this configuration, the table below lists the groups and TDs in scope:

Group	Test IDs	# TDs
MEC Application Lifecycle	TD_MEC_APP_ONBOARD TD_MEC_APP_START TD_MEC_APP_STOP TD_MEC_APP_STATUS TD_MEC_APP_CHANGE	5
MEC Traffic	TD_MEC_NTW_ACTIVATE TD_MEC_NTW_UPDATE TD_MEC_NTW_DEACTIVATE TD_MEC_NTW_DNS_ACTIVATE TD_MEC_NTW_DNS_DEACTIVATE	5

Table 27. MEC in NFV Test Groups

The complete list of Test Cases can be found in [MEC-IOP-TP] clause 8.

8.3.5 MEC in NFV Platforms orchestrated by MANO

The SUT_MEC_MANO focuses on the MEC-in-NFV scenario. In this scenario the MEC application(s) and the MEC platform are packaged as VNFs and are managed by a third-party MANO platform in an NFV infrastructure. The availability of other components of the MEC system (such as MEAO, MEPM and specific VNFM) is out of scope.

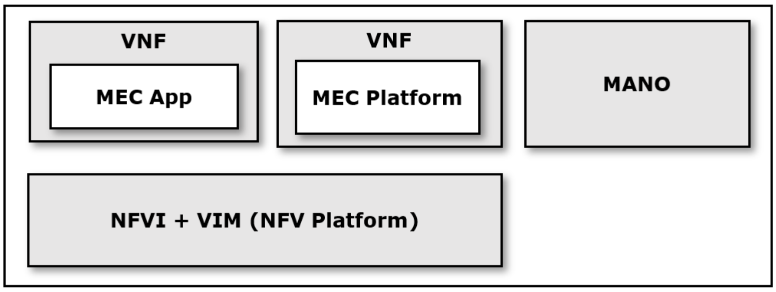

Figure 32: Test configuration SUT_MEC_MANO

For this configuration, the table below lists the groups and TDs in scope:

Group	Test IDs	# TDs
MEC Application Lifecycle	TD_MEC_APP_START TD_MEC_APP_STOP TD_MEC_APP_STATUS TD_MEC_APP_CHANGE	5
MEC Traffic	TD_MEC_NTW_ACTIVATE TD_MEC_NTW_UPDATE TD_MEC_NTW_DEACTIVATE TD_MEC_NTW_DNS_ACTIVATE TD_MEC_NTW_DNS_DEACTIVATE	5

Table 28. MEC MANO Test Groups

The complete list of Test Cases can be found in [MEC-IOP-TP] clause 8.

8.3 MEC API Conformance

The test plan for the MEC API Conformance test track was based upon the latest stable drafts of ETSI GS MEC-DEC 032 [DEC032].

The work item provides a database of test purposes for MEC APIs and implementation in TTCN-3 and Robot Framework as ATS. Both flavours of the test suites were available for Plugtests participants and has been tailored to be executed over the HIVE TAP.

Based on the capabilities and selections of the participating FUTs, the API Tests prepared and executed during the NFV&MEC Plugtests 2020 targeted the MEC specifications [MEC-011], [MEC-012] and [MEC-013]. The test suites were publicly available at [MEC-ROBOT-TS] and [MEC-TTCN3-TS].

The test suite structure followed [DEC032], Clause 5.

9 Results

9.1 NFV Interoperability

9.1.1 Overall Results

During the Plugtests, a total of 89 NFV interoperability Test Sessions were run with different combinations of the Functions Under Test (FUTs) in scope: VNFs, MANOs, VIM&NFVI, OSS/BSS and Test Systems.

The following sections provide an overview of the reported results: overall, per test group, pert tests case. To facilitate the analysis, results are presented as follows:

Result	Meaning
OK	Test Case run. Interoperability (or API test) successfully achieved.
NO	Test Case run. Interoperability (or API test) not achieved.
NA	Not Applicable: Feature not supported by one or more Functions Under Test
Run	Total number of Test Cases Run = OK + NO
Total	Total number of Test Cases = OK + NO + NA = Run + Not Run

Table 29: Results Interpretation

Note that the tests cases for which no result was reported (i.e. when the test session run out of time) are not considered in the Total Results

The table below provides the overall results (aggregated data) for all the test cases run during the NFV Interoperability Test Sessions, from all participating companies:

Overall Results	Number of Test Sessions	Interoperability (TCs Run)		TCs Not Run	TCs Totals	
		OK	NO	NA	Run	Total
	89	702	49	565	751	1316

Table 30: NFV IOP Overall Results

During each Test Session, depending on the targeted configuration and features to be tested, a different number of test cases were offered to the involved participants.

The interoperability test plans included 112 test cases, organised in different groups as described in clause 8.1. Through the 89 Test Sessions run, a total of 1316 Test Results were reported. This figure includes both the executed and non-executed test cases. Overall, a total of 751 individual test cases were executed and results (OK or NO as per table above) reported for them.

	Interoperability		Not Executed	Totals		Totals			
	OK	NO	N/A	Run	Results	% Run	% OK	% NO	% N/A
NS	448	11	367	459	826	55,57%	97,60%	2,40%	44,43%
NS-CNF	69	5	34	74	108	68,52%	93,24%	6,76%	31,48%
S-VNFM	111	21	129	132	261	50,57%	84,09%	15,91%	49,43%
OSS/BSS	12	0	0	12	12	100,00%	100,00%	0,00%	0,00%
Multi-Site	12	0	0	12	12	100,00%	100,00%	0,00%	0,00%

MULTI-S-VNFM	11	0	5	11	16	68,75%	100,00%	0,00%	31,25%
Auto-IOP	36	11	28	47	75	62,67%	76,60%	23,40%	37,33%
Auto-IOP S-VNFM	3	1	2	4	6	66,67%	75,00%	25,00%	33,33%
TOTAL	702	49	565	751	1.316	57,07%	93,48%	6,52%	42,93%

Table 31. NFV IOP Overall results per Group

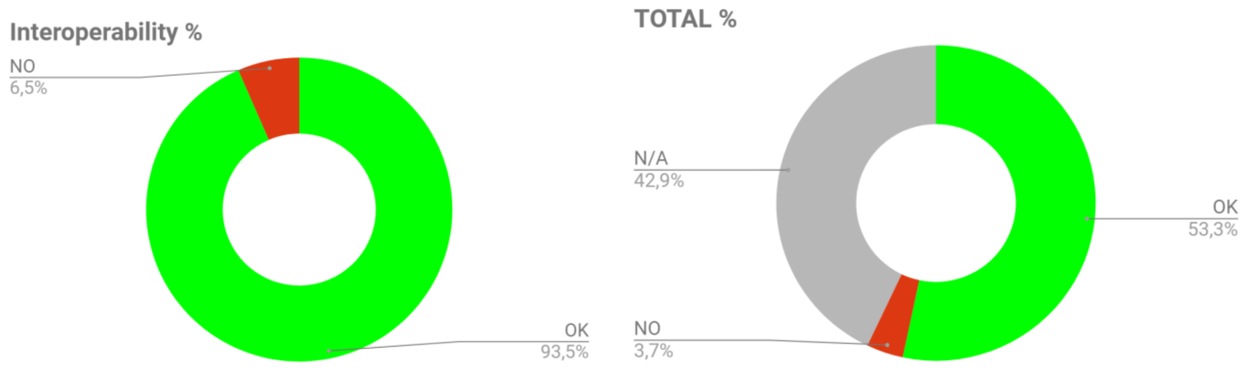

Figure 33. NFV IOP Overall results (%)

The next clauses present more detailed results per test group and test cases and will allow to identify the areas and features with higher execution and interoperability rates.

9.1.2 Results per Group

9.1.2.1 NS

The table and figure below provide an overview of the results for the NS group. Overall, 45 NS test sessions were run.

	Interoperability		Not Executed	Totals		Totals			
	OK	**NO**	**NA**	**Run**	**Results**	**% Run**	**% OK**	**% NO**	**% NA**
NS_ONBOARD	89	0	0	89	89	100,00%	100,00%	0,00%	0,00%
NS_INSTANTIATE	44	1	0	45	45	100,00%	97,78%	2,22%	0,00%
NS_SCALE_NS_MANUAL	44	0	40	44	84	52,38%	100,00%	0,00%	47,62%
NS_SCALE_NS_VNF_IND	16	0	32	16	48	33,33%	100,00%	0,00%	66,67%
NS_SCALE_NS_VIM_METRIC	6	0	40	6	46	13,04%	100,00%	0,00%	86,96%
NS_SCALE_VNF_MANUAL	22	0	34	22	56	39,29%	100,00%	0,00%	60,71%
NS_SCALE_VNF_VNF_IND	4	0	36	4	40	10,00%	100,00%	0,00%	90,00%
NS_SCALE_VNF_VIM_METRIC	8	0	38	8	46	17,39%	100,00%	0,00%	82,61%
NS_SCALE_NS_TO_LEVEL_MANUAL	14	0	13	14	27	51,85%	100,00%	0,00%	48,15%
NS_SCALE_VNF_TO_LEVEL_MANUAL	4	0	10	4	14	28,57%	100,00%	0,00%	71,43%
NS_SCALE_VNF_TO_LEVEL_VIM_METRIC	1	0	11	1	12	8,33%	100,00%	0,00%	91,67%
NS_UPDATE_VNF	31	6	35	37	72	51,39%	83,78%	16,22%	48,61%

NS_PM_JOB	33	2	75	35	110	31,82%	94,29%	5,71%	68,18%
NS_TERMINATE	45	0	0	45	45	100,00%	100,00%	0,00%	0,00%
NS_DELETE	86	2	2	88	90	97,78%	97,73%	2,27%	2,22%
NS_TERMINATE_SFC	1	0	1	1	2	50,00%	100,00%	0,00%	50,00%
TOTAL	448	11	367	459	826	55,57%	97,60%	2,40%	44,43%

Table 32. Results for NS Sub-Group

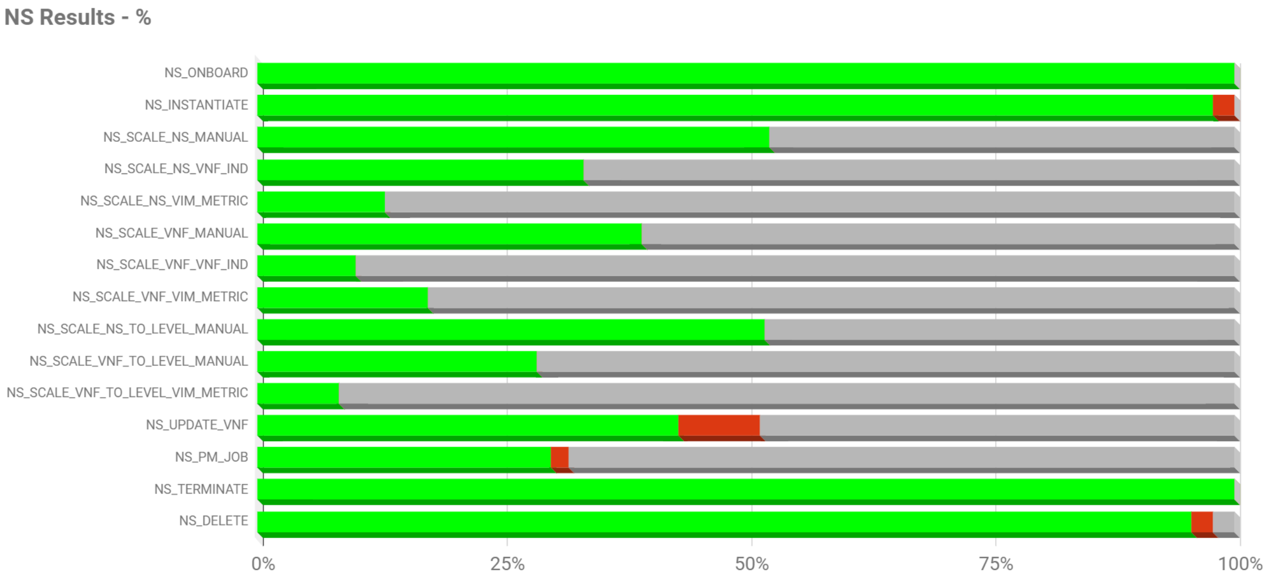

Figure 34. Results per NS Sub-Group - %

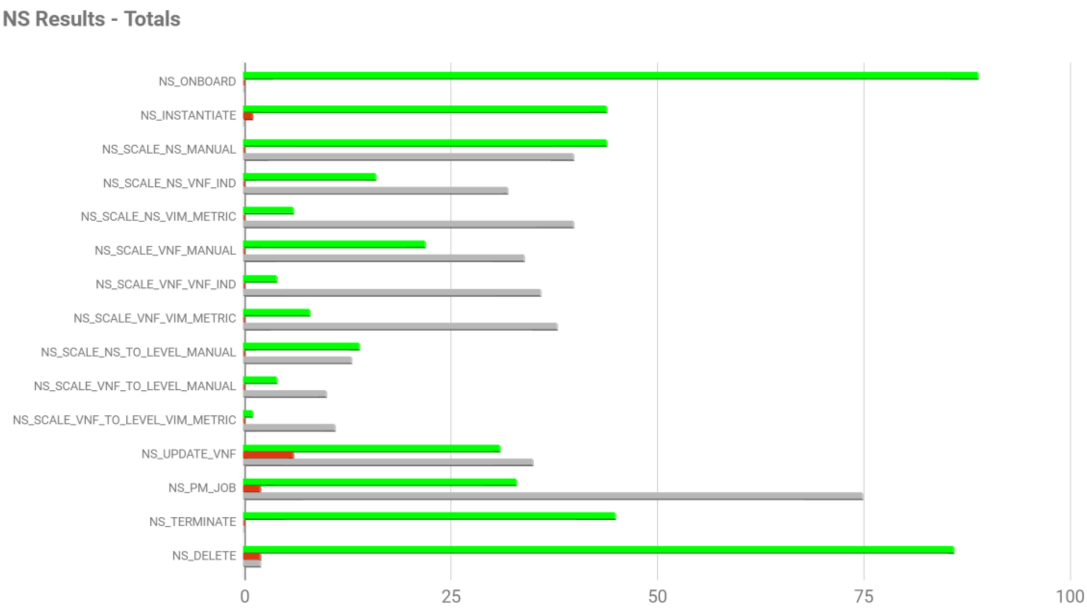

Figure 35. Results per NS Sub-Group - Totals

9.1.2.2 NS CNF

The table and figure below provide an overview of the results for the NS-CNF group. Overall, 12 NS-CNF test sessions were run.

	Interoperability		Not Executed	Totals		Totals			
	OK	NO	NA	Run	Results	% Run	% OK	% NO	% NA
NS-CNF_ONBOARD	21	2	0	23	23	100,00%	91,30%	8,70%	0,00%
NS-CNF_INSTANTIATE	9	3	0	12	12	100,00%	75,00%	25,00%	0,00%
NS-CNF_SCALE_NS_MANUAL	2	0	16	2	18	11,11%	100,00%	0,00%	88,89%
NS-CNF_SCALE_VNF_TO_LEVEL_MANUAL	1	0	5	1	6	16,67%	100,00%	0,00%	83,33%
NS-CNF_UPDATE_VNF	10	0	6	10	16	62,50%	100,00%	0,00%	37,50%
NS-CNF_NS_TERMINATE	10	0	1	10	11	90,91%	100,00%	0,00%	9,09%
NS-CNF_NS_DELETE	16	0	6	16	22	72,73%	100,00%	0,00%	27,27%
TOTAL	69	5	34	74	108	68,52%	93,24%	6,76%	31,48%

Table 33. Results per NS-CNF Sub-Group

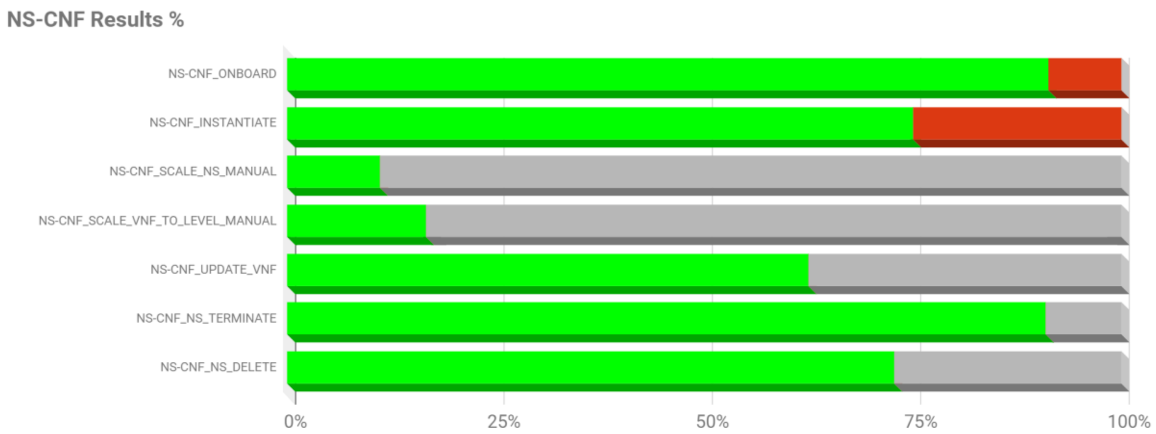

Figure 36. Results per NS-CNF Sub-Group - %

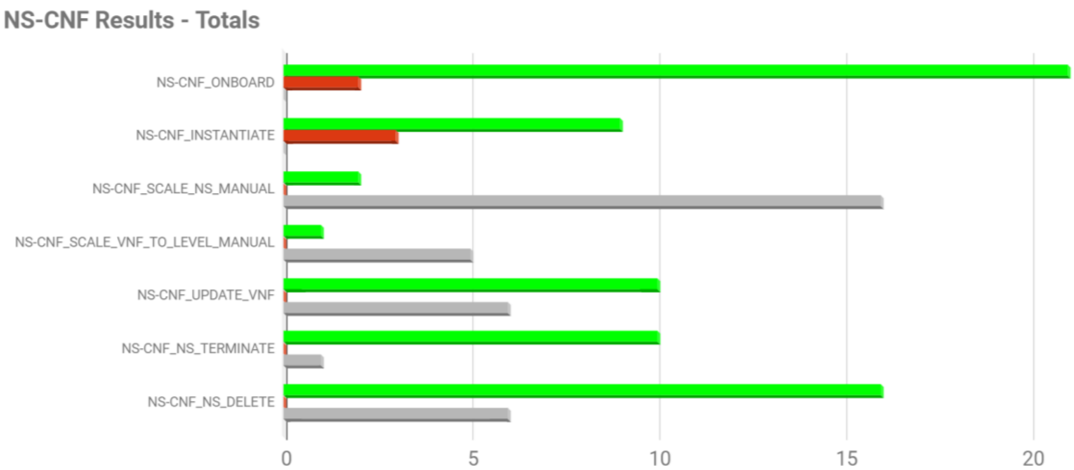

Figure 37. Results per NS-CNF Sub-Group - Totals

9.1.2.3 Specific VNFM

The table and figure below provide an overview of the results for the S-VNFM group. Overall, 18 Test Sessions involving different combinations of FUTs were run, and results reported as follows:

	Interoperability		Not Executed	Totals		Totals			
	OK	NO	NA	Run	Results	% Run	% OK	% NO	% NA
S-VNFM_ONBOARD	31	0	3	31	34	91,18%	100,00%	0,00%	8,82%
S-VNFM_INSTANTIATE	16	0	0	16	16	100,00%	100,00%	0,00%	0,00%
S-VNFM_SCALE_NS_MANUAL	0	2	26	2	28	7,14%	0,00%	100,00%	92,86%
S-VNFM_SCALE_NS_VNF_IND	0	4	16	4	20	20,00%	0,00%	100,00%	80,00%
S-VNFM_SCALE_NS_VIM_METRIC	0	2	14	2	16	12,50%	0,00%	100,00%	87,50%
S-VNFM_SCALE_VNF_MANUAL	8	2	18	10	28	35,71%	80,00%	20,00%	64,29%
S-VNFM_SCALE_VNF_VIM_METRIC	0	2	14	2	16	12,50%	0,00%	100,00%	87,50%
S-VNFM_SCALE_VNF_TO_LEVEL_MANUAL	0	1	1	1	2	50,00%	0,00%	100,00%	50,00%
S-VNFM_UPDATE_VNF	3	6	31	9	40	22,50%	33,33%	66,67%	77,50%
S-VNFM_FM_SUBSCRIPTION	4	0	0	4	4	100,00%	100,00%	0,00%	0,00%
S-VNFM_FM_ALARMS	4	2	0	6	6	100,00%	66,67%	33,33%	0,00%
S-VNFM_NS_TERMINATE	16	0	1	16	17	94,12%	100,00%	0,00%	5,88%
S-VNFM_NS_DELETE	29	0	5	29	34	85,29%	100,00%	0,00%	14,71%
TOTAL	111	21	129	132	261	50,57%	84,09%	15,91%	49,43%

Table 34. Results per Specific VNFM Sub-Group

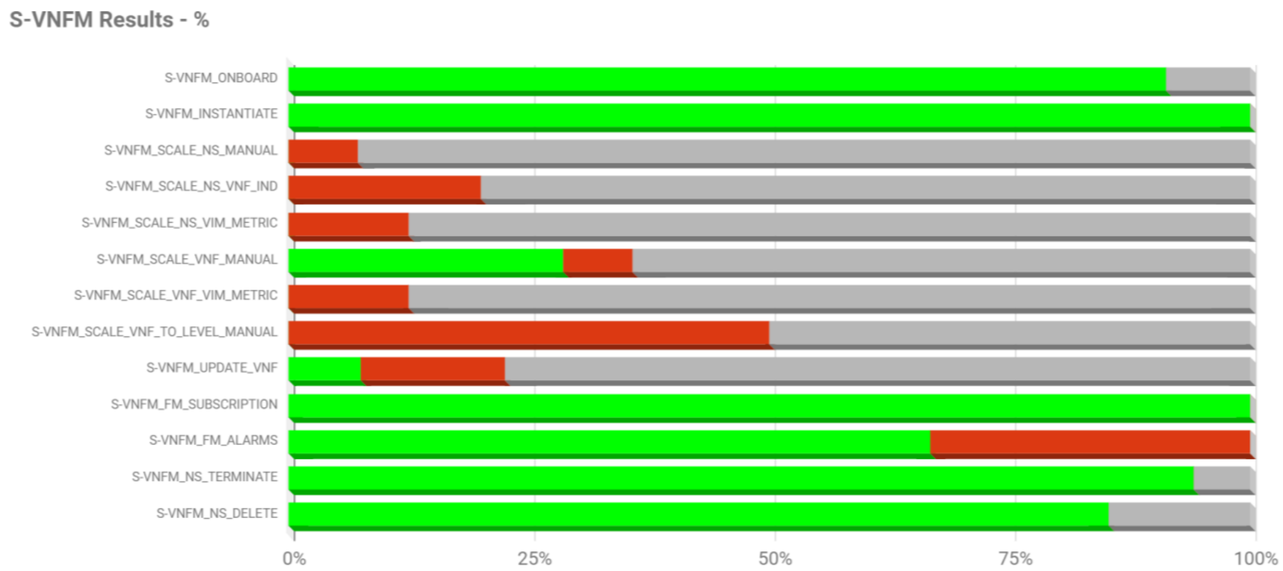

Figure 38. Results per Specific VNFM Sub-Group - %

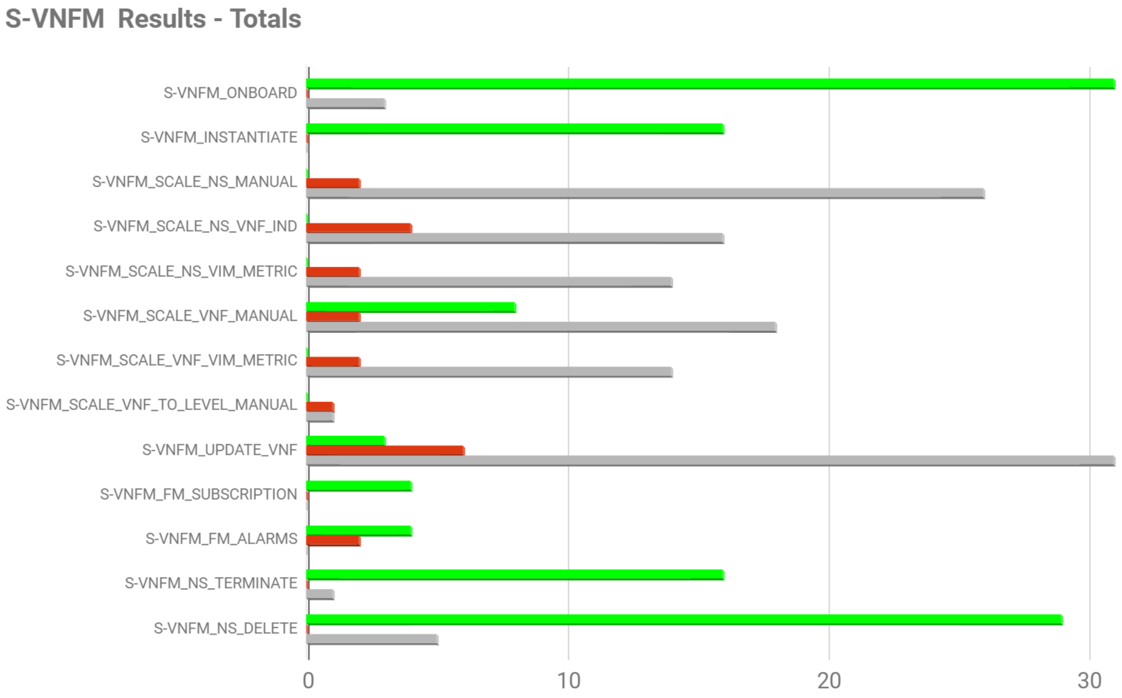

Figure 39. Results per S-VNFM Sub-Group - Totals

9.1.2.4 OSS/BSS

The table and figure below provide an overview of the results for the OSS/BSS group. Overall, 2 OSS/BSS test sessions were run.

	Interoperability		Not Executed	Totals		Totals			
	OK	NO	NA	Run	Results	% Run	% OK	% NO	% NA
NS_OSS-BSS_ONBOARD	4	0	0	4	4	100,00%	100,00%	0,00%	0,00%
NS_OSS-BSS_INSTANTIATE	2	0	0	2	2	100,00%	100,00%	0,00%	0,00%
NS_OSS-BSS_NS_TERMINATE	2	0	0	2	2	100,00%	100,00%	0,00%	0,00%
NS_OSS-BSS_NS_DELETE	4	0	0	4	4	100,00%	100,00%	0,00%	0,00%
TOTAL	12	0	0	12	12	100,00%	100,00%	0,00%	0,00%

Table 35. Results per OSS/BSS Sub-Groups

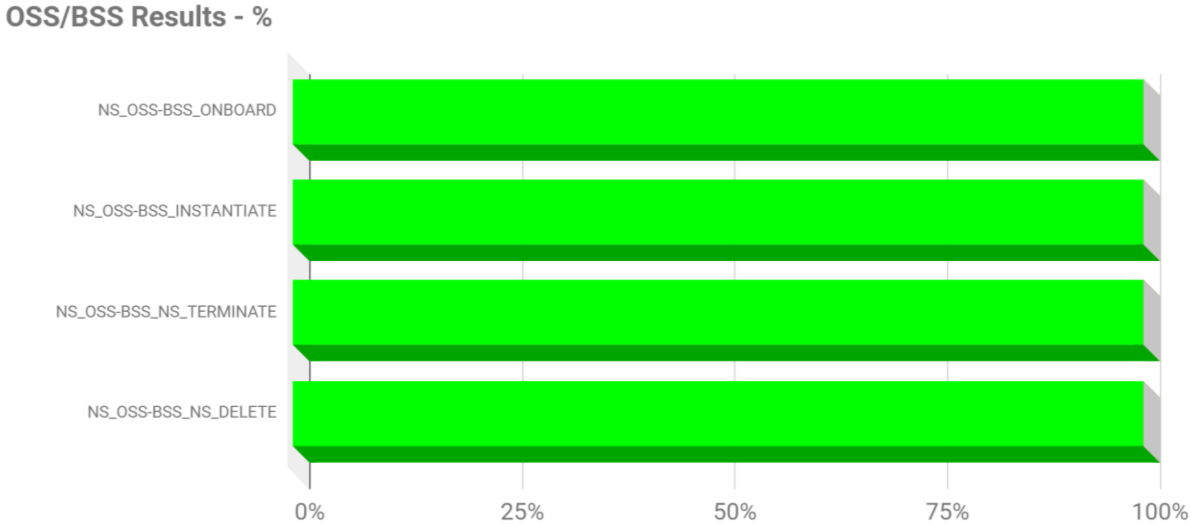

Figure 40. Results per OSS/BSS Sub-Group - %

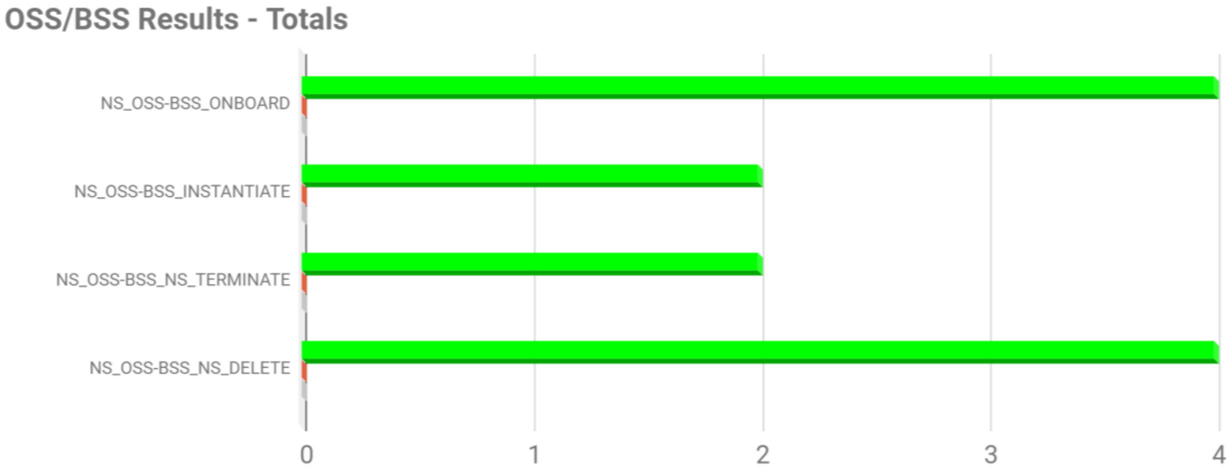

Figure 41. Results per OSS/BSS Sub-Group – Totals

9.1.2.5 Multi-Site

The table and figure below provide an overview of the results for the MULTI-SITE group. Overall, 2 Multi-SITE test sessions were run.

	Interoperability		Not Executed	Totals		Totals			
	OK	NO	NA	Run	Results	% Run	% OK	% NO	% NA
MULTISITE_ONBOARD	4	0	0	4	4	100,00%	100,00%	0,00%	0,00%
MULTISITE_INSTANTIATE	2	0	0	2	2	100,00%	100,00%	0,00%	0,00%
MULTISITE_NS_TERMINATE	2	0	0	2	2	100,00%	100,00%	0,00%	0,00%
MULTISITE_NS_DELETE	4	0	0	4	4	100,00%	100,00%	0,00%	0,00%
TOTAL	12	0	0	12	12	100,00%	100,00%	0,00%	0,00%

Table 36. Results per MULTI-SITE Sub-Groups

Figure 42. Results per MULTI-SITE Sub-Group - %

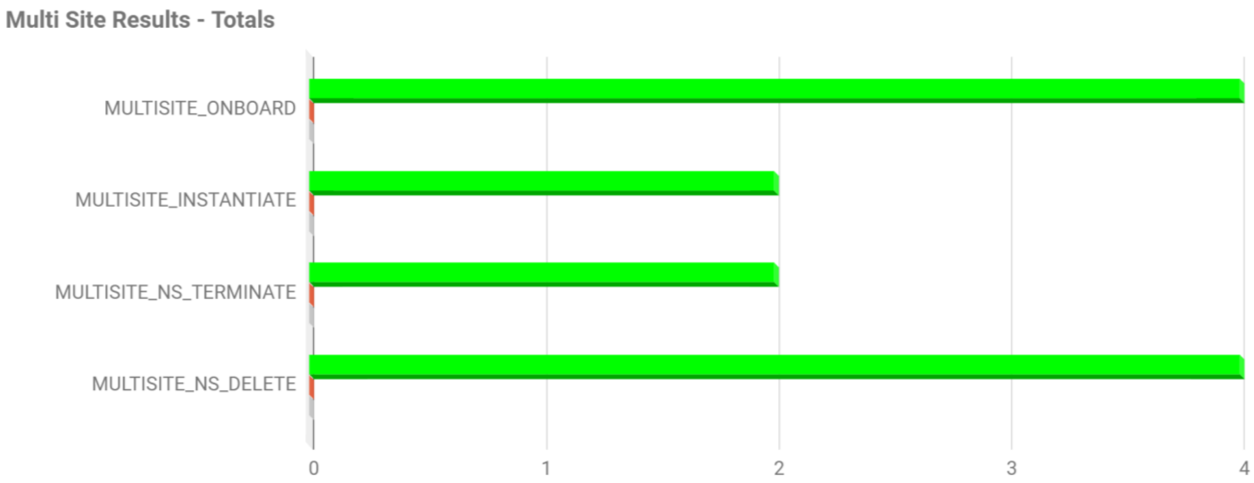

Figure 43. Results per MULTI-SITE Sub-Group – Totals

9.1.2.6 Multi-VNFM

The table and figure below provide an overview of the results for the MULTI-S-VNFM group. Overall, 2 Multi-S-VNFM test sessions were run.

	Interoperability		Not Executed	Totals		Totals			
	OK	NO	NA	Run	Results	% Run	% OK	% NO	% NA
MULTI_S-VNFM_ONBOARD	4	0	0	4	4	100,00%	100,00%	0,00%	0,00%
MULTI_S-VNFM_INSTANTIATE	2	0	0	2	2	100,00%	100,00%	0,00%	0,00%
MULTI_S-VNFM_SCALE_VNF_MANUAL	2	0	2	2	4	50,00%	100,00%	0,00%	50,00%
MULTI_S-VNFM_TERMINATE	1	0	1	1	2	50,00%	100,00%	0,00%	50,00%
MULTI_S-VNFM_DELETE	2	0	2	2	4	50,00%	100,00%	0,00%	50,00%
TOTAL	11	0	5	11	16	68,75%	100,00%	0,00%	31,25%

Table 37. Results per MULTI-S-VNFM Sub-Groups

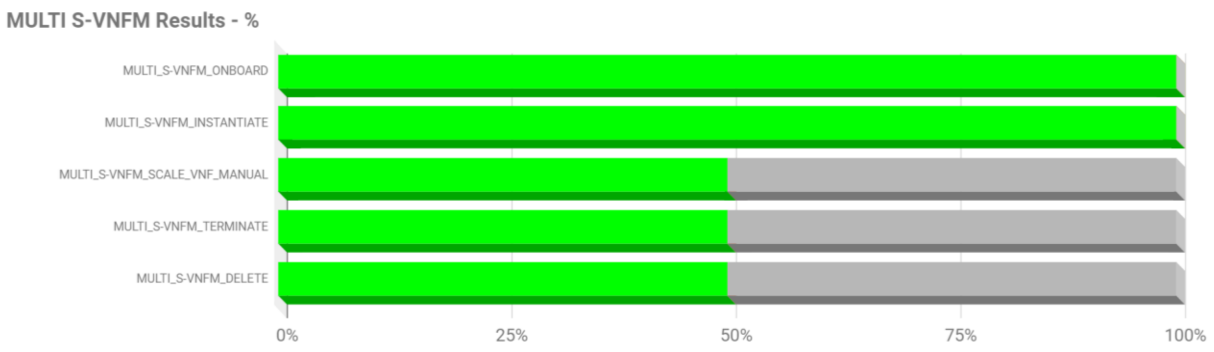

Figure 44. Results per MULTI-S-VNFM Sub-Group - %

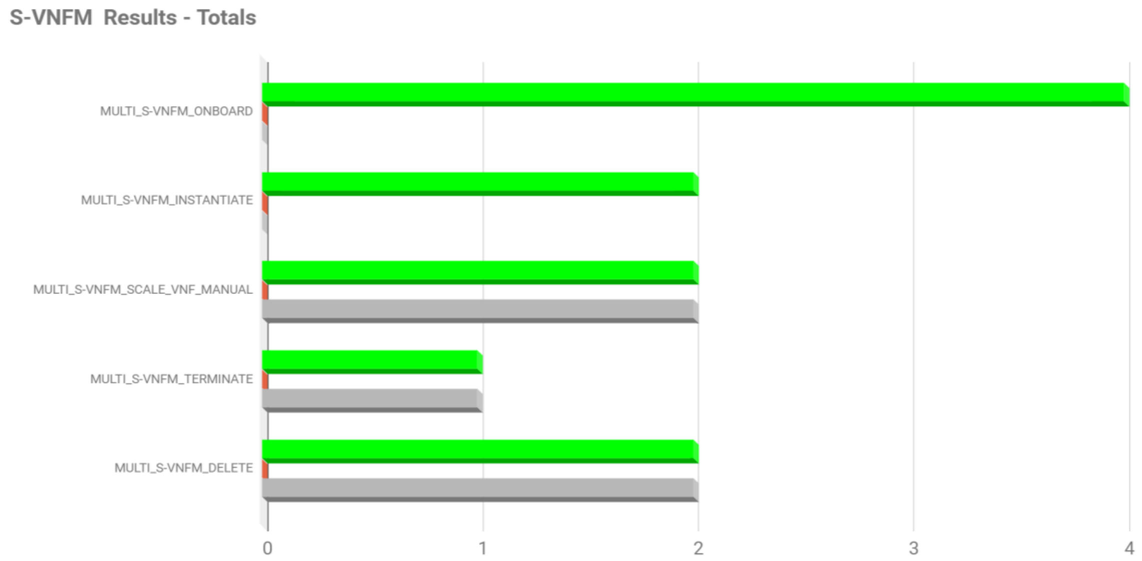

Figure 45. Results per MULTI-S-VNFM Sub-Group – Totals

9.1.2.7 Auto-IOP

The table and figure below provide an overview of the results for the AUTO-IOP group. Overall, 7 AUTO-IOP test sessions were run.

	Interoperability		Not Executed	Totals		Totals			
	OK	NO	NA	Run	Results	% Run	% OK	% NO	% NA
AUTO_IOP_ONBOARD	11	2	0	13	13	100,00%	84,62%	15,38%	0,00%
AUTO_IOP_INSTANTIATE	5	1	0	6	6	100,00%	83,33%	16,67%	0,00%
AUTO_IOP_SCALE_VNF_MANUAL	5	2	5	7	12	58,33%	71,43%	28,57%	41,67%
AUTO_IOP_PM_JOB	0	2	23	2	25	8,00%	0,00%	100,00%	92,00%
AUTO_IOP_NS_TERMINATE	6	1	0	7	7	100,00%	85,71%	14,29%	0,00%
AUTO_IOP_NS_DELETE	9	3	0	12	12	100,00%	75,00%	25,00%	0,00%
TOTAL	36	11	28	47	75	62,67%	76,60%	23,40%	37,33%

Table 38. Results per AUTO-IOP Sub-Groups

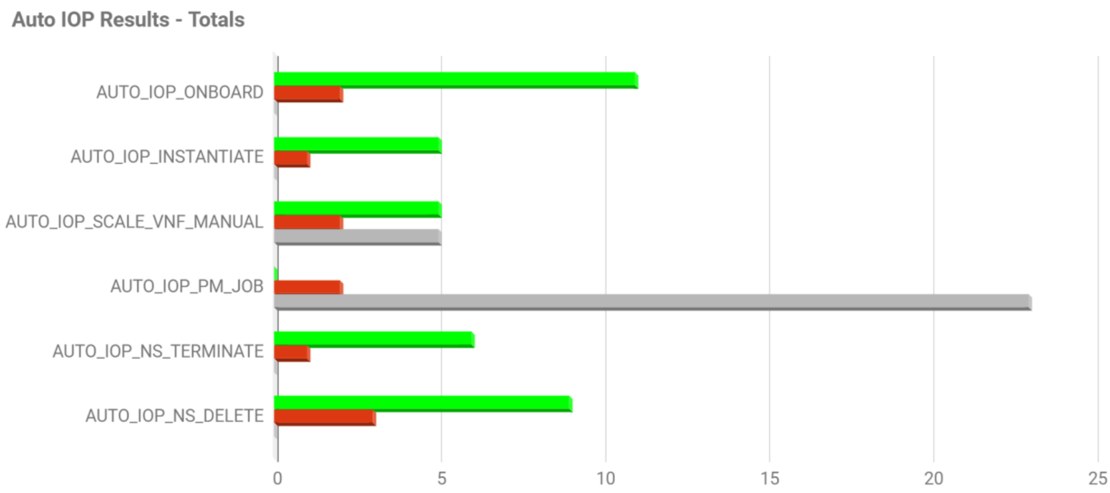

Figure 46. Results per AUTO-IOP Sub-Group - %

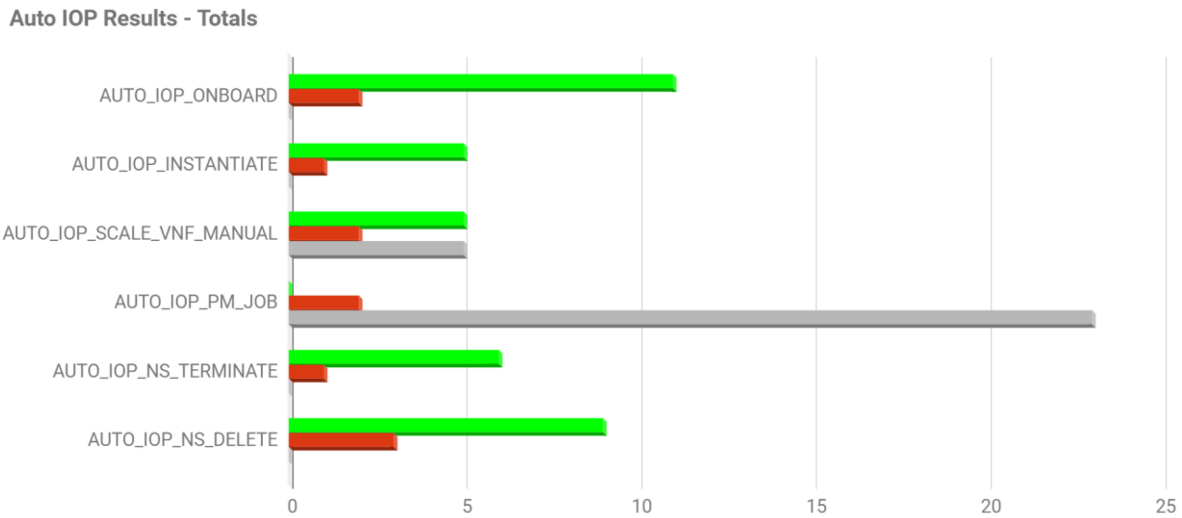

Figure 47. Results per AUTO-IOP Sub-Group – Totals

9.1.2.8 Auto-IOP S-VNFM

The table and figure below provide an overview of the results for the AUTO-IOP S-VNFM group. Overall, 1 AUTO-IOP S-VNFM test sessions were run.

	Interoperability		Not Executed	Totals		Totals			
	OK	NO	NA	Run	Results	% Run	% OK	% NO	% NA
AUTO_IOP_S-VNFM_INSTANTIATE	0	1	0	1	1	100,00%	0,00%	100,00%	0,00%
AUTO_IOP_S-VNFM_SCALE_NS_MANUAL	0	0	2	0	2	0,00%	0,00%	0,00%	100,00%
AUTO_IOP_S-VNFM_NS_TERMINATE	1	0	0	1	1	100,00%	100,00%	0,00%	0,00%
AUTO_IOP_S-VNFM_NS_DELETE	2	0	0	2	2	100,00%	100,00%	0,00%	0,00%
TOTAL	3	1	2	4	6	66,67%	75,00%	25,00%	33,33%

Table 39. Results per AUTO-IOP S-VNFM Sub-Groups

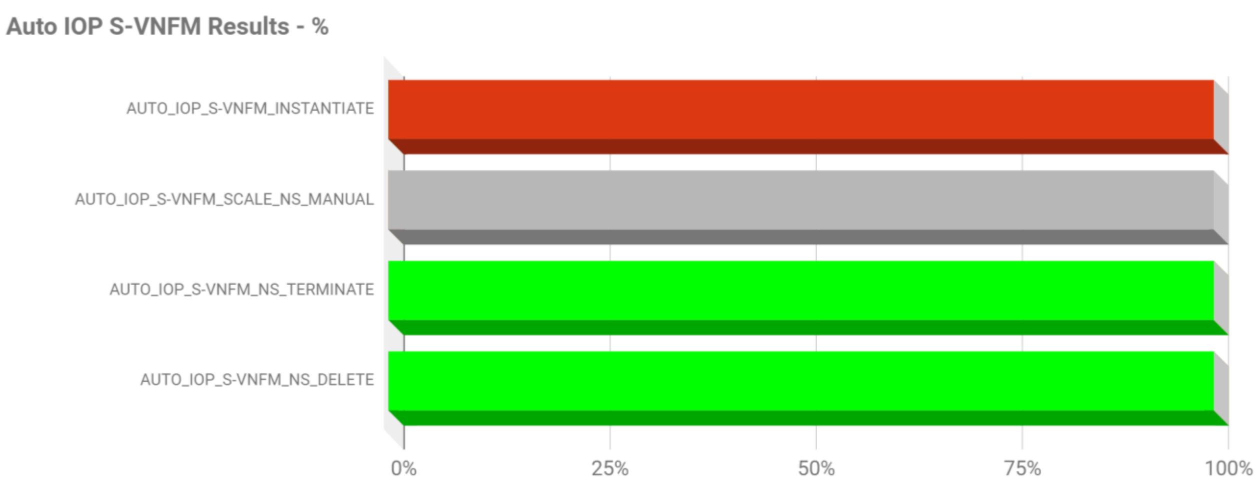

Figure 48. Results per AUTO-IOP S-VNFM Sub-Group - %

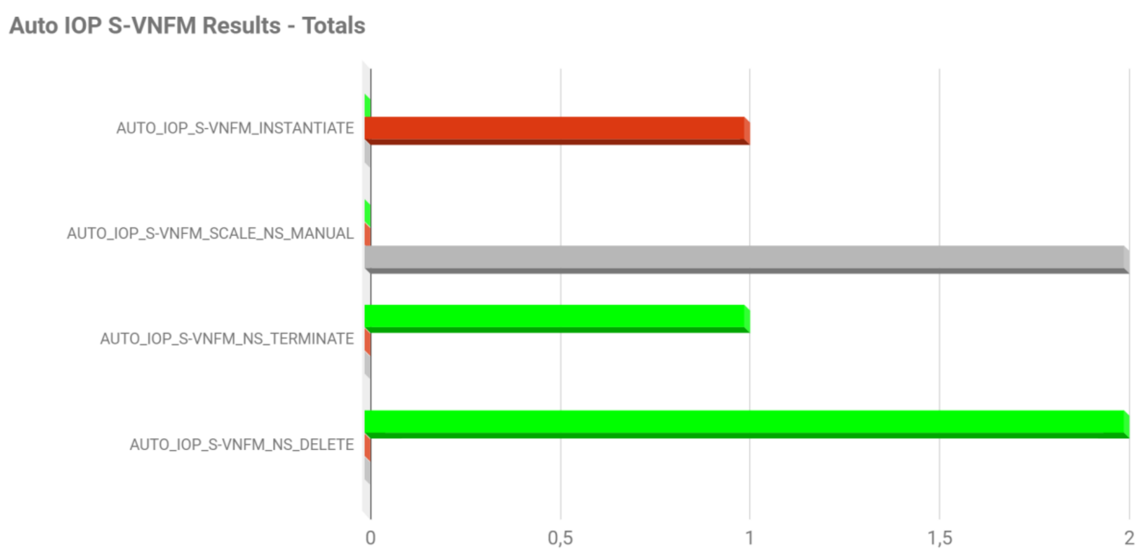

Figure 49. Results per AUTO-IOP S-VNFM Sub-Group – Totals

9.1.3 Results per Test Case

The full list of NFV interoperability results per Test Case is provided in [NFV-IOP-TR].

9.2 NFV API Conformance Results

During the NFV&MEC Plugtests 2020 event, several NFV API Conformance Test Sessions were run, as described in an automated and on-demand fashion. Out of all executed test sessions, all executed tests have been reported and for each of the tests the best result has been used in the final calculation of the outcomes reported below. The API conformance tests were executed for two different Functions Under Test (FUTs): VNFMs and NFVOs. A total of 8 FUTs participated to the NFV API Conformance sessions, that is:

- 3 VNFMs
- 5 NFVOs

The table below provides the overall results (aggregated data) for all the NFV API Conformance tests run during the NFV&MEC Plugtests 2020, from all participating organisations.

Overall Results	API Conformance (TCs Run)		TCs Totals
	OK	NO	Total
	531	638	1169

Table 40: NFV API Conformance overall results

For each remote Test Session, depending on the involved FUT and the features to be tested, the involved participant was able to select different number of test cases.

Overall, the test plan included more than 2000 NFV API Conformance test cases, organised in different groups as described in clause 8.2. The test plan was based on ETSI GS TST010 [TST010], in both versions 2.4.1 and 2.6.1. Participants were free to select the version of the test suite according to their implementations.

With respect to the previous NFV Plugtests event, a larger number of tests was made available. Through the Test Sessions run, a total of 1169 Test Results were executed and reported.

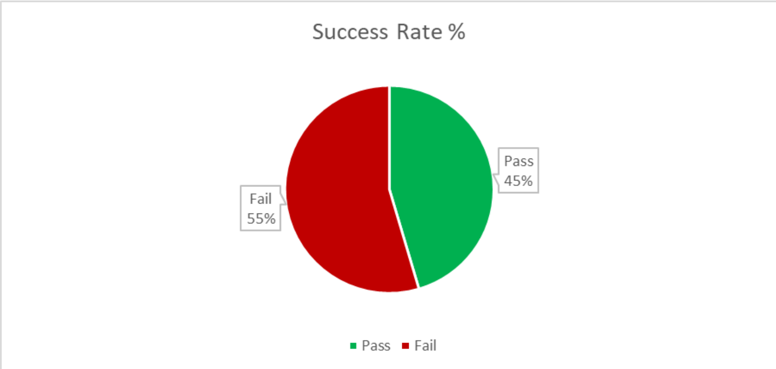

Figure 50. NFVAPI Conformance Overall results (%)

The next clauses present more detailed results per SOL Specification (and for each version of), per FUT type, per SUT configuration and per test group and will allow to identify the areas and APIs with higher execution and conformance rates.

9.2.1 Results per SOL Specification

The tables and figures below provide an overview of the results for the API conformance per SOL specifications, i.e. SOL003 and SOL005. Overall, the SOL005 APIs have been those with the higher number of Test Cases run while SOL003 had the highest success rate.

	API Conformance		Totals	Totals	
	OK	NO	Results	% OK	% NO
SOL003	239	513	752	47%	53%
SOL005	292	656	948	45%	55%
TOTAL	531	638	1169	45%	55%

Table 41: Test Results summary per-SOL Specification

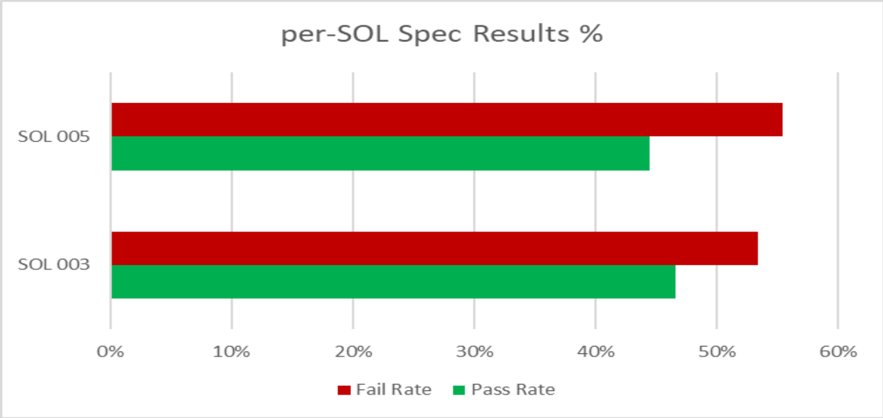

Figure 51. Test results per-SOL Specification - %

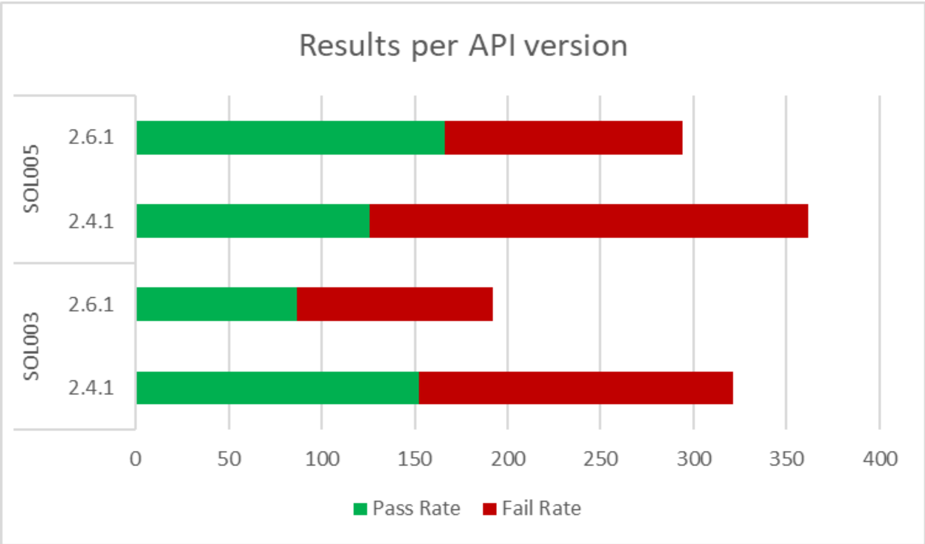

Figure 52. Test results per version

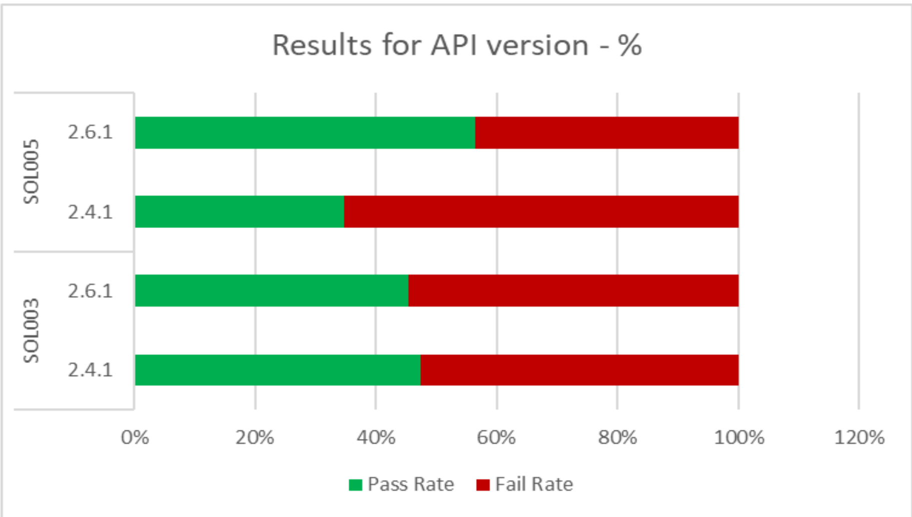

Figure 53. Test results per version - %

9.2.2 Results per FUT Type

The tables and figures below summarize the results for the API conformance tests per type of FUT involved in the API conformance test sessions, i.e. VNFM and NFVO. NFVO FUTs were those most tested.

	API Conformance		Totals	Totals	
	OK	NO	Run	% OK	% NO
VNFM	153	157	310	49%	51%
NFVO	378	481	859	55%	56%
TOTAL	531	638	1169	45%	55%

Table 42: Test Results summary per-FUT type

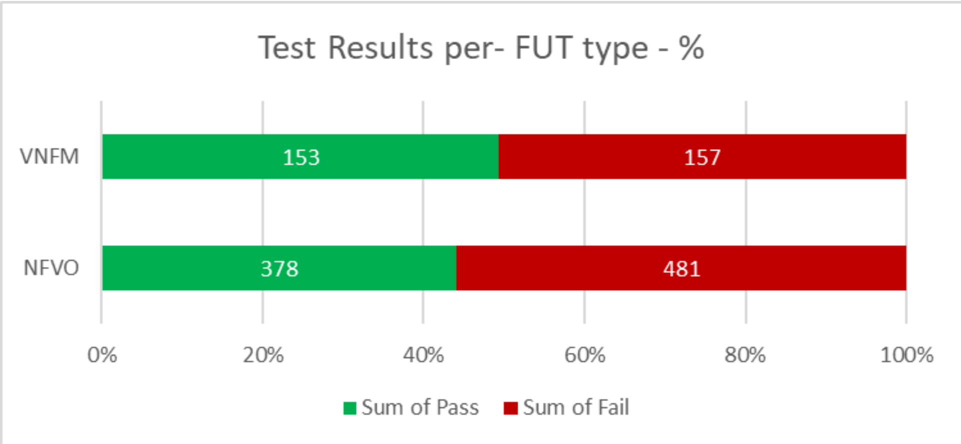

Figure 54. Test Results per- FUT type - %

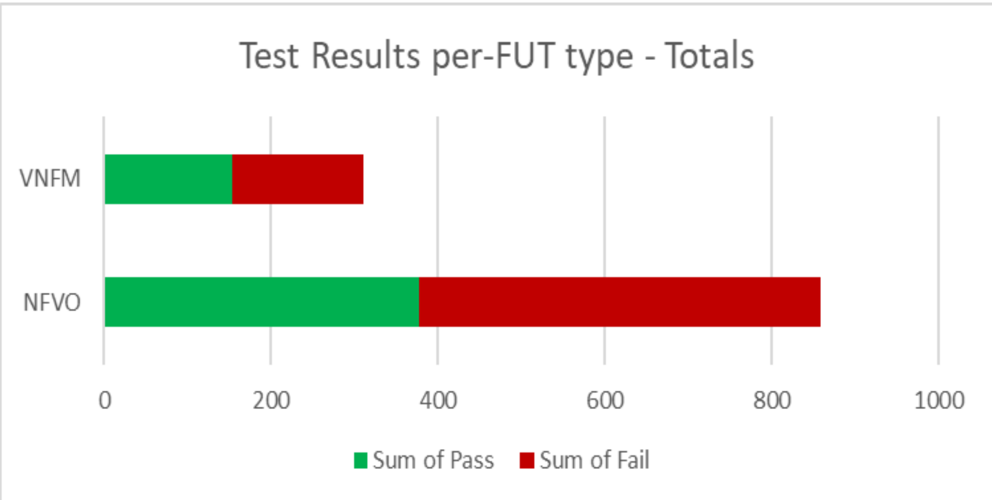

Figure 55. Test Results per-FUT type - Totals

9.2.3 Results per API

The following clauses provide tables and figures which summarize the results for the NFV API conformance tests for the different APIs in each test configuration.

9.2.3.1 VNFM - SOL003

	API Conformance		Totals	Totals	
	OK	NO	Run	% OK	% NO
VNF Lifecycle Management API	152	157	309	49%	51%
VNF Fault Management API	1	0	1	100,00%	0,00%
TOTAL	153	157	310	49%	51%

Table 43: SOL003 VNFM test results summary

9.2.3.4 NFVO - SOL003

	API Conformance		Totals	Totals	
	OK	NO	Run	% OK	% NO
VNF Package Management API	83	117	200	42%	58%
VNF Lifecycle Operation Granting API	3	0	3	100%	0%
TOTAL	86	117	203	42%	58%

Table 44: NFV SOL003 test results summary

9.2.3.5 NFVO - SOL005

	API Conformance		Totals	Totals	
	OK	NO	Run	% OK	% NO
NSD Management API	88	78	166	53%	47%
NS Lifecycle Management API	148	140	288	51%	49%
VNF Package Management API	9	45	54	17%	83%
NS Performance Management API	47	101	148	32%	68%
TOTAL	292	364	656	45%	55%

Table 45: NFVO SOL005 test results summary

9.2.4 Results per Test Case

The full list of NFV API Conformance results per Test Case is provided in [NFV-API-TR].

9.3 MEC Interoperability

9.3.1 Overall Results

As part of the MEC Track a total of 7 Test Sessions were run, combining 5 different Functions Under Test. The MEC interoperability tests were executed for 2 different configurations, one including a MEC App and a MEC Platform (running on its own NFVI) and a second one with a MEC App and a MEC Platform running on an NFVI from a different provider. The subsequent sections report the aggregated testing results.

Overall Results	Number of Test Sessions	Interoperability (TCs Run)		TCs Not Run	TCs Totals	
		OK	NO	NA	Run	Total
	7	58	21	21	79	100

Table 46: MEC IOP Overall Results

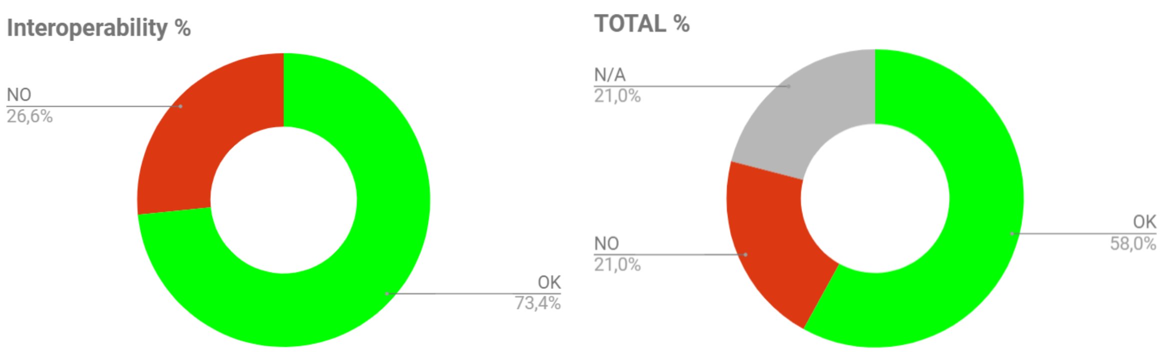

Figure 56. MEC IOP Overall results (%)

The table below provides an overview of the results for each group in the MEC Track.

	Interoperability		Not Executed	Totals		Totals (%)			
	OK	NO	N/A	Run	Results	% Run	% OK	% NO	% N/A
MEC_BASIC	22	13	4	35	39	89,74%	62,86%	37,14%	10,26%
MEC_SERVICES_SINGLE_APP	5	0	7	5	12	41,67%	100,00%	0,00%	58,33%
MEC_SERVICES_MULTI_APP	31	8	10	39	49	79,59%	79,49%	20,51%	20,41%
MEC_NFVI	0	0	0	0	0	0,00%	0,00%	0,00%	0,00%
MEC_NFVI_MANO	0	0	0	0	0	0,00%	0,00%	0,00%	0,00%

Table 47: MEC IOP Overall results per group

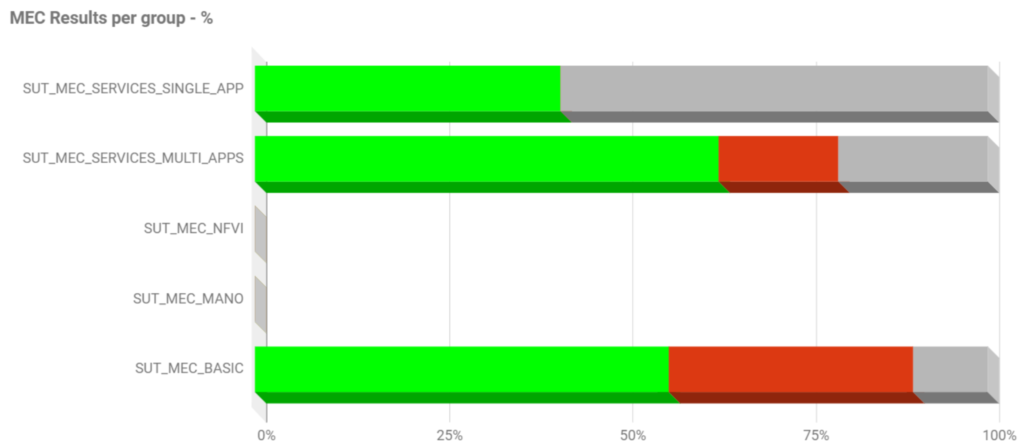

Figure 57. MEC IOP results per group - %

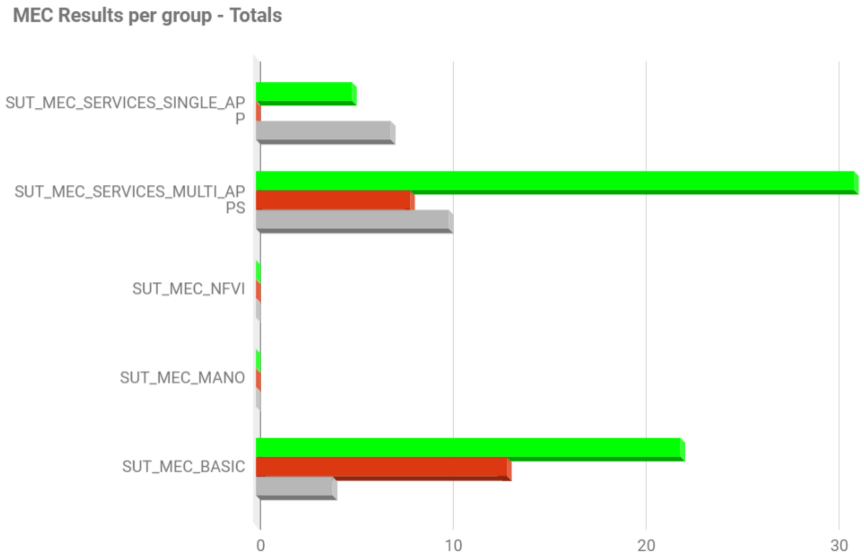

Figure 58. MEC IOP results per group - Totals

9.3.2 Results per Test Case

The Table below provides an overview of the results for each test descriptions in the MEC Track.

	Interoperability		Not Executed		Totals	
	OK	NO	NA	OT	Run	Results
TD_MEC_NTW_ACTIVATE	0	4	2	0	4	6
TD_MEC_NTW_UPDATE	0	4	2	0	4	6
TD_MEC_NTW_DEACTIVATE	0	4	2	0	4	6
TD_MEC_NTW_DNS_ACTIVATE	3	1	2	0	4	6
TD_MEC_NTW_DNS_DEACTIVATE	3	1	2	0	4	6
TD_MEC_SVC_QUERY	5	1	0	0	6	6
TD_MEC_SVC_REGISTER	4	1	0	0	5	5
TD_MEC_SVC_UPDATE	0	4	1	0	4	5
TD_MEC_SVC_DEREGISTER	4	1	0	0	5	5
TD_MEC_SVC_CONSUMPTION	2	0	2	0	2	4
TD_MEC_APP_ONBOARD	8	0	0	0	8	8
TD_MEC_APP_START	8	0	0	0	8	8
TD_MEC_APP_STOP	8	0	0	0	8	8
TD_MEC_APP_STATUS	8	0	0	0	8	8
TD_MEC_APP_CHANGE	0	0	8	0	0	8
TD_MEC_SVC_TRANSPORTS_QUERY	3	0	0	0	3	3
TD_MEC_SVC_TIMEQUERY	2	0	0	0	2	2

Table 48: MEC IOP results per TD

9.4 MEC API Conformance Results

During the NFV&MEC Plugtests 2020 event, several MEC API Conformance Test Sessions were run, as described, in an automated and on-demand fashion. Out of all executed test sessions, all executed tests have been reported and for each of the tests the best result has been used in the final calculation of the outcomes reported below. The API conformance tests were executed for Functions Under Test (FUTs) of type “MEC Platform”. A total of 2 MEC Platforms participated to the MEC API Conformance sessions.

The table below provides the overall results (aggregated data) for all the NFV API Conformance tests run during the NFV&MEC Plugtests 2020, from all participating organisations.

Overall Results	API Conformance (TCs Run)		TCs Totals
	OK	NO	Total
	147	52	199

Table 49: NFV API Conformance overall results

For each remote Test Session, depending on the involved FUT and the features to be tested, the involved participant was able to select different number of test cases.

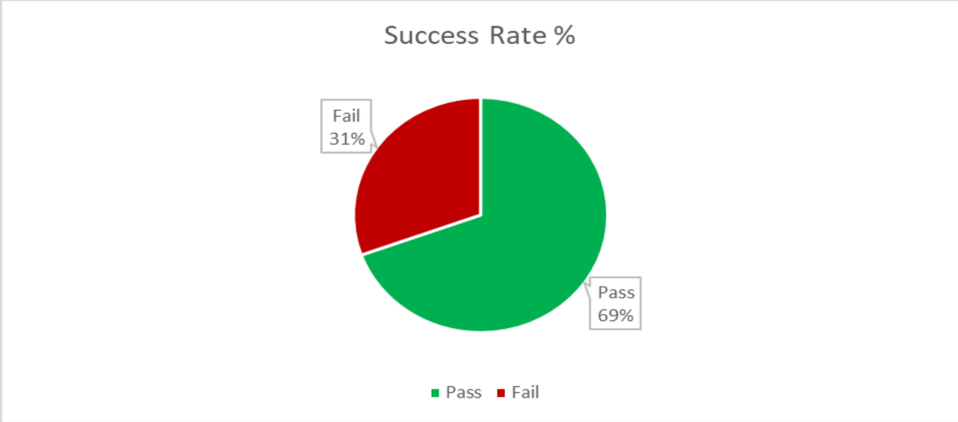

Figure 57. MEC API Conformance Overall results (%)

The next clauses present more detailed results per MEC Specification and per test group and will allow to identify the areas and APIs with higher execution and conformance rates.

9.2.1 Results per MEC Specification

The tables and figures below provide an overview of the results for the API conformance per MEC specifications, i.e. MEC011, MEC012 and MEC013. Overall, the MEC011 APIs have been those with the higher number of Test Cases run and had the highest success rate.

	API Conformance		Totals	Totals	
	OK	NO	Results	% OK	% NO
MEC011	114	21	135	84%	16%
MEC012	1	15	16	6%	94%
MEC013	32	16	48	67%	33%
TOTAL	147	52	199	69%	31%

Table 50: Test Results summary per-MEC Specification

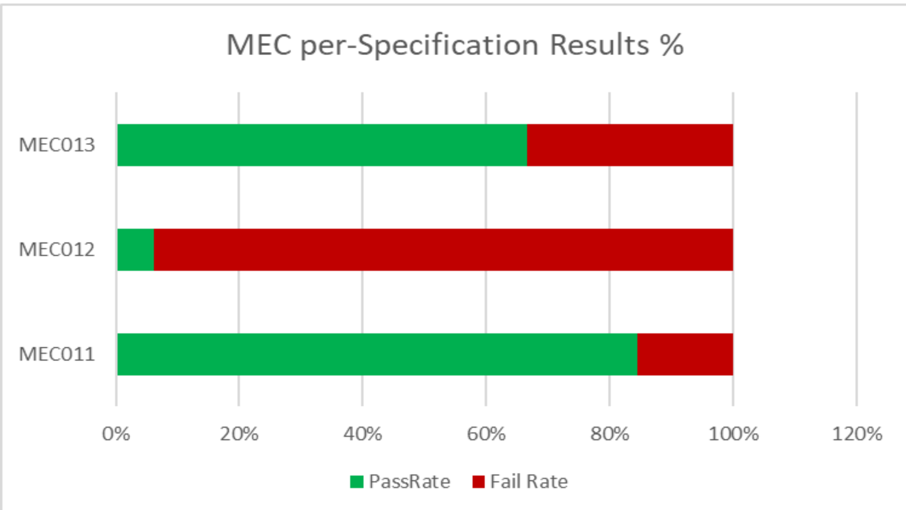

Figure 58. Test results per-MEC Specification - %

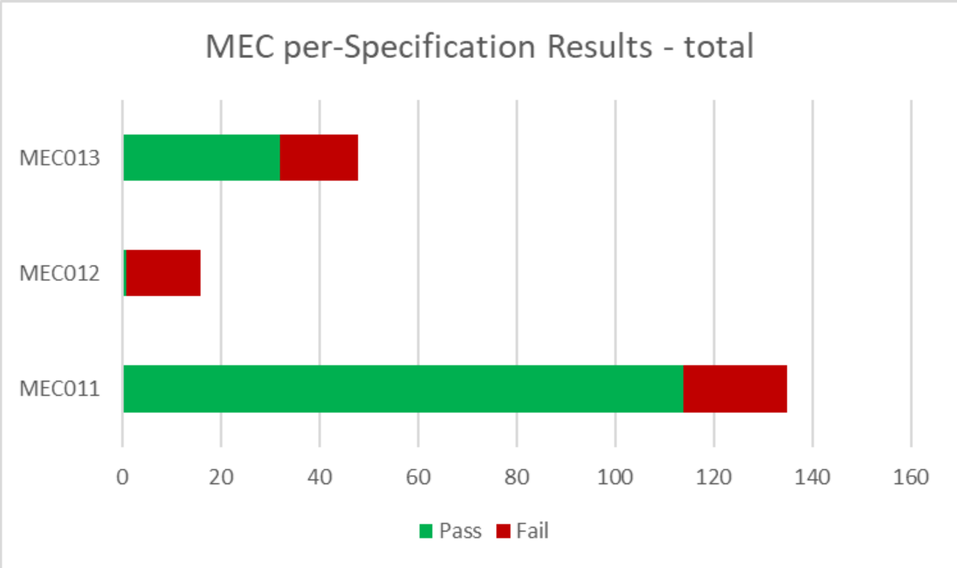

Figure 59. Test results MEC Specification – total

9.2.2 Results per test groups

The present clause reports the results of the MEC API test sessions focussing on the test groups. For MEC 012, only one test group (RNIS) was tested, therefore it is omitted in the present clause.

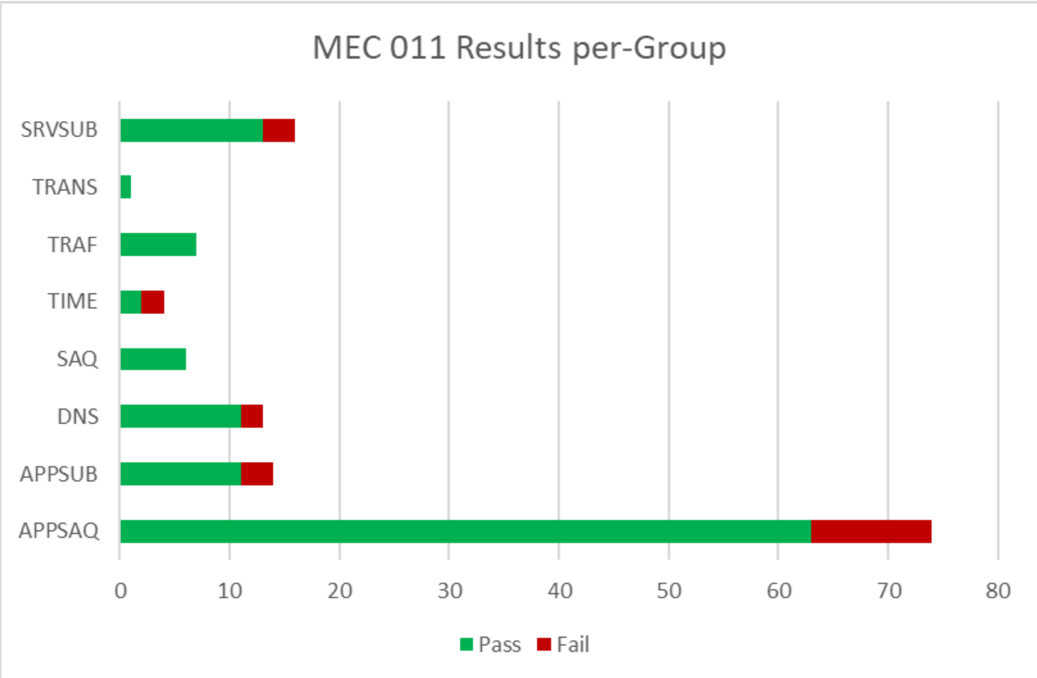

Figure 60. MEC 011 Test Results per group

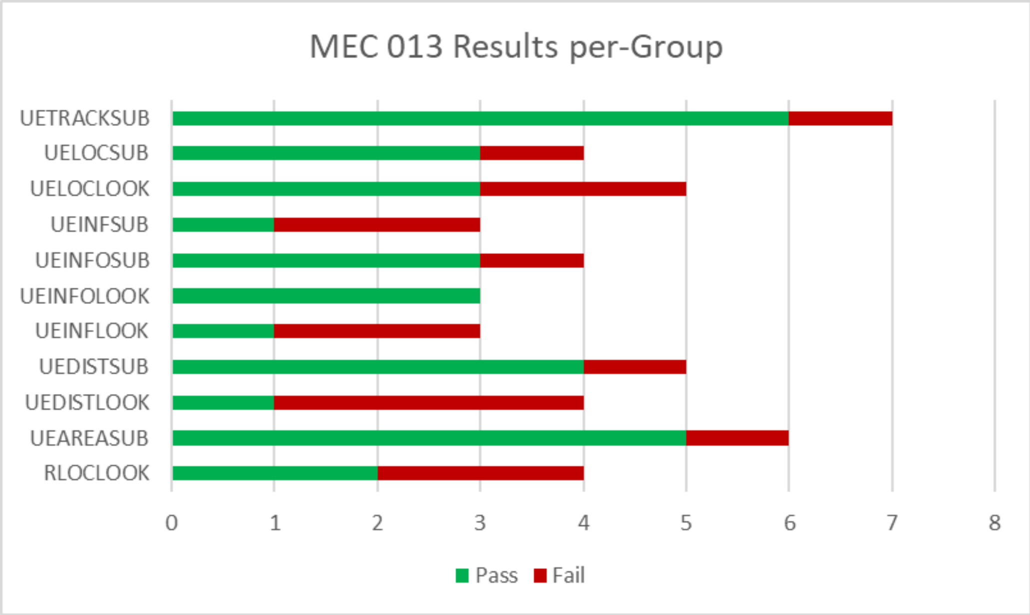

Figure 61. MEC 013 Test results per-Group

10 Plugtests Outcome

As for previous event, the NFV&MEC Plugtests 2020 allowed to identify a number of potential issues or inconsistencies in NFV&MEC specifications and triggered interesting discussions among participants, organisers and some experts from NFV and MEC ISGs. This chapter compiles the highlights of these discussions, the list of reported bugs and some recommendations on NFV and MEC specifications, Open APIs, and the test plans. This feedback is submitted to NFV and MEC ISGs for further discussion and eventual resolution in the appropriate working groups.

10.1 Feedback on Base Specifications

10.1.1 NFV Specifications

10.1.1.1 Experimental testing of CNFs

MANO solutions using helm or juju clients for instantiating CNFs on top of Kubernetes (i.e ETSI OSM) reported the necessity of specific requirements for instantiation to work, like applying “per-namespace” secrets for adding credentials for the private registry, or for ceph backends, etc, and since namespace names in that case are dynamic, that secrets needed to be applied after instantiation, which implies the automation of “per instantiation” secrets.

Furthermore, it was also noted that many Kubernetes clusters in the Plugtests were not delivering reachable IP addresses to the CNF requests of external IPs, which made Day-0 management not possible.

In conclusion, instantiation often required some manual operations, and as such test cases were marked as “failed” even if instantiation succeed after manual operations.

It was outlined the need to define a common way to automate those manual steps, so that CNF instantiation can be achieved for any combination of CNF, MANO and Kubernetes platform.

These concepts are currently being specified by ISG NFV in stage 2 (requirements) within the scope of NFV Rel-4 FEAT-17 “Cloud Native VNFs & Container Infrastructure Management”, FEAT-17 related information can be found at [NFV-FEAT17-WIKI]

The main contact for FEAT-17 was invited to join the discussion and provided the following clarifications:

The general concept of specifying instance specific parameter values during instantiation is based on values.yaml files being injected to the Helm create command. For NFV this is assumed to be part of new functions for the VNFM which will call the Helm API. Corresponding enhancements of the functional requirements for the VNFM and NFVO will be specified in the NFV Rel-4 version of [IFA010], including functional requirements for the new functions Container Infrastructure Service Manager (CISM, possibly within MANO) and Container Image Registry (CIR).

Properties of the CNFs which are relevant to be exposed to the NFV-MANO will be specified in enhancements of the VNFD and included in the NFV Rel-4 version of [IFA011].

Concerning namespaces, the current assumption is that the NFVO will be responsible for maintaining the container cluster namespaces. The CISM will expose the service interfaces for namespace management, to be consumed by the NFVO. This will include the namespace quota management. The idea is that as the namespaces provide some isolation for resources, which needs to be specified by the CSP. This also means that the namespace value cannot be specified by the VNF vendor in the descriptors (i.e. VNFD and Helm charts). Additionally, the current idea to convey the namespace value from NFVO to the VNFM within the grant response message. Finally, the namespace value is intended to be used for deploying a Helm chart. All K8s objects described within a Helm will be created within the same namespace. The namespace value can be specified as input parameter to the Helm API. The concepts in NFV only consider the Helm version 3 architecture where Helm consumes the native K8s APIs.

Concerning the secrets and credentials for the applications, these are application specific, and as such out of the scope of NFV.

Concerning the external connectivity of the CNFs, a Kubernetes cluster is expected to be configured to provide external connectivity. In principle, the Helm charts for the CNF need to contain manifests for K8s objects to specify the external connectivity/exposure (such as K8s Service and/or K8s Ingress). The POD objects and their containers (such as K8s

Deployment or StatefulSet) will only be connected to the cluster internal network with ephemeral addresses which are not exposed externally.

NFV ISG is currently working on modelling the container networking aspects both for the VNFD and the Or-Vnfm reference point. One of the current approaches is that the NFVO will be able to convey the IP address value for a K8s Service of type IP load balancer to the VNFM. The VNFM would then be able to inject this value (via a values.yaml file) to the Helm Create operation and overwrite a potential default value in the manifests.

10.1.1.2 SOL002

During the final wrap-up, some participants expressed their concern about SOL002 not being rich enough to support complex operations among MANO and VNFs, with the consequence of some MANOs using their own methods to complete the range of operations. A consolidation and eventual standardization of these additional operations could be beneficial for VNF – MANO interoperability. Following the Plugtests, this topic generated some discussion on the Plugtests mailing list, concretely the OSM VNF On-boarding Task Force shared the following concerns:

SOL002 VNF Configuration Interface proposes a single REST "patch" call to change the configuration of a VNFC that some VNF vendors might be reluctant to implement it because:

- They prefer to offer their customers other mechanisms that provide a better control of the configuration lifecycle, like NETCONF (which has useful operations like commit, confirm, rollback, etc), existing REST APIs with a richer structure, and even Ansible modules (which implement idempotency and parallelism on top of a CLI or existing REST API)
- The call is optional, which added to the above, decreases the motivation for implementing it.

As a result, and given that Day-1 & 2 configuration lifecycle management is, for many, one of the most important values of NFV Orchestration, we see both orchestrator and VNF providers implementing configuration management based on NETCONF, Ansible, richer REST APIs, or even direct CLI, instead of implementing a simplistic and optional standard.

It might be suggested that the "optional" nature of SOL002 VNF Configuration deals with this issue, giving vendors freedom to implement it as they prefer (assuming more generic mechanisms can somehow comply with IFA008 input parameters), however, this might as well be demotivating the implementation and testing of configuration management at all, and as a consequence, NFV Orchestration adoption all together, if we agree that Day 1 & 2 operations automation is one of its biggest values. A proof of this is that there has been very little configuration management testing with IFA008/SOL002 during this (and previous) Plugtests, while at the same time we see a lot of demand for Day-1 & 2 automation in NFV Orchestration POCs and implementations, sometimes being the most important topic.

A possible way forward could be to expand IFA008 / SOL002 VNF Configuration Interface to cover VNF Day-1 & 2 operations with other well-known and widely used configuration management mechanisms **and** declaring it mandatory, so VNFMs and VNFs can be motivated to implement it and achieve conformance without having to develop a new specific REST API call.

This would also motivate more configuration management related testing and highlight the value of NFV orchestration with regards to **VNF operations**, thus increasing its general adoption.

The above feedback was processed by the NFV SOL WG Chair, who while acknowledging the limitation of the scope of SOL002/IFA008 to virtualisation aspects reminded that the current NFV specs do not proscribe the possibility of exchanging application-related configuration information through complementary means (e.g. NETCONF through OSS/EM). The creation of a new NFV Group Report to review and evaluate existing possibilities and standardisation gaps and guidelines was evoked.

10.1.1.3 VNF Descriptors (SOL001, SOL006)

It was a general concern among participants, specially VNF vendors, the lack of uniformity and the number of variations in VNF Descriptors across MANO solutions. Concrete concerns were expressed on the ability of NFV Descriptors to provide interoperability for complex operations, where often, MANO specific extensions seemed to be required to achieve successful results.

10.1.2 MEC Specifications

10.1.2.1 MEC010-2 – Redundant operations

In MEC 010-2 v2.1.1, section 6.3.1.5 “Query application instance information operation”, this method should be to retrieve the status of a given application instance. It seems we can also change its status, which is the same as section 6.3.1.4 “Change application instance operational state operation”. This was seen by participants as redundant.

10.1.2.2 MEC013 – Location API

A request for clarification was raised on JSON serialization for [MEC013] and other MEC ISG specifications.

While the specification sets clear requirements on the information exchanged, the JSON serialization is only defined in informative examples. This may pose interoperability issues between different implementations of the interfaces.

Taking the example of the User Info date type (Clause 6.2.2.of [MEC013]), it is missing a clear requirement that the JSON object should contain the name of the date structure as the root member of the object:

```
{
  "userInfo": {
        ...
   }
}
```

versus the simple

```
{
   ...
}
```

This is depicted in examples and in the Open APIs but these are all informative materials.

Our recommendation would be to clarify this with normative requirements or to leave it to the implementation (but raising the barriers for interoperability and portability).

The issue has been raised on the MEC DECODE WG mailing list and addressed by ETSI GS [MEC009] Rapporteurs with contribution MEC(20)000224 “MEC 009 Data model and serialization”.

10.1.2.3 MEC in NFV

While the interest for MEC in NFV deployments remains high in the Plugtests community, the lack of normative specification is preventing implementation and interoperability opportunities, in particular regarding packetization, data models for artefacts required for onboarding and lifecycle management as well as interfaces between MEC components and NFV components.

10.1.2.4 MEC App Descriptors examples

Availability of a standard common example of MEC Application Descriptor and details on how it is consumed by different components of the MEC architecture would be helpful in speeding up MEC Apps developments.

10.2 Feedback on OpenAPIs

No issue was found on NFV or MEC Open APIs during this Plugtests

10.3 Feedback on the Test Plans

10.3.1 NFV Interoperability Test Plan – TST007

The NFV Interoperability testing was run following the NS related Test Descriptions in [TST007] (mainly clauses 7.3, 7.5, 7.6 and 7.7). See the complete list of Test Descriptions in scope in clause 8.1 of the Plugtests Report. The IFS templates had to be significantly simplified so that they could be used to configure the Test Reporting Tool. See the NFV IFS templates used for this Plugtests in Annex C

10.3.2 NFV Automated Interoperability Test Plan

As explained in 8.1.3, during the NFV&MEC Plugtests 2020 preparation, a new Robot Framework Test Suite, leveraging existing API Conformance Test Cases, was developed to automate IOP testing according to [SOL016] workflow specifications. The full Robot Test suite is available at [NFV-AUTO-IOP] and has been contributed back to NFV TST.

10.3.3 NFV API Conformance Test Plan – TST010

The NFV API Conformance Test Sessions run during the NFV&MEC Plugtests 2020 were based on [TST010]

The main improvement with respect to previous Plugtests was the support of v2.6.1 (in addition of v.2.4.1) and the full support of notifications.

Overall, the NFV&MEC Plugtests 2020 allowed to identify and file 20 new issues on the NFV API Conformance Test Suites. The table below summarises all the issues and indicates the impacted SOL Specification and the number under which the issue was filed in the [ISSUE-TRACKER] set up for that purpose in the ETSI Forge.

Issue	Description	SOL002	SOL003	SOL005
#100	Assignment of response variable @ vs $ in VnfLcmMntOperationKeywords and NSLCMOperationKeywords			X
#101	"Check HTTP Response Status Code Is" method is implemented differently	X	X	X
#102	2.6.1 Check LINK in Header Bug	X	X	
#103	Typos in NsdInfo schemas SOL005 NSDManagement 2.6.1			X
#104	Missing keywords	X	X	X
#105	Notification API / NotificationEndpoint test purposes to be re-designed?	X	X	X
#108	Disable Individual Network Service Descriptor issue 2.6.1-fix-plu			X
#109	Seaparator missing 2.6.1 NSLCM			X
#110	Document ZIP files required to execute the tests and not contained in the repository		X	X
#111	Unresolvable JSON pointer in NSLifecycleManagement-API .NSInstances.robot			X
#112	NSInstances.robot 2.6.1 Attribute Error			X
#113	instantiationState instead of nsState in InstantiateNSTask.robot 2.6.1			X
#114	Etag instead of ETag 2.6.1 IndividualNSInstance.robot			X

#115	NSLifecycleManagement-API.NSLCMOccurences.robot small bugs 2.6.1			X
#116	VnfLcmMntOperationKeywords.robot SOL003 2.6.1 Missing separator		X	
#117	VnfLcmMntOperationKeywords.robot SOL003 2.6.1 AttributeError headers		X	
#118	Response variable not updated after the GET call		X	
#119	Additional parameter to specify arttifact type		X	

Table 51. Issues Reported fort NFV TST010 Robot Test Suites

10.3.4 MEC Interoperability Test Plan

A dedicated MEC Interoperability Test Plan [MEC-IOP-TP] was developed by the Plugtests team to support MEC IOP Test Sessions. The Test Plan will be contributed to ISG MEC DEC WG.

The following feedback was captured on the MEC Interoperability Test Plan:

- Availability in the test plan of examples and conventions for MEC App descriptors may ease integration and foster participation of MEC App providers.
- Upon request of the participants, two new test cases have been added to the test plan: TD_MEC_SVC_TRANSPORTS_QUERY and TD_MEC_SVC_TIMEQUERY.

10.3.5 MEC API Conformance Test Plan – MEC DEC032

The MEC API Conformance Test Plan run during the NFV&MEC Plugtests 2020 were based on MEC [DEC032]. The test suite was implemented in TTCN-3 [MEC-TTCN3-TS] and Robot Framework [MEC-ROBOT-TS].

Overall, the NFV&MEC Plugtests 2020 allowed to identify and file 22 new issues on the MEC API Conformance Test Suites, where 6 of them were related to the TTCN-3 implementation and the other 16 to the Robot Framework implementation. The tables below summarise all the issues and indicates the impacted MEC Test Specifications and the number under which the issue was filed in the [MEC-TTCN-3-ISSUE] and [MEC-ROBOT-ISSUE] set up for that purpose in the ETSI Forge.

Issue	Description	Base Spec
#10	MEC-011: Missing some codec entries	MEC-011
#11	MEC-013: Add missing codec entries	MEC-013
#12	MEC-013: Update version to v2	MEC-013
#13	MEC-013: Review TTCN-3 typing due to latest version of the standard	MEC-013
#14	MEC-011: Review TTCN-3 typing due to latest version of the standard	MEC-013
#15	Add function to create/delete ServiceInfo, Subscription...	MEC-011
#16	TC_MEC_SRV_TRAF_003_PF, review of headers	MEC-011

Table 52: Issues reported for MEC DEC032 TTCN-3 Test Suites

Issue	Description	Base Spec
#7	Response check on DNS testcase for individual query is not proper	MEC-011
#8	Message type wrong in DNS update	MEC-011
#9	Not sending update request on DNS etag error case	MEC-011
#10	SRV/APPSAQ/PlatAppServices.robot check result is not right	MEC-011
#11	SRV/APPSAQ/PlatAppServices.robot should be PUT method @line 214	MEC-011
#12	SRV/SAQ/PlatServices.robot TC_MEC_SRV_SAQ_002_OK get result keyword wrong	MEC-011
#13	SRV/APPSAQ/PlatAppServices.robot TC_MEC_SRV_APPSAQ_002_OK better set service_id as global variable, so the test suite can run together	MEC-011
#14	SRV/SRVSUB/PlatSrvSubscriptions.robot TC_MEC_SRV_SRVSUB_003_OK Check Result Contains keyword wrong	MEC-011
#15	SRV/SRVSUB/PlatSrvSubscriptions.robot TC_MEC_SRV_SRVSUB_002_OK better set $SUBSCRIPTION_ID as golbal variable	MEC-011
#16	SRV/APPSUB/PlatAppSubscriptions.robot TC_MEC_SRV_APPSUB_001_OK SubscriptionLinkList name not right	MEC-011
#17	SRV/APPSUB/PlatAppSubscriptions.robot TC_MEC_SRV_APPSUB_002_OK better set SUBSCRIPTION_ID as global variable and keyword error	MEC-011
#18	SRV/TRAF/PlatTrafficRules.robot TC_MEC_SRV_TRAF_002_OK has multiple issues	MEC-011
#19	SRV/TRAF/PlatTrafficRules.robot TC_MEC_SRV_TRAF_003_OK has multiple issues	MEC-011
#20	TC_MEC_SRV_APPSAQ_002_OK response check is wrong	MEC-011
#21	Schema file name is wrong	MEC-011
#22	Delete trailing comma from json files	MEC-012

Table 53: Issues reported for MEC DEC032 Robot Test Suites

10.4 Other achievements

The NFV&MEC Plugtests saw for the first time some 5G core VNFs and NS participating to the interoperability test sessions. During these, aspects beyond pure virtualisations got tested, such as:

- Network Function (NF) registration and Session Management Function (SMF) discovery in 5G Network Repository Function (NRF) through 5G Core Access and Mobility Management Function (AMF) emulation
- End-to-end 5G core testing through 5G RAN emulation for Registration and PDU session establishment in the virtualised 5G core

The 5G core functions were split in 2 different Network Services, each of them deployed and orchestrated by a different MANO solution, one of which was also in charge of managing the Virtual Test Server.

The virtual Test Administration Server was deployed on a different platform.

These complex test sessions and additional testing allow to provide a glimpse of the ability to run lab and production 5G scenarios within a multi-vendor NFV deployment according to the NFV ISG paradigm.

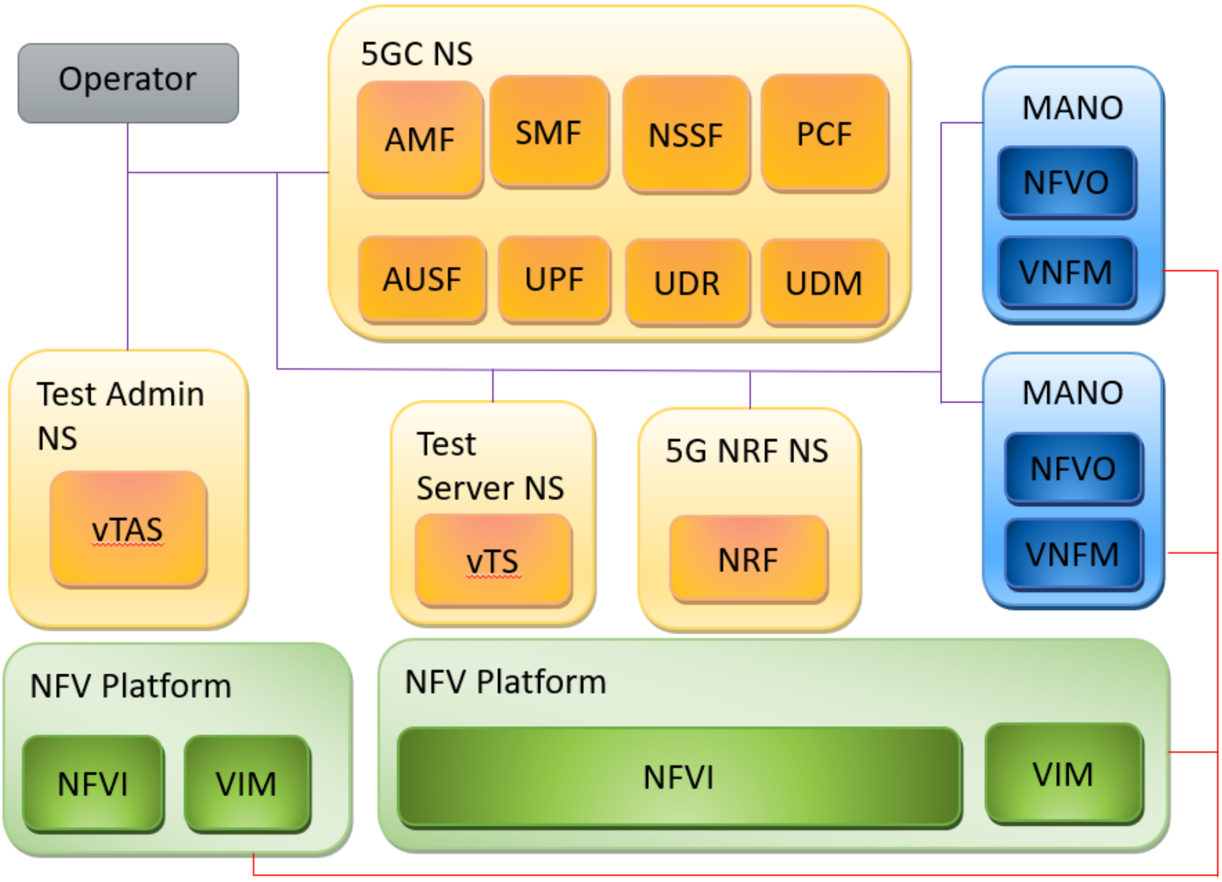

Figure 62 5G Core deployment and functional testing

Annex A – NFV Interoperability Feature Statements

IFS MANO	
IFS_NFV_NFVO_1	NFVO supports multi-site deployments
IFS_NFV_NFVO_2	NFVO supports NS scaling out/in by adding/removing VNF instances
IFS_NFV_NFVO_3	NFVO supports NS scale to level
IFS_NFV_NFVO_4	NFVO supports NS update
IFS_NFV_NFVO_5	NFVO supports receiving VNF indicators from VNFM (query)
IFS_NFV_MFVO_6	NFVO supports receiving VNF indicators from VNFM (notifications)
IFS_NFV_NFVO_7	NFVO supports receiving VNF performance metrics from VNFM (query)
IFS_NFV_NFVO_8	NFVO supports receiving VNF performance metrics from VNFM (notifications)
IFS_NFV_NFVO_9	NFVO supports receiving VNF faults/alarms from VNFM (query)
IFS_NFV_NFVO_10	NFVO supports receiving VNF faults/alarms from VNFM (notifications)
IFS_NFV_NFVO_11	NFVO supports automatic NS scaling out/in triggered by VNF Indicators
IFS_NFV_NFVO_12	NFVO supports automatic NS scaling out/in triggered by performance metrics
IFS_NFV_NFVO_13	NFVO supports provisioning and configuration of network forwarding paths
IFS_NFV_VNFM_2	VNFM supports VNF scaling out/in by adding/removing VNFC instances
IFS_NFV_VNFM_5	VNFM supports receiving VNF indicators from VNF/EM (notifications)
IFS_NFV_VNFM_6	VNFM supports requesting VNF indicators from VNF/EM (query)
IFS_NFV_VNFM_8	VNFM supports receiving virtualised resource performance metrics from VIM
IFS_NFV_VNFM_11	VNFM supports receiving virtualised resource faults/alarms
IFS_NFV_VNFM_14	VNFM supports automatic VNF scaling triggered by VNF indicators
IFS_NFV_VNFM_15	VNFM supports automatic VNF scaling out/in triggered by performance metrics

Table 54: NFV MANO IFS

IFS VNFM	
IFS_NFV_VNFM_1	VNFM supports multi-site deployments (i.e. two or more geographically distributed sites managed by different VIM instances)
IFS_NFV_VNFM_2	VNFM supports VNF scaling out/in by adding/removing VNFC instances
IFS_NFV_VNFM_3	VNFM supports scale-to-level
IFS_NFV_VNFM_4	VNFM supports receiving VNF indicators from VNF/EM (notifications)
IFS_NFV_VNFM_5	VNFM supports requesting VNF indicators from VNF/EM (query)
IFS_NFV_VNFM_6	VNFM exposes VNF Indicators towards NFVO (notifications)
IFS_NFV_VNFM_7	VNFM exposes VNF Indicators towards NFVO (query response)
IFS_NFV_VNFM_8	VNFM supports receiving virtualised resource performance metrics from VIM
IFS_NFV_VNFM_9	VNFM exposes VNF performance metrics towards NFVO (query response)
IFS_NFV_VNFM_10	VNFM exposes VNF performance metrics towards NFVO (notifications)
IFS_NFV_VNFM_11	VNFM supports receiving virtualised resource faults/alarms from VIM
IFS_NFV_VNFM_12	VNFM exposes VNF alarms towards NFVO (query response)
IFS_NFV_VNFM_13	VNFM exposes VNF alarms towards NFVO (notifications)
IFS_NFV_VNFM_14	VNFM supports automatic VNF scaling triggered by VNF indicators from VNF/EM
IFS_NFV_VNFM_15	VNFM supports automatic scaling out/in triggered by performance metrics from VIM

Table 55: NFV VNFM IFS

IFS VNF/CNF	
IFS_NFV_VNF_1	VNF can scale out/in by adding/removing VNFC instances
IFS_NFV_VNF_2	VNF can send VNF Indicators to VNFM (notifications)
IFS_NFV_VNF_3	VNF can send VNF Indicators to VNFM (query response)
IFS_NFV_VNF_4	VNF supports Network Service Headers (NSH) encapsulation
IFS_NFV_VNF_5	VNF supports VNF configuration modification
IFS_NFV_VNF_6	VNF instance can be started/stopped by VNFM/NFVO

Table 56: NFV VNF/CNF IFS

IFS NFVI/VIM	
IFS_NFV_NFVI-VIM_1	VIM exposes virtualised resource performance metrics
IFS_NFV_NFVI-VIM_2	VIM exposes virtualised resource alarms
IFS_NFV_NFVI-VIM_3	VIM offers network forwarding path functionality
IFS_NFV_NFVI-VIM_4	NFVI/VIM supports NSH

Table 57: NFV NFV-VIM IFS

History

Document history		
V1.0.0	30/08/2020	Published version