ETSI GR NFV-REL 007 V1.1.1 (2017-09)

Network Function Virtualisation (NFV);
Reliability;
Report on the resilience of NFV-MANO critical capabilities

Disclaimer

The present document has been produced and approved by the Network Functions Virtualisation (NFV) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Reference
DGR/NFV-REL007

Keywords
management, MANO, NFV, orchestration, resilience

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

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This Group Report (GR) has been produced by ETSI Industry Specification Group (ISG) Network Functions Virtualisation (NFV).

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document **investigates the building of** a resilient NFV-MANO functional block from the reliability/availability perspective. **In order to achieve this objective**, the present document:

1) Identifies critical NFV-MANO capabilities required to provide reliable services to the VNFs and the NSs.

2) Maps the resiliency requirements, e.g. established in ETSI GS NFV-REL 001 [i.2], with existing NFV-MANO capabilities as listed in up to Release 2 GSs.

3) Studies specific needs and constraints for the identified capabilities.

The work reports on possible mechanisms that enable high-availability within the different entities of NFV-MANO to render the identified capabilities dependable and trustworthy.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative 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.

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

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI GS NFV 003: "Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV".

[i.2] ETSI GS NFV-REL 001: "Network Functions Virtualisation (NFV); Resiliency Requirements".

[i.3] ETSI GS NFV-IFA 010 (V2.2.1): "Network Functions Virtualisation (NFV); Management and Orchestration; Functional requirements specification".

[i.4] ETSI GS NFV-IFA 005: "Network Functions Virtualisation (NFV) Release 2; Management and Orchestration; Or-Vi reference point - Interface and Information Model Specification".

[i.5] ETSI GS NFV-IFA 006: "Network Functions Virtualisation (NFV) Release 2; Management and Orchestration; Vi-Vnfm reference point - Interface and Information Model Specification".

[i.6] ETSI GS NFV-IFA 008: "Network Functions Virtualisation (NFV) Release 2; Management and Orchestration; Ve-Vnfm reference point - Interface and Information Model Specification".

[i.7] ETSI GS NFV-IFA 011: "Network Functions Virtualisation (NFV) Release 2; Management and Orchestration; VNF Packaging Specification".

[i.8] ETSI GS NFV-SWA 001: "Network Functions Virtualisation (NFV); Virtual Network Functions Architecture".

[i.9] ETSI GS NFV-REL 003: "Network Functions Virtualisation (NFV); Reliability; Report on Models and Features for End-to-End Reliability".

[i.10] M. Chiosi et al.: "Network Functions Virtualisation - An Introduction, Benefits, Enablers, Challenges & Call for Action", SDN and OpenFlow World Congress, Darmstadt, Germany, October 22-24, 2012.

[i.11] M. Chiosi et al.: "Network Functions Virtualisation - Network Operator Perspectives on Industry Progress", SDN and OpenFlow World Congress, Frankfurt, Germany, October 15-17, 2013.

[i.12] M. Chiosi et al.: "Network Functions Virtualisation - Network Operator Perspectives on Industry Progress", SDN and OpenFlow World Congress, Dusseldorf, Germany, October 14-17, 2014.

[i.13] ETSI GS NFV-MAN 001: "Network Functions Virtualisation (NFV); Management and Orchestration".

[i.14] ETSI GS NFV-IFA 014: "Network Functions Virtualisation (NFV) Release 2; Management and Orchestration; Network Service Templates Specification".

[i.15] ETSI GS NFV-SEC 013: "Network Functions Virtualisation (NFV) Release 3; Security; Security Management and Monitoring Specification".

[i.16] ETSI GR NFV-IFA 021: "Network Functions Virtualisation (NFV); Management and Orchestration; Report on Management of NFV-MANO and Automated Deployment of EM and Other OSS Functions".

[i.17] IETF RFC 4412 (2006): "Communications Resource Priority for the Session Initiation Protocol (SIP)".

[i.18] IETF RFC 4594 (2006): "Configuration Guidelines for DiffServ Service Classes".

[i.19] IETF RFC 5865 (2010): "A Differentiated Services Code Point (DSCP) for Capacity-Admitted Traffic".

[i.20] IETF RFC 4090 (2005): "Fast Reroute Extensions to RSVP-TE for LSP Tunnels".

[i.21] Recommendation ITU-T E.412 (2003): "Network management controls".

3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

(D)DoS	(Distributed) Denial of Service
API	Application Programming Interface
BFD	Bidirectional Forwarding Detection
BSS	Business Support System
COTS	Commercial Off The Shelf
CPU	Central Processor Unit
DF	Deployment Flavour
EM	Element Manager
FCAPS	Fault, Configuration, Accounting, Performance and Security
HA	High Availability
ID	Infrastructure Domain
IFA	(ETSI ISG NFV) Interface and Architecture (Working Group)
IMS	IP Multimedia Subsystem
IO	Input Output
IT	Information Technology
KPI	Key Performance Indicator
L2	Layer 2
L3	Layer 3
LSA	Link State Advertisement
MANO	Management and Orchestration
NF	Network Function
NFP	Network Forwarding Path
NFV	Network Functions Virtualisation

NFVI	NFV Infrastructure
NFVO	NFV Orchestrator
NS	Network Service
NSD	NS Descriptor
NSR	NS Record
OS	Operating System
OSPF	Open Shortest Path First
OSS	Operations Support System
P-CSCF	Proxy - Call Session Control Function
PNF	Physical Network Functions
SAL	Service Availability Level
S-CSCF	Serving - Call Session Control Function
SDN	Software Defined Networking
SDO	Standard Developing Organization
SLA	Service Level Agreement
TD	Tenant Domain
TTM	Time To Market
VDU	Virtualisation Deployment Unit
VIM	Virtualised Infrastructure Manager
VLD	Virtual Link Descriptor
VLR	Virtual Link Record
VM	Virtual Machine
VN	Virtual Network
VNF	Virtualised Network Function
VNFC	VNF Component
VNFD	VNF Descriptor
VNFFG	VNF Forwarding Graph
VNFFGD	VNFFG Descriptor
VNFFGR	VNFFG Record
VNFM	VNF Manager
VNFR	VNF Record
WG	Working Group

4 Introduction

Based on NFV resiliency requirements resulting from [i.2], the present document aims to verify that the current MANO capabilities as specified in [i.3] cover all these requirements. It also proposes, with respect to network services' reliability and availability, a classification of those resiliency requirements in two categories: requirements of normal importance and critical requirements. Through a mapping of the MANO capabilities vs. the resiliency requirements, it identifies the missing capabilities for some resiliency requirements. It finally lists some mechanisms which could be exploited to enhance the identified critical capabilities' resiliency.

To this end, the items processed are the followings:

- listing of the NFV resiliency characteristics;
- identification of resiliency requirements related to MANO;
- alignment of the MANO resiliency requirements within its functional blocks;
- resiliency requirements classification;
- mapping of MANO capabilities vs. resiliency requirements;
- identification of and recommendations for missing MANO capabilities following the mapping;
- proposals for resiliency mechanisms related to the identified critical capabilities.

5 NFV resiliency and its recommendations towards MANO

5.1 Resiliency and its application to NFV

Resiliency - as defined in [i.1] - provides the ability "*to limit disruption and return to normal or at a minimum acceptable service delivery level in the face of a fault, failure, or an event that disrupts the normal operation*".

The definition above mentions fault as distinct from failure. [i.2] has introduced the fault as leading to a failure chain. In a PNF environment, redundancy, the core mechanism of dependability, can be exploited in the following way. In a redundant system composed of two servers, one active and one standby, if the active fails (i.e. a fault occurs), the service is still rendered thanks to the standby one which takes over. It is noteworthy to mention that, in some cases (e.g. the standby server is less performant), the service may be degraded after the failover. Moreover, if the failed server is not replaced in a timely fashion and the new active one also fails, the failure is observed at the service level.

The resiliency engineering for a network service consists of the following phases:

- definition of the resiliency requirements, e.g. availability which can be done through different service availability levels and grades of the service [i.2];
- design for resiliency - this important step mainly targets failure prevention, e.g. with the use of a redundant architecture;
- operations - once the network service is deployed, three other tasks help to maintain the service resiliency:
  - continuous monitoring, e.g. using health check, watchdog mechanisms;
  - detection of abnormal situations, e.g. threshold exceeded;
  - potential actions based on anomalous events - with the use of, e.g. fault correlation, root cause analysis:
    - launch corrective actions, e.g. prevent failures to happen;
    - prevent propagation of a breakdown, i.e. failure containment - needless to say, as not all failures are equal, the study of failure severity, e.g. number of users impacted, is generally done in the design phase;
    - diagnose that an entity is deteriorating, i.e. failure prediction.

During the operations, the failures are diverse: they can come from the infrastructure level, the VNF layer, or can concern the MANO software (see annex A for a reminder of MANO). As such, the MANO components NFVO, VNFM and VIM play critical roles in ensuring VNFs resiliency from their instantiation through their operation and recovery from failures. With respect to some life cycle management tasks (e.g. instantiation, scaling, migration), MANO is responsible for adhering to affinity and anti-affinity rules to ensure there is no single point of failure. Actually, the application of redundancy in the NFV framework implies the creation of secondary VNF instances (for those VNF considered as critical), but this duplication is useless if both instances run on the same server or NFVI-PoP (i.e. a single point of failure). Anti-affinity rules thus impose the use of different infrastructures for those instances. On the other hand, performance constraints may need that different VNF instances run in the same location (affinity rules).

Based on the type of failure, the MANO components provide the appropriate recovery or remediation such as VNF re-instantiation/re-creation, VNF scaling (i.e. adding a VNF instance to provide more capacity so that the service is not degraded), VNF migration (i.e. moving the VNF to another server during maintenance or hardware failures), or failure containment (i.e. keep it from propagating further and negatively impact other VNFs), while ensuring service continuity.

NOTE: For the sake of clarity, the present document has adopted a simplistic picture of VNFs as composed of a single VM wherever it was possible. It is explicitly noted when this is not the case.

The MANO components, particularly the NFVO, can have an impact beyond a single VNF, as they are able to provide orchestration within a service chain or between geographically separated VNFs or between multi-vendor VNFs. The next clause will provide details on the recommendations and the role of the individual MANO components.

5.2 Resiliency recommendations towards MANO

5.2.1 MANO-related resiliency requirements classification

Annex B shows the resiliency requirements identified in [i.2]. Most of these items are related to MANO and are thus analysed in this clause. The ones which have been excluded from the analysis are treated in clause 5.2.2.

This clause classifies the [i.2] requirements related to MANO: the classification is based on the pertinent MANO component i.e. NFVO, VNFM, VIM. The requirements which involve more than one MANO component are gathered under the MANO umbrella.

Note that any "item" not expressed as such in [i.2], but implied by a requirement, is set in italic. As an example, the requirement 10.8.13 in [i.2] states that "The VNF shall inform the VNFM in the event that it cannot remediate a failure that results in a degradation of the overall capacity of the VNF". This "item" thus states that "The VNFM receives remediation failure information from VNFs and acts accordingly" and it is listed in italic in reference to the requirement 10.8.13 in [i.2]. The resiliency keywords and main actions are underlined for ease of reading.

NFVO

The NFV Orchestrator has two main responsibilities:

(i) orchestration of NFVI resources across multiple VIMs;

(ii) lifecycle management of network services.

The NFVO is also responsible for activities ensuring that the VNFs and the network services they support meet any reliability and availability requirements by passing on resiliency requirement data or failure information to other MANO components and/or taking corrective actions on its own:

- interpret **resiliency requirement data** contained in NSD and VNFD (e.g. NFVI geo-redundancy) and transfer it to VIM and potentially to VNFM as appropriate (see [i.2], Req. 4.2.2);
- take **corrective actions** for other VNFs of a service chain in case a VNF failure is detected (see [i.2], Req. 10.8.18).

VNFM

The VNF Manager supports operations of VNF instances lifecycle management, e.g. health monitoring and recovery upon failure, which includes the following:

- provide an **interface** to VNFs for health checking (see [i.2], Req. 10.8.17);
- **restart** a VNF having exceeded its health check response timeout (see [i.2], Req. 10.8.20);
- *receive remediation **failure information** from VNFs and act accordingly (see [i.2], Req. 10.8.13);*
- **request additional resources** in case VNF encounters failures (see [i.2], Req. 10.8.14).

NOTE: In case of VNF scaling, the VNF may consist of more than one VM.

VIM

The Virtualised Infrastructure Manager (VIM) is responsible for controlling and managing the NFVI compute, storage and network resources. In that capacity if the VIM detects a possible impending resource shortage it can initiate pre-emptive actions to provide additional resources before the shortage occurs, and/or launch admission mechanisms to limit the access to resources. If an NFVI-related failure occurs, it can take corrective actions such as migrating VMs to other servers:

- receive **resiliency requirement data** derived from NSD and VNFD (e.g. NFVI geo-redundancy) sent by NFVO and act accordingly (see [i.2], Req. 4.2.2);
- apply anti-affinity policies for preventing **single point of failure** (see [i.2], Req. 9.5.1 and 9.5.2);
- hardware resource **monitoring** (see [i.2], Req. 5.4.10, 10.8.5 and 10.8.11);

- get real-time **resource** (e.g. CPU, memory) **usage information** and act accordingly (see [i.2], Req. 9.3.1);
- get **health information** from the infrastructure and act accordingly (see [i.2], Req. 9.3.6);

NOTE: VNFs are not able to report to the VIM nor the VIM is aware of the VNFs.

- get indication of hardware and environmental events from the NFVI layer and **migrate** pro-actively the VMs impacted by such faulty events (see [i.2], Req. 10.8.15);
- receive **failure detection information** from the hypervisor (and/or its underlying host) and act accordingly (see [i.2], Req. 10.8.4);
- take **corrective actions** following NFVI failures (see [i.2], Req. 10.8.16);
- assign dedicated resources to a VM for failure **containment** (i.e. propagation avoidance) (see [i.2], Req. 9.2.1):
  - in conjunction with the hypervisor (see [i.2], Req. 9.2.2);
  - by defining maximum virtual resources allocated to VMs, e.g. for attack containment (see [i.2], Req. 9.2.3).
- transport network (e.g. virtual network) redundancy (see [i.2], Req. 9.5.3).

MANO

The MANO components work together to ensure service availability and continuity for VNFs and adherence to the terms laid out in SLAs. In support of life cycle management operations, the MANO components make sure that the operations are carried out successfully, or provide support for the recovery of any failed actions. During the instantiation of VNFs, anti-affinity rules have to be followed such that there is no single point of failure. This rule is enforced by MANO:

- no **single point of failure** (see [i.2], Req. 4.2.13), e.g. redundant deployment (see [i.2], Req. 10.8.21) such as geo-redundancy (see [i.2], Req. 10.8.22) → NFVO, VNFM, VIM.

The MANO components that are configured to manage multiple VNFs have to be able to manage each of them separately and, in each case, according to their requirements:

- resiliency for **multi-vendor** environment (see [i.2], Req. 4.2.14) → NFVO, VNFM, VIM.

When priorities are assigned to VNFs, the MANO components are responsible for complying with those priorities by ensuring that when resources are reaching capacity limits, sufficient resources are allocated as per those priorities (i.e. VNFs with higher priorities are satisfied first even when VNFs of lower priorities are only partially or not satisfied at all):

- VNFs **priority handling** for resource allocation (see [i.2], Req. 5.4.7) → NFVO, VNFM;
- support of service availability levels and grades of service (e.g. following the SLA expressed in VNFD, NFVO/VNFM will order VIM to allocate the appropriate resources) (see [i.2], Req. 7.3.1, 7.3.2 and 7.3.3) → NFVO, VNFM, VIM;
- *analysis of load information (from VNFs, hypervisor, …), e.g. determine if scaling or migration or … is needed (see [i.2], Req. 9.4.2)* → VNFM, VIM;
- support **watchdog** functionality whose characteristics include reliable triggering mechanisms and low latency notifications (see [i.2], Req. 10.8.10) → VNFM, VIM;
- support of a means to indicate VNFs' ability to support external health check (see [i.2], Req. 10.8.19) → NFVO, VNFM;
- ensure **service continuity** (at session/service establishment and during service relocation) (see [i.2] Req. 4.2.11) by respecting the acceptable **disruption time** (see [i.2], Req. 5.4.2) defined in the SLA (see [i.2], Req. 5.4.3) → NFVO, VNFM, VIM;

- support (receive, analyse) **failure notification** at run time and act accordingly (see [i.2], Req. 4.2.5), e.g. data from NFVI (see [i.2], Req. 4.2.6) → NFVO, VNFM, VIM;
- support of VNF **scaling** → NFVO, VNFM:
  - on site and/or offsite (see [i.2], Req. 7.3.8);
  - within the VNF's limits, limit the number of users in order to minimize the failures impact (see [i.2] Req. 4.2.10);
  - including efficient load distribution/balancing (see [i.2], Req. 5.4.9).
- appropriate handling of malicious traffic to avoid excessive resource consumption (see [i.2], Req. 7.3.9) → NFVO, VNFM, VIM;
- support of VNF **migration**, e.g. following a hardware, a software failure, or resource shortage (see [i.2] Req. 5.4.1) → NFVO, VNFM, VIM (support of VM migration when requested):
  - on site and/or offsite (see [i.2], Req. 7.3.8);
  - maintaining pre-migration communication between VNFs (see [i.2], Req. 5.4.4);
  - the pre-migration process (which may be activated during the initial instantiation of the VNF) includes the availability of VNFD (to retrieve deployment conditions and/or VNF constraints) (see [i.2], Req. 5.4.6).
- support of automated fail-over on the NFVI level (see [i.2], Req. 4.2.12) → NFVO, VNFM, VIM.

NOTE 1: Automated fail-over of compute resources hosting stateful VNFs with internal redundancy is undesirable as it may result in fail-overs at both NFVI and VNF levels resulting in a conflicting situation.

- **pre-emption** capability (VNFM) and support of this capability (VIM), e.g. in case of resource shortage (see [i.2], Req. 5.4.8) → VNFM, VIM;
- overload control to be considered in the implementation (see [i.2], Req. 9.4.3) → NFVO, VNFM, VIM;
- *handling of fields error **severity** (see [i.2], Req. 9.3.7, 10.8.8) and failure cause (see [*i.2*], Req. 9.3.8) received* → NFVO, VNFM, VIM;
- provide visibility (into specific classes of hardware failures) to the VNFs (see [i.2], Req. 10.8.2) → NFVO, VNFM, VIM:
  - send hardware failure notifications (see [i.2], Req. 10.8.3) → NFVO, VNFM, VIM.

NOTE 2: The exploitation of hardware failure information by the VNFs is for further study.

- enabling to provide a **degraded service** under abnormal conditions (see [i.2], Req. 7.3.4) → NFVO, VNFM, VIM (support of virtualised resource management when requested);
- **failure prediction** framework:
  - false alarm filtering to avoid triggering unnecessary prevention procedure for anomaly alarming (see [i.2], Req. 9.3.3) → VNFM, VIM (involved as alarm manager);
  - trend identification, period or seasonal variations, and randomness analysis of data on resource usage to predict unhealthy state (e.g. resource exhausting) (see [i.2], Req. 9.3.4) → VNFM, VIM;
  - diagnose/verify which entity is progressing towards a failure and which VNFs might be affected (see [i.2], Req. 9.3.5) → VNFM, VIM (support of fault management related to virtualised resources).
- distributed **fault correlation** processing: avoiding propagating large number of failure notifications to a centralized entity by sending locally correlated reports (see [i.2], Req. 10.8.7) → NFVO, VNFM, VIM;
- automatic VNF **re-instantiation** following a failure, e.g. VM failure (see [i.2], Req. 4.2.4) → NFVO, VNFM:
  - within specified time intervals (see [i.2], Req. 7.3.7);

- retrieve VNF state information (previously stored) before VNF recreation, e.g. after a failure (see [i.2], Req. 4.2.9).

- service **admission control** capability (see [i.2], Req. 7.3.5) → NFVO, VNFM, VIM;
- MANO failure:
  - not affecting existing VNF instances and no change in existing VNFs service level (see [i.2] Req. 10.8.23) → NFVO, VNFM, VIM;
  - integrity preserved (e.g. during orchestration) thanks to the atomicity of internal state data change (see [i.2], Req. 10.8.25) → NFVO, VNFM, VIM;
  - recover the state of the environment by means of restoring persistent storage and audit the environment for determining its state, without interruption to the in service VNF instances (see [i.2], Req. 10.8.24) → NFVO, VNFM, VIM.

Figure 5.1 shows the resiliency requirements analysed in this clause and their positioning within the MANO functional blocks.

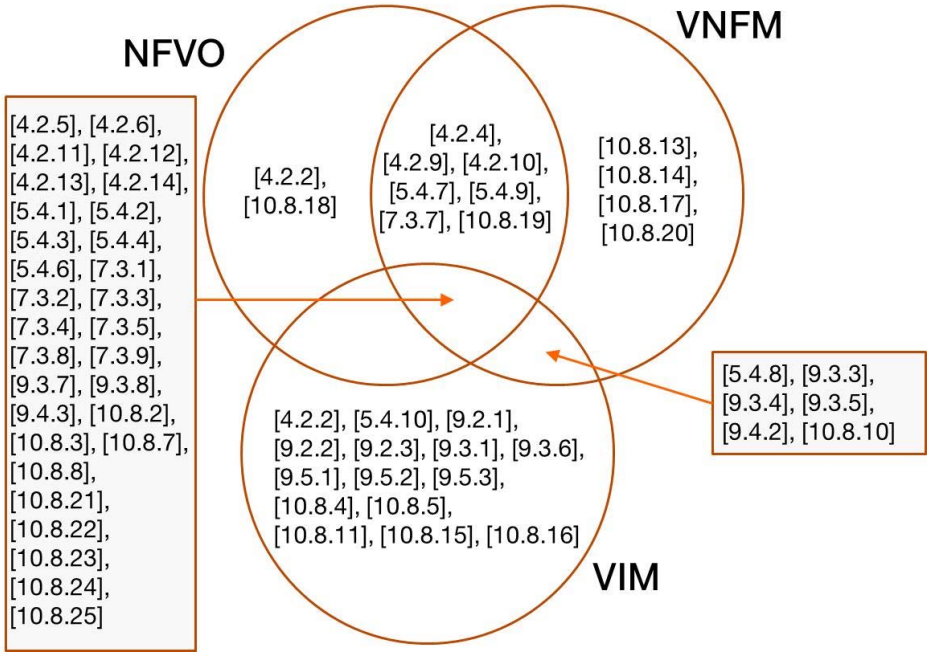

Figure 5.1: Positioning of resiliency requirements within the MANO functional blocks

Table 5.1 shows the resiliency requirements with respect to the phases of resiliency engineering in the 'Life cycle vs. MANO functional blocks' matrix. Note that the failure prediction and fault correlation tasks are regrouped under 'Analysis'.

Table 5.1: Resiliency requirements in the 'Life cycle vs. MANO functional blocks' matrix

	Requir.	Design		Operations								
				Service Instant.	Monitoring		Analysis		Reaction			
NFVO	[4.2.2]	[10.8.19]	[4.2.11] [4.2.13] [4.2.14] [5.4.2] [5.4.3] [5.4.7] [7.3.1] [7.3.2] [7.3.3] [10.8.21] [10.8.22] [10.8.23] [10.8.24] [10.8.25]		[10.8.17]	[10.8.10]	[9.3.3] [9.3.4] [9.3.5]	[10.8.7]	[10.8.18]	[4.2.4] [4.2.9] [4.2.10] [5.4.9] [7.3.7]	[5.4.8]	[4.2.5] [4.2.6] [4.2.12] [5.4.1] [5.4.4] [5.4.6] [7.3.4] [7.3.5] [7.3.8] [7.3.9] [9.3.7] [9.3.8] [9.4.2] [9.4.3] [10.8.2] [10.8.3] [10.8.8]
VNFM									[10.8.13] [10.8.14] [10.8.20]			
VIM		[9.5.3]		[4.2.1] [9.5.1] [9.5.2]	[5.4.10] [10.8.5] [10.8.11]				[9.2.1] [9.2.2] [9.2.3] [9.3.1] [9.3.6] [10.8.4] [10.8.15] [10.8.16]			

5.2.2 Non MANO-related resiliency requirements

The [i.2] resiliency requirements shown in italic in annex B were not analysed in the present document. They refer to subjects such as VNFs, NS and VNF descriptors, NFVI (including hypervisor), resiliency classes, use of reference interfaces, communication of performance and consumption figures, communication of failure reports.

6 Critical capabilities identification and ranking

6.1 Critical capabilities analysis

As the three NFV-MANO functional blocks constitute the control plane of the NFV system, the common expectation is that NFVO, VNFM, and VIM are all highly available, i.e. that all ETSI GS NFV-REL 001 [i.2] requirements related to them are classified "critical". It is proposed in this clause to relax this expectation by introducing a second, and lower, class of severity for them. Two different categories can thus be defined:

- critical ETSI GS NFV-REL 001 [i.2] requirements necessitating a high level of availability;
- other ETSI GS NFV-REL 001 [i.2] requirements with a fair level of availability.

The input used in this clause results from the classification realized in clause 5.2.1.

NFVO

Critical resiliency requirements:

- interpret **resiliency requirement data** contained in NSD and VNFD (e.g. NFVI geo-redundancy) and transfer it to VIM and potentially to VNFM as appropriate (see [i.2], Req. 4.2.2);

- take **corrective actions** for other VNFs of a service chain in case a VNF failure is detected (see [i.2], Req. 10.8.18).

VNFM

Critical resiliency requirements:

- provide an **interface** to VNFs for health checking (see [i.2], Req. 10.8.17);
- **restart** a VNF having exceeded its health check response timeout (see [i.2], Req. 10.8.20);
- *receive remediation **failure information** from VNFs and act accordingly (see [i.2], Req. 10.8.13);*
- **request additional resources** in case VNF encounters failures (see [i.2], Req. 10.8.14).

VIM

Critical resiliency requirements:

- receive **resiliency requirement data** derived from NSD and VNFD (e.g. NFVI geo-redundancy) sent by NFVO and act accordingly (see [i.2], Req. 4.2.2);
- apply anti-affinity policies for preventing **single point of failure** (see [i.2], Req. 9.5.1 and 9.5.2);
- hardware resource **monitoring** (see [i.2], Req. 5.4.10, 10.8.5 and 10.8.11);
- get real-time **resource** (e.g. CPU, memory) **usage information** and act accordingly (see [i.2], Req. 9.3.1);
- get **health information** from the infrastructure and act accordingly (see [i.2], Req. 9.3.6);
- receive **failure detection information** from the hypervisor (and/or its underlying host) and act accordingly (see [i.2], Req. 10.8.4);
- take **corrective actions** following NFVI failures (see [i.2], Req. 10.8.16);
- assign dedicated resources to a VM for failure **containment** (i.e. propagation avoidance) (see [i.2], Req. 9.2.1):
  - in conjunction with the hypervisor (see [i.2], Req. 9.2.2);
  - by defining maximum virtual resources allocated to VMs, e.g. for attack containment (see [i.2], Req. 9.2.3).
- transport network (e.g. virtual network) redundancy (see [i.2], Req. 9.5.3).

Other resiliency requirements:

- Get indication of hardware and environmental events from the NFVI layer and **migrate** pro-actively the VMs impacted by such faulty events (see [i.2], Req. 10.8.15).

NOTE: A bad execution of this pro-active action does not necessarily lead to a failure.

MANO

Critical resiliency requirements:

- no **single point of failure** (see [i.2], Req. 4.2.13), e.g. redundant deployment (see [i.2], Req. 10.8.21) such as geo-redundancy (see [i.2], Req. 10.8.22) → NFVO, VNFM, VIM;
- resiliency for **multi-vendor** environment (see [i.2], Req. 4.2.14) → NFVO, VNFM, VIM;
- support of service availability levels and grades of service (e.g. following the SLA expressed in VNFD, NFVO/VNFM will order VIM to allocate the appropriate resources) (see [i.2], Req. 7.3.1, 7.3.2 and 7.3.3) → NFVO, VNFM, VIM;

- support **watchdog** functionality whose characteristics include reliable triggering mechanisms and low latency notifications (see [i.2], Req. 10.8.10) → VNFM, VIM;
- support of a means to indicate VNFs' ability to support external health check (see [i.2], Req. 10.8.19) → NFVO, VNFM;
- ensure **service continuity** (at session/service establishment and during service relocation) (see [i.2], Req. 4.2.11) by respecting the acceptable **disruption time** (see [i.2], Req. 5.4.2) defined in the SLA (see [i.2], Req. 5.4.3) → NFVO, VNFM, VIM;
- support (receive, analyse) **failure notification** at run time and act accordingly (see [i.2], Req. 4.2.5), e.g. data from NFVI (see [i.2], Req. 4.2.6) → NFVO, VNFM, VIM;
- appropriate handling of malicious traffic to avoid excessive resource consumption (see [i.2], Req. 7.3.9) → NFVO, VNFM, VIM;
- support of VNF **migration**, e.g. following a hardware, a software failure, or resource shortage (see [i.2], Req. 5.4.1) → NFVO, VNFM, VIM (support of VM migration when requested):
  - on site and/or offsite (see [i.2], Req. 7.3.8);
  - maintaining pre-migration communication between VNFs (see [i.2], Req. 5.4.4);
  - the pre-migration process (which may be activated during the initial instantiation of the VNF) includes the availability of VNFD (to retrieve deployment conditions and/or VNF constraints) (see [i.2], Req. 5.4.6).
- support of automated fail-over on the NFVI level (see [i.2], Req. 4.2.12) → NFVO, VNFM, VIM;
- overload control to be considered in the implementation (see [i.2], Req. 9.4.3) → NFVO, VNFM, VIM;
- VNFs **priority handling** for resource allocation (see [i.2], Req. 5.4.7) → NFVO, VNFM;
- *handling of fields error **severity** (see [i.2], Req. 9.3.7 and 10.8.8) and failure cause (see [i.2], Req. 9.3.8) received* → NFVO, VNFM, VIM;
- provide visibility (into specific classes of hardware failures) to the VNFs (see [i.2], Req. 10.8.2) → NFVO, VNFM, VIM:
  - send hardware failure notifications (see [i.2], Req. 10.8.3) → NFVO, VNFM, VIM.
- enabling to provide a **degraded service** under abnormal conditions (see [i.2], Req. 7.3.4) → NFVO, VNFM, VIM (support of virtualised resource management when requested);
- automatic VNF **re-instantiation** following a failure, e.g. VM failure (see [i.2], Req. 4.2.4) → NFVO, VNFM:
  - within specified time intervals (see [i.2], Req. 7.3.7);
  - retrieve VNF state information (previously stored) before VNF recreation, e.g. after a failure (see [i.2], Req. 4.2.9).
- MANO failure: [i.2]:
  - not affecting existing VNF instances and no change in existing VNFs service level (see [i.2], Req. 10.8.23) → NFVO, VNFM, VIM;
  - integrity preserved (e.g. during orchestration) thanks to the atomicity of internal state data change (see [i.2], Req. 10.8.25) → NFVO, VNFM, VIM;
  - recover the state of the environment by means of restoring persistent storage and audit the environment for determining its state, without interruption to the in service VNF instances (see [i.2], Req. 10.8.24) → NFVO, VNFM, VIM.

Other resiliency requirements:

- **Pre-emption** capability (VNFM) and support of this capability (VIM), e.g. in case of resource shortage (see [i.2], Req. 5.4.8) → VNFM, VIM.

NOTE 1: Ignoring this requirement may lead to performance issue, but not necessarily to failures.

- Service **admission control** capability (see [i.2], Req. 7.3.5) → NFVO, VNFM, VIM.

NOTE 2: Ignoring this requirement may lead to performance issue, but not necessarily to failures.

- Analysis of load information (from VNFs, hypervisor, …), e.g. determine if scaling or migration or … is needed (see [i.2], Req. 9.4.2) → VNFM, VIM.

NOTE 3: Scaling/migration are considered to have a lower level of criticality.

- Support of VNF **scaling** → NFVO, VNFM:
  - on site and/or offsite (see [i.2], Req. 7.3.8);
  - within the VNF's limits, limit the number of users in order to minimize the failures impact (see [i.2], Req. 4.2.10);
  - including efficient load distribution/balancing (see [i.2], Req. 5.4.9).

NOTE 4: For instance, if the scale out does not work, additional customers are not accepted but current ones are not (should not be) impacted.

- Failure prediction framework:
  - false alarm filtering to avoid triggering unnecessary prevention procedure for anomaly alarming (see [i.2], Req. 9.3.3) → VNFM, VIM (involved as alarm manager);
  - trend identification, period or seasonal variations, and randomness analysis of data on resource usage to predict unhealthy state (e.g. resource exhausting) (see [i.2], Req. 9.3.4) → VNFM, VIM;
  - diagnose/verify which entity is progressing towards a failure and which VNFs might be affected (see [i.2], Req. 9.3.5) → VNFM, VIM (support of fault management related to virtualised resources).

NOTE 5: Ignoring this requirement has an indirect impact on failure.

- Distributed **fault correlation** processing: avoiding propagating large number of failure notifications to a centralized entity by sending locally correlated reports (see [i.2], Req. 10.8.7) → NFVO, VNFM, VIM.

NOTE 6: Ignoring this requirement may lead to performance issue, but not necessarily to failures.

6.2 Mapping of capabilities vs. resiliency requirements

6.2.1 Introduction

By mapping the MANO-related resiliency requirements shown in clause 6.1 against the NFV-MANO requirements expressed in [i.3], clauses 6.2.2 and 6.2.3 help to identify those resiliency requirements which have not been addressed by the current capabilities of NFV-MANO.

6.2.2 Mapped resiliency requirements

NFVO

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Capability description
Interpret **resiliency requirement data** contained in NSD and VNFD (e.g. NFVI geo-redundancy) and transfer it to VIM and potentially to VNFM as appropriate (see [i.2], Req. 4.2.2)	**General virtualised resource management**	
	Nfvo.Gvrm.003	Provide deployment-specific configuration information for virtualised resources related to NS
	VNF-related resource management in indirect mode (if applicable)	
	Nfvo.VnfR pbNfvo.005	Request to the VIM affinity and anti-affinity policies for the VNF's virtualised resources
	Resource reservation management	
	Nfvo.Rrm.002	Consider affinity/anti-affinity rules for resource reservation management
	Network forwarding path management	
	Nfvo.Nfpm.002	Provide or update the classification and selection rules applied to a specific NFP instance
	Nfvo.Nfpm.003	Receive the classification and selection rules applied to NFP(s) from an authorized entity, e.g. OSS/BSS
Take **corrective actions** for other VNFs of a service chain in case a VNF failure is detected (see [i.2], Req. 10.8.18) Support of a means to indicate VNFs' ability to support external health check (see [i.2], Req. 10.8.19)	**Virtualised resource fault management**	
	Nfvo.Vrfm.003	Request corrective operations on virtualised network resources to VIM in order to perform NS healing
	Virtualisation-related fault management	
	Nfvo.VirFm.001	Request VNF healing to VNFM (see NOTE)
	NS fault management	
	Nfvo.NsFm.004	Perform automated or on-demand healing on the NSs it manages
	Nfvo.NsFm.006	Evaluate the impact on NS instance(s) the NFVO manages when NS healing needs to be performed on a component instance (i.e. a VNF or nested NS) shared or not

NOTE 1: [i.4] and [i.5] do not seem to support this healing concept.

VNFM

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement (see note 3)	Description
Provide an **interface** to VNFs for **health check** registration (see [i.2], Req. 10.8.17) Support of a means to indicate VNFs' ability to support external health check (see [i.2], Req. 10.8.19) (see note 2)	**VNF-related resource management in direct mode**	
	Vnfm.VnfRmpbVnfm.003	Receive notifications regarding the resources being allocated to, or released from, specific VNF instances, as well as regarding events and relevant fault reports related to those resources
	Virtualised resource fault management	
	Vnfm.Vrfm.001	Collect fault information related to the virtualised resources assigned to VNF instance(s) which are managed by the VNFM
	Vnfm.Vrfm.002	Correlate virtualised resource fault information with the impacted VNF(C) instance(s) which are managed by the VNFM
	Virtualised resource-related VNF fault management	
	Vnfm.VrVnfFm.001	Provide notifications of virtualised resource-related fault information on the VNFs which are managed by the VNFM
	Vnfm.VrVnfFm.002	Provide virtualised resource-related fault information on the VNFs which are managed by the VNFM
	Vnfm.VrVnfFm.003	Provide notifications of changes in virtualised resource-related fault information related to the VNFs which are managed by the VNFM
	Vnfm.VrVnfFm.004	Notify about alarms on VNF and any of its VNF Components as a consequence of state changes in the virtualised resources used by the VNF and its VNF Components

NOTE 2: At first glance, this requirement is related to the provision of an interface by VNFM. However, as VNFs' health is concerned, notifications regarding fault reports, collection and correlation of fault information from resources, virtualised or not, have also been included in the mapping above.

NOTE 3: The functionalities related to virtualisation-related fault management (Vnfm.VirFm.xxx) were not considered as the approach distinguished two steps in restoration:

(i) interface and reporting, collection and correlation;

(ii) reaction (performing actions tasks).

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Restart a VNF having exceeded its health check response timeout (see [i.2], Req. 10.8.20)	**Virtualised resource-related VNF fault management**	
	Vnfm.VrVnfFm.005	Request corrective operations on virtualised resources to VIM in order to perform VNF healing (see note 4)
	Virtualisation-related fault management	
	Vnfm.VirFm.001	Perform on-demand VNF healing on the VNF(s) which are managed by the VNFM
	Vnfm.VirFm.002	Perform automated VNF healing on the VNF(s) which are managed by the VNFM

NOTE 4: [i.6] defines some VNF indicators which can be used for this purpose.

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement (see note 5)	Description
Receive remediation **failure information** from VNFs and act accordingly (see [i.2], Req. 10.8.13)	**VNF-related resource management in direct mode**	
	Vnfm.VnfRmpbVnfm.003	Receive notifications regarding the resources being allocated to or released from specific VNF instances, as well as regarding events and relevant fault reports related to those resources
	Virtualised resource fault management	
	Vnfm.Vrfm.001	Collect fault information related to the virtualised resources assigned to VNF instance(s) which are managed by the VNFM
	Vnfm.Vrfm.002	Correlate virtualised resource fault information with the impacted VNF(C) instance(s) which are managed by the VNFM
	Virtualised resource-related VNF fault management	
	Vnfm.VrVnfFm.001	Provide notifications of virtualised resource-related fault information on the VNFs which are managed by the VNFM
	Vnfm.VrVnfFm.002	Provide virtualised resource-related fault information on the VNFs which are managed by the VNFM
	Vnfm.VrVnfFm.003	Provide notifications of changes in virtualised resource-related fault information related to the VNFs which are managed by the VNFM
	Vnfm.VrVnfFm.004	Notify about alarms on VNF and any of its VNF Components as a consequence of state changes in the virtualised resources used by the VNF and its VNF Components
	Vnfm.VrVnfFm.005	Request corrective operations on virtualised resources to VIM in order to perform VNF healing
	Virtualisation-related fault management	
	Vnfm.VirFm.001	Perform on-demand VNF healing on the VNF(s) which are managed by the VNFM
	Vnfm.VirFm.002	Perform automated VNF healing on the VNF(s) which are managed by the VNFM

NOTE 5: Acting accordingly implies diverse tasks such as resource verification such as receiving notifications from VIM (Vnfm.VnfRmpbVnfm.003), treating virtualised resource fault information (Vnfm.Vrfm.xxx), providing appropriate data for restoration, e.g. to NFVO (Vnfm.VrVnfFm.xxx).

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Request additional resources in case VNF encounters failures (see [i.2], Req. 10.8.14)	**VNF-related resource management in direct mode**	
	Vnfm.VnfRmpbVnfm.004	Request allocation and update of resources in the different resource commitment models
	Vnfm.VnfRmpbVnfm.006	Use resource reservation identification information obtained from the NFVO to request allocation of virtualised resources for a VNF (if a resource reservation model is used)

VIM

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
*Receive **resiliency requirement data*** derived from NSD and VNFD (e.g. NFVI geo-redundancy) sent by NFVO and act accordingly (see [i.2], Req. 4.2.2)	**Virtualised resource management**	
	Vim.Vrm.006	Enforce affinity and anti-affinity policies for NFVI resource management
	Vim.Vrm.007	Receive the virtualised resource management requests from VNFM and/or NFVO, and conduct the corresponding resource management operations
	Resource reservation management	
	Vim.Rrm.003	Infer information about what reservation is applicable by using input information received with the allocation or update request (case of the reservation model)
	Vim.Rrm.006	Consider affinity/anti-affinity rules for resource reservation management
	Virtualised resource configuration management	
	Vim.Vrcm.001	Configuration management of an individual virtualised resource using specific deployment configuration information received
	Vim.Vrcm.002	Configuration management of a set of related virtualised resources using specific deployment configuration information received

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Apply anti-affinity policies for preventing **single point of failure** (see [i.2], Req. 9.5.1 and 9.5.2]	**Virtualised resource management**	
	Vim.Vrm.006	Enforce affinity and anti-affinity policies for NFVI resource management
	Resource reservation management	
	Vim.Rrm.006	Consider affinity/anti-affinity rules for resource reservation management

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Hardware resource **monitoring** (see [i.2], Req. 5.4.10, 10.8.5 and 10.8.11)	**Infrastructure resource fault management**	
	Vim.Irfm.001	Correlate fault information on virtualised resources with fault information related to underlying used software and hardware resources within the NFVI
	Vim.Irfm.002	Collection of fault information related to software and hardware resources within the NFVI
	Infrastructure resource performance management	
	Vim.Irpm.001	Collection of performance information related to software and hardware resources within the NFVI

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Get real-time **resource** (e.g. CPU, memory) **usage information** and act accordingly (see [i.2], Req. 9.3.1) (see note 6)	**Infrastructure resource performance management**	
	Vim.Irpm.001	Collection of performance information related to software and hardware resources within the NFVI

NOTE 6: The understanding is that infrastructure resources are taken into account here, while virtualised resources are treated in the next item.

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Get **health information** from the infrastructure and act accordingly (see [i.2], Req. 9.3.6)	**Virtualised resource management**	
	Vim.Vrm.001	Support NFVI resource management within the VIM area of responsibility
	Vim.Vrm.004	Correlate allocated and reserved virtualised resources with changes on underlying hardware/software resources due to maintenance, operation and management of the NFVI, and change the state of the allocated and reserved virtualised resources accordingly
	Vim.Vrm.005	Notify changes about allocated and reserved virtualised resources

NOTE 7: At first glance, this requirement is related to the provision of an interface by VIM. However, as VNFs' health is concerned, notifications regarding fault reports, collection and correlation of fault information from resources, virtualised or not, have also been included in the mapping.

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Get indication of hardware and environmental events from the NFVI layer and **migrate** pro-actively the VMs impacted by such faulty events (see [i.2], Req. 10.8.15) - (see note 8)	**Virtualised resource management (see note 9)**	
	Vim.Vrm.001	Support NFVI resource management within the VIM area of responsibility
	Vim.Vrm.004	Correlate allocated and reserved virtualised resources with changes on underlying hardware/software resources due to maintenance, operation and management of the NFVI, and change the state of the allocated and reserved virtualised resources accordingly

NOTE 8: This resiliency requirement was identified as non-critical in clause 7.1.

NOTE 9: "Virtualised Resources Management operations include allocation, migration, scaling, update, query, operation and termination of virtualised resources" [i.3].

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Receive **failure detection information** from the hypervisor (and/or its underlying host) and act accordingly (see [i.2], Req. 10.8.4)	**Virtualised resource fault management**	
	Vim.Vrfm.001	Collect fault information related to virtualised resources
	Vim.Vrfm.004	Perform automated or on-demand corrective operations on virtualised resources failure
	Infrastructure resource fault management	
	Vim.Irfm.001	Correlate fault information on virtualised resources with fault information related to underlying used software and hardware resources within the NFVI
	Vim.Irfm.002	Collection of fault information related to software and hardware resources within the NFVI

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Take **corrective actions** following NFVI failures (see [i.2] Req. 10.8.16)	**Virtualised resource fault management**	
	Vim.Vrfm.004	Perform automated or on-demand corrective operations on virtualised resources failure

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Assign dedicated resources to a VM for failure **containment** (i.e. propagation avoidance) (see [i.2], Req. 9.2.1) - in conjunction with the hypervisor (see [i.2], Req. 9.2.2) - by defining maximum virtual resources allocated to VMs, e.g. for attack containment (see [i.2], Req. 9.2.3)	**Virtualised resource management**	
	Vim.Vrm.001	Support NFVI resource management within the VIM area of responsibility
	Vim.Vrm.007	Receive the virtualised resource management requests from VNFM and/or NFVO, and conduct the corresponding resource management operations

MANO

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Analysis of load information (from VNFs, hypervisor, …), e.g. determine if scaling or migration or … is needed (see [i.2], Req. 9.4.2) → VNFM, VIM	**VNF-related resource management in indirect mode (if applicable)**	
	Nfvo.VnfRmpbNfvo.001	Request to the VIM the management of virtualised resources needed for VNFs instantiation, scaling and termination
	Nfvo.VnfRmpbNfvo.002	Invoke resource management operations toward the VIM as requested by the VNFM
	VNF scaling	
	Nfvo.VnfS.001	Request expanding the capacity of a VNF instance
	Nfvo.VnfS.002	Request contracting the capacity of a VNF instance
	NS scaling	
	Nfvo.NsS.001	Manage the expansion of a NS instance
	Nfvo.NsS.002	Manage the contraction of a NS instance
	Nfvo.NsS.003	Request to scale a VNF instance as part of the expansion/contraction of a NS instance
	VNF-related resource management in indirect mode	
	Vnfm.VnfRmpbNfvo.001	Request to NFVO the management of virtualised resources needed for VNFs instantiation, scaling and termination
	VNF-related resource management in direct mode	
	Vnfm.VnfRmpbVnfm.001	Request to the VIM the management of virtualised resources needed for VNFs instantiation, scaling and termination
	VNF scaling	
	Vnfm.VnfS.001	Manage the expansion of the capacity of a VNF instance
	Vnfm.VnfS.002	Manage the contraction of the capacity of a VNF instance
	Vnfm.VnfS.003	Manage the scaling out/in of a VNF instance in order to perform expansion/contraction
	Vnfm.VnfS.004	Expand/contract a VNF instance based on a request from the VNF instance or its EM if it exists
	Vnfm.VnfS.005	Expand/contract a VNF instance based on a request from NFVO

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
	Vnfm.VnfS.006	Monitor the state of a VNF instance and trigger its expansion/contraction when certain conditions are met
	Virtualised resource management	
	Vim.Vrm.001	Support NFVI resource management within the VIM area of responsibility
	Vim.Vrm.007	Receive the virtualised resource management requests from VNFM and/or NFVO, and conduct the corresponding resource management operations
	Virtualised resource capacity management	
	Vim.Vrcm.001	Collect and maintain information regarding the capacity of the NFVI which is managed by the VIM
	Virtualised resource performance management	
	Vim.Vrpm.001	Collect performance information related to virtualised resources
	Vim.Vrpm.003	Manage virtualised resource performance in response to the request
	Infrastructure resource performance management	
	Vim.Irpm.001	Collection of performance information related to software and hardware resources within the NFVI

NOTE 10: The resiliency requirement above was identified as non-critical in clause 7.1.

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Support (receive, analyse) **failure notification** at run time and act accordingly (see [i.2], Req. 4.2.5), e.g. data from NFVI (see [i.2], Req. 4.2.6) → NFVO, VNFM, VIM Support of automated fail-over on the NFVI level (see [i.2], Req. 4.2.12) → NFVO, VNFM, VIM	**Virtualised resource fault management**	
	Nfvo.Vrfm.001	Collect fault information related to the virtualised resources assigned to NS(s) managed by the NFVO.
	Nfvo.Vrfm.002	Correlate the virtualised network resource fault information with the impacted NS(s) managed by the NFVO
	Nfvo.Vrfm.003	Request corrective operations on virtualised network resources to VIM in order to perform NS healing
	Virtualisation-related fault management	
	Nfvo.VirFm.001	Request VNF healing to VNFM
	Nfvo.VirFm.002	Collect notifications about alarms on a VNF instance as a consequence of state change in the virtualised resources used by the VNF
	NS fault management	
	Nfvo.NsFm.004	Perform automated or on-demand healing on the NSs it manages
	Nfvo.NsFm.006	Evaluate the impact on NS instance(s) the NFVO manages when NS healing needs to be performed on a component instance (i.e. a VNF or nested NS) shared or not
	Virtualised resource fault management	
	Vnfm.Vrfm.001	Collect fault information related to the virtualised resources assigned to VNF instance(s) which are managed by the VNFM
	Vnfm.Vrfm.002	Correlate virtualised resource fault information with the impacted VNF(C) instance(s) which are managed by the VNFM
	Virtualised resource-related VNF fault management	
	Vnfm.VrVnfFm.005	Request corrective operations on virtualised resources to VIM in order to perform VNF healing
	Virtualisation-related fault management	
	Vnfm.VirFm.001	Perform on-demand VNF healing on the VNF(s) which are managed by the VNFM
	Vnfm.VirFm.002	Perform automated VNF healing on the VNF(s) which are managed by the VNFM
	Virtualised resource fault management	
	Vim.Vrfm.001	Collect fault information related to virtualised resources

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
	Vim.Vrfm.004	Perform automated or on-demand corrective operations on virtualised resources failure
	Infrastructure resource fault management	
	Vim.Irfm.001	Correlate fault information on virtualised resources with fault information related to underlying used software and hardware resources within the NFVI
	Vim.Irfm.002	Collection of fault information related to software and hardware resources within the NFVI

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Provide visibility (into specific classes of hardware failures) to the VNFs (see [i.2], Req. 10.8.2) → NFVO, VNFM, VIM (see note 11) *- send hardware failure notifications (see [i.2], Req. 10.8.3)* → NFVO, VNFM, VIM	**Virtualised resource-related VNF fault management**	
	Vnfm.VrVnfFm.001	Provide notifications of virtualised resource-related fault information on the VNFs which are managed by the VNFM
	Vnfm.VrVnfFm.002	Provide virtualised resource-related fault information on the VNFs which are managed by the VNFM
	Vnfm.VrVnfFm.003	Provide notifications of changes in virtualised resource-related fault information related to the VNFs which are managed by the VNFM
	Vnfm.VrVnfFm.004	Notify about alarms on VNF and any of its VNF Components as a consequence of state changes in the virtualised resources used by the VNF and its VNF Components
	Infrastructure resource fault management	
	Vim.Irfm.001	Correlate fault information on virtualised resources with fault information related to underlying used software and hardware resources within the NFVI
	Vim.Irfm.002	Collection of fault information related to software and hardware resources within the NFVI

NOTE 11: Classes of failures are not considered in [i.3].

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Automatic VNF **re-instantiation** following a failure, e.g. VM failure (see [i.2], Req. 4.2.4) → NFVO, VNFM - within specified time intervals (see [i.2], Req. 7.3.7) - retrieve VNF state information (previously stored) before VNF recreation, e.g. after a failure (see [i.2], Req. 4.2.9) (see note 12)	**NS fault management**	
	Nfvo.NsFm.004	Perform automated or on-demand healing on the NSs it manages
	Virtualisation-related fault management	
	Vnfm.VirFm.001	Perform on-demand VNF healing on the VNF(s) which are managed by the VNFM
	Vnfm.VirFm.002	Perform automated VNF healing on the VNF(s) which are managed by the VNFM
	Virtualised resource fault management	
	Vim.Vrfm.004	Perform automated or on-demand corrective operations on virtualised resources failure

NOTE 12: Concerning 7.3.7 and 4.2.9, the requirements are not addressed as such in [i.3].

Resiliency requirement	ETSI GS NFV-IFA 010 [i.3] requirement	Description
Service **admission control** capability (see [i.2], Req. 7.3.5) → NFVO, VNFM, VIM	**VNF instantiation**	
	Nfvo.VnfI.001	Request the instantiation of a VNF instance
	Nfvo.VnfI.002	Send to the VNFM, as part of the VNF instantiation request, input parameters specific for the VNF instance being instantiated
	VNF scaling	
	Nfvo.VnfS.001	Request expanding the capacity of a VNF instance
	Nfvo.VnfS.002	Request contracting the capacity of a VNF instance
	VNF termination	
	Nfvo.VnfT.001	Request the termination of a VNF instance
	Nfvo.VnfT.002	Check the dependencies between VNF instances before granting the termination of a particular VNF instance
	VNF instantiation	
	Vnfm.VnfI.001	Manage the instantiation of a VNF instance
	Vnfm.VnfI.002	Request VIM to allocate resources for the VNF instance being instantiated
	Vnfm.VnfI.003	Configure deployment specific parameters for the VNF instance being instantiated
	Vnfm.VnfI.004	Store the information of the allocated resources and configured deployment specific parameters for the instantiated VNF
	VNF scaling	
	Vnfm.VnfS.001	Manage the expansion of the capacity of a VNF instance
	Vnfm.VnfS.002	Manage the contraction of the capacity of a VNF instance
	Vnfm.VnfS.003	Manage the scaling out/in of a VNF instance in order to perform expansion/contraction
	Vnfm.VnfS.004	Expand/contract a VNF instance based on a request from the VNF instance or its Element Manager (EM) if it exists
	Vnfm.VnfS.005	Expand/contract a VNF instance based on a request from NFVO
	Vnfm.VnfS.006	Monitor the state of a VNF instance and trigger its expansion/contraction when certain conditions are met
	VNF termination	
	Vnfm.VnfT.001	Capability to terminate a VNF instance
	Virtualised resource management	
	Vim.Vrm.001	NFVI resource management within VIM's area of responsibility
	Vim.Vrm.002	Resource reservation management
	Vim.Vrm.007	Receive the virtualised resource management requests from VNFM and/or NFVO, and conduct the corresponding resource management operations

6.2.3 Unmapped resiliency requirements

MANO

Critical resiliency requirements:

- MANO failure:
  - not affecting existing VNF instances and no change in existing VNFs service level (see [i.2], Req. 10.8.23) → NFVO, VNFM, VIM
  - integrity preserved (e.g. during orchestration) thanks to the atomicity of internal state data change (see [i.2], Req. 10.8.25) → NFVO, VNFM, VIM

- recover the state of the environment by means of restoring persistent storage and audit the environment for determining its state, without interruption to the in service VNF instances (see [i.2], Req. 10.8.24) → NFVO, VNFM, VIM

→ No MANO capability is found to map with these requirements, as they deal with design/deployment choices.

- No **single point of failure** (see [i.2], Req. 4.2.13), e.g. redundant deployment (see [i.2], Req. 10.8.21) such as geo-redundancy (see [i.2], Req. 10.8.22) → NFVO, VNFM, VIM.

→ No MANO capability is found to map with this requirement, as it deals with design/deployment choices.

- Resiliency for **multi-vendor** environment (see [i.2], Req. 4.2.14) → NFVO, VNFM, VIM.

→ The concept of multi-vendor environment is not found in [i.3].

- Support of service availability levels and grades of service (e.g. following the SLA expressed in VNFD, NFVO/VNFM will order VIM to allocate the appropriate resources) (see [i.2], Req. 7.3.1, 7.3.2 and 7.3.3) → NFVO, VNFM, VIM.

→ The service availability level and grade of service concepts are not found in [i.3], although 'service availability level' is introduced in [i.7].

- Transport network (e.g. virtual network) redundancy (see [i.2], Req. 9.5.3) → VIM.

→ No mapping to [i.3] is found, but this is a design/deployment option.

- Support **watchdog** functionality whose characteristics include reliable triggering mechanisms and low latency notifications (see [i.2], Req. 10.8.10) → VNFM, VIM.

→ No mapping is found in [i.3].

- Ensure **service continuity** (at session/service establishment and during service relocation) (see [i.2], Req. 4.2.11) by respecting the acceptable **disruption time** (see [i.2], Req. 5.4.2) defined in the SLA (see [i.2] Req. 5.4.3) → NFVO, VNFM, VIM.

→ Service continuity is not clearly expressed in [i.3], although it is briefly cited in [i.8].

- Appropriate handling of malicious traffic to avoid excessive resource consumption (see [i.2], Req. 7.3.9) → NFVO, VNFM, VIM;

- Support of VM **migration**, e.g. following a hardware, a software failure, or resource shortage (see [i.2], Req. 5.4.1).

→ NFVO, VNFM, VIM (support of VM migration when requested):

- on site and/or offsite (see [i.2], Req. 7.3.8);

- maintaining pre-migration communication between VNFs (see [i.2], Req. 5.4.4);

- the pre-migration process (which may be activated during the initial instantiation of the VNF) includes the availability of VNFD (to retrieve deployment conditions and/or VNF constraints) (see [i.2], Req. 5.4.6).

→ Migration as expressed in Req. 10.8.15 [i.2], is supported, but not the one mentioned above.

- Support of overload control (see [i.2], Req. 9.4.3) → NFVO, VNFM, VIM.

- Enabling to provide a **degraded service** under abnormal conditions (see [i.2], Req. 7.3.4) → NFVO, VNFM, VIM (support of virtualised resource management when requested).

→ No MANO capability is found to map with this requirement.

- VNFs **priority handling** for resource allocation (see [i.2], Req. 5.4.7) → NFVO, VNFM

→ Priority handling is not found in [i.3].

- Handling of fields error **severity** (see [i.2], Req. 9.3.7 and 10.8.8) and failure cause (see [i.2], Req. 9.3.8) received → NFVO, VNFM, VIM.

→ No MANO capability is found to map with this requirement. It is noteworthy that besides VNFM and VIM, EM and OSS may also take action.

Other resiliency requirements:

- **Pre-emption** capability (VNFM) and support of this capability (VIM), e.g. in case of resource shortage (see [i.2], Req. 5.4.8) → VNFM, VIM.

→ There is no pre-emption capability mentioned in [i.3].

- Support of VNF **scaling** → NFVO, VNFM:
  - on site and/or offsite (see [i.2], Req. 7.3.8);
  - within the VNF's limits, limit the number of users in order to minimize the failures impact (see [i.2], Req. 4.2.10);
  - including efficient load distribution/balancing (see [i.2], Req. 5.4.9).;

→ Although scaling is considered in Req. 9.4.2 [i.2], no mapping with these requirements is possible. It is noteworthy that Req. 7.3.8 [i.2] can be supported implicitly, Req. 4.2.10 [i.2] can be classified as a deployment option, while Req. 5.4.9 [i.2] should be accommodated by scaling out/in procedures.

- Failure prediction framework:
  - false alarm filtering to avoid triggering unnecessary prevention procedure for anomaly alarming (see [i.2], Req. 9.3.3) → VNFM, VIM (involved as alarm manager);
  - trend identification, period or seasonal variations, and randomness analysis of data on resource usage to predict unhealthy state (e.g. resource exhausting) (see [i.2], Req. 9.3.4) → VNFM, VIM;
  - diagnose/verify which entity is progressing towards a failure and which VNFs might be affected (see [i.2], Req. 9.3.5) → VNFM, VIM (support of fault management related to virtualised resources).

→ Prediction, filtering, diagnosis are not found in [i.3] - it can be understood that they are part of "corrective operations" (see Nfvo.Vrfm.003, Vnfm.VrVnfFm.005, Vim.Vrfm.004).

- Distributed **fault correlation** processing: avoiding propagating large number of failure notifications to a centralized entity by sending locally correlated reports (see [i.2], Req. 10.8.7) → NFVO, VNFM, VIM.

→ The functionalities Nfvo.Vrfm.002, Vnfm.Vrfm.002 and Vim.Irfm.001 deal with fault correlation in NFVO, VNFM and VIM, but if the resiliency requirement requests for distributed processing within each of the MANO building blocks, then there is no possible mapping.

7 Resiliency mechanisms applied to MANO's critical capabilities

7.1 Introduction

Clause 6 has identified the subset of MANO capabilities which is essential for the dependability of network services once the NFV system is put in operations. The present clause shows a non-exhaustive list of methods and techniques which, when used, can help to reach this overall reliability target. As the MANO capabilities can be summarized as pieces of code deployed in a centralized (e.g. cloud-like) or distributed (e.g. PoP) environment, their behaviour relies on:

(i) the intrinsic nature of the code;

(ii) the program architectural construction;

(iii) the infrastructure on top of which the code is running;

(iv) the way the code is operated (including the daily operations and the periodic maintenance tasks, e.g. software modification).

A first analysis of the code reliability shows that in the software quality, milestones span from financial concerns (e.g. developing code respecting budget and time planning) to numerous technical aspects which include (see [i.9] for the complete list):

- easily maintainable code;
- usability, i.e. programs easy to operate thanks to a well-designed user interface;
- compliance to functional requirements;
- performance efficiency;
- robustness from a *security* standpoint;
- software reliability.

The latest aspect, software reliability, is the probability of failure-free code operation for a specified environment and a specific period of time. The root causes of failures include a combination of non-compliance with good architectural and coding practices. This non-compliance can be detected by measuring the static quality attributes of an application. Assessing the static attributes underlying an application's reliability provides an estimate of the level of business risk and the likelihood of potential application failures and defects the application will experience when placed in operation.

As design defects characterize software (i.e. they can occur without warning and are not function of operational time), traditional techniques such as failing over to a redundant component does not always ensure mitigation of the initial defect since it is more likely the same defect will be present in a redundant software component (compared to a redundant hardware component). Software debug is expected to increase its reliability, although the introduction of new bugs can influence reliability the other way round. Software reliability is also strongly related to the *operational profile*; indeed, although a program still contains plenty of bugs after its release, failures may not appear if the code usage does not activate these faults. Software failures thus result from changes in usage, in operating conditions, or new features added through a software modification.

As faults are the source of unreliability, a traditional way to apprehend the methods used to maximize software reliability is to classify them in the following categories:

- prevent: avoid fault introduction during software design, e.g. proving program correctness using model checking or formal verification;
- remove: detect faults after their introduction in a program generally through software testing - as an example, extensive testing is strongly recommended for the MANO's capabilities identified as critical;
- tolerate: the use of *qualitative* methods helps a software system to behave correctly, even in presence of faults, e.g. redundancy as relaunching a program could be enough to restart it after a software failure;
- forecast: estimate the number of remaining faults in order to predict future failures - this *quantitative* way allows to verify that reliability objectives are reached, often through probabilistic/statistic concepts, i.e. evaluation and measure of software reliability as described in [i.9].

NOTE: [i.9] also provides guidance concerning operations reliability, e.g. software upgrade/update.

7.2 Diversity

This clause provides a synthesis of this important fault-tolerance technique described in [i.9]. As for redundancy, diversity is considered as an essential means for building resiliency. Managed technical approaches aim to encourage or control software diversity, while diversity in software can be synthesized in artificial and automatic techniques.

Managed diversity

Diversity may be introduced intentionally during the design phase, or may be exploited thanks to the existence of different software solutions supporting the same functional requirements:

- Design diversity: the pioneering work has investigated N-version programming and recovery blocks to increase the reliability of embedded systems. These works advocate design and implementation diversity as a means for tolerating faults. Indeed, similar to natural systems, software systems including diverse functions and elements are able to cope with many kinds of unexpected problems and failures.

- Natural diversity: this case covers the existence of different software that provide similar functionalities and emerging spontaneously from software development processes. As an example, the programs that can be customized through several parameters embed a natural mechanism for diversification: two instances of the same program, tuned with different parameters can have different behaviours in terms of robustness and performance.

Automated diversity

This family of techniques emphasizes the absence of human in the loop and is in clear opposition to managed diversity. It can be classified in two categories:

- Randomization: it is about automatic randomization of some aspects of a program, thus producing a diversity of program versions. Diversity occurs in memory, in the operating system, in the bytecode or in the source code but, in all cases, it happens with no human intervention, through random processes which can be static, dynamic or 'unsound'.

- Integrated diversity: this approach is about automatically injecting different forms of diversity at the same time in the same program. In this context, the diversity is stacked or the different forms of diversity are managed with a specific diversity controller.

7.3 Redundancy

This unavoidable technique is obviously part of the set of means used for maximising the resiliency of telecommunication systems, be it hardware or software.

The different redundant configurations of VNFCs have been extensively described in the clauses 6.2.2.2-6.2.2.4 of [i.9]:

- non-redundancy/on-demand redundancy: no pre-established redundant entity to fail to exists in this configuration;

- active/standby redundancy: N active entities are running the application component software and are protected by one or more standby entities (which are not actively providing service);

- active/active redundancy: all the entities are actively running the application component software.

From the whole set of MANO capabilities listed in [i.3], only a small fraction is associated with resiliency whose main target is to maximise the NFV system availability. Indeed, capabilities such as the management of software image, or VNF package, are not concerned with NFV resiliency requirements defined in [i.2]. Such capabilities can benefit from the non-redundancy/on-demand redundancy configuration cited earlier, e.g. the NFVO functionalities related to quota management (Nfvo.Qm.001-Nfvo.Qm .006). It is noteworthy that since there is no pre-instantiated and active redundant entity in place in this scheme, direct state replication for stateful functions is not possible and, therefore, state protection mechanisms require the use of external state repository.

Thanks to the classification of requirements into critical and other categories (see clause 6.1), and the mapping of MANO capabilities vs. the resiliency requirements (see clause 6.2), the remaining MANO capabilities are:

- linked with critical resiliency requirements, e.g. the NFVO functionalities related to virtualised resource fault management (Nfvo.Vrfm.001- Nfvo.Vrfm.003);

- linked with other (i.e. less critical) resiliency requirements, e.g. the NFVO functionalities related to VNF scaling (Nfvo.VnfS.001- Nfvo.VnfS.002).

The first group of capabilities can be deployed using different active/active redundancy schemes:

- 1+1 active-active;
- N+1 active;
- N+M active.

The second group of capabilities can be addressed using an active/standby architecture through different cases:

- 1:1, single active protected by single standby;
- N:1, N actives protected by single standby;
- N:M, N actives protected by M standbys.

7.4 Availability monitoring

7.4.1 MANO components monitoring

The present clause deals with the liveness of any "element" used for management and orchestration in NFV systems. The mechanisms discussed in this clause are thus applicable to monitor the availability of an "element" such as:

(i) a capability within a MANO component, or

(ii) a MANO component itself (i.e. NFVO, VNFM, VIM).

Heartbeat

Heartbeat is a mechanism for indicating aliveness. As applied to the MANO components (i.e. NFVO, VNFM, VIM), each of them needs to generate a heartbeat used to monitor the internal capabilities of the MANO component by its HA manager. The monitoring with heartbeat could be a part of the services provided by the MANO components. The duration between heartbeats is dependent on the MANO component and the availability requirement of the monitored capabilities.

NOTE: The introduction of "HA manager" does not intend to create a new functional block; it is used to describe its functionality, e.g. the HA manager role may be played by the "MANO monitor" function as described in [i.16].

The second level of heartbeat is between the HA managers of a cluster made of MANO components for monitoring the functioning of one of its parts.

Elements of a heartbeat

A heartbeat message could be the following:

- unique sequence number which is a free running counter with which the receiver can obtain the passing of time information;
- identifier for the monitored element (capability or component) for the service.

Heartbeat is used by an HA Manager to ascertain over time if the monitored elements (capabilities or components) are alive. The heartbeat would be generated by each of the MANO components with a time duration dependent on the monitored element. The HA Manager of the MANO component (either within the MANO component or in its redundant pair) would determine the liveness of an element if no heartbeat message is received within the agreed time interval from the monitored element.

Figure 7.1 describes a possible implementation scenario of how the heartbeat communication between the MANO components and their redundant counterpart could happen with the possible resiliency pathways between the various MANO components highlighted. VIM, VNFM and NFVO services generate heartbeats whose period is configured at the start of the service and its redundant pair.

- If the VNFM heartbeat stops, the redundant (Active or Standby) VNFM starts servicing the VNFs. The NFVO comes to know of this change state and manages the service orchestration through the Standby VNFM.

- If the VIM heartbeat stops, the redundant (Active or Standby) VIM starts servicing the NFVI. The NFVO comes to know of this change state and manages the resource orchestration through the Standby VIM.
- If the NFVO services heartbeat stops, the appropriate redundant (Active or Standby) NFVO service starts servicing the VNFM and the VIM for resource and service orchestration.

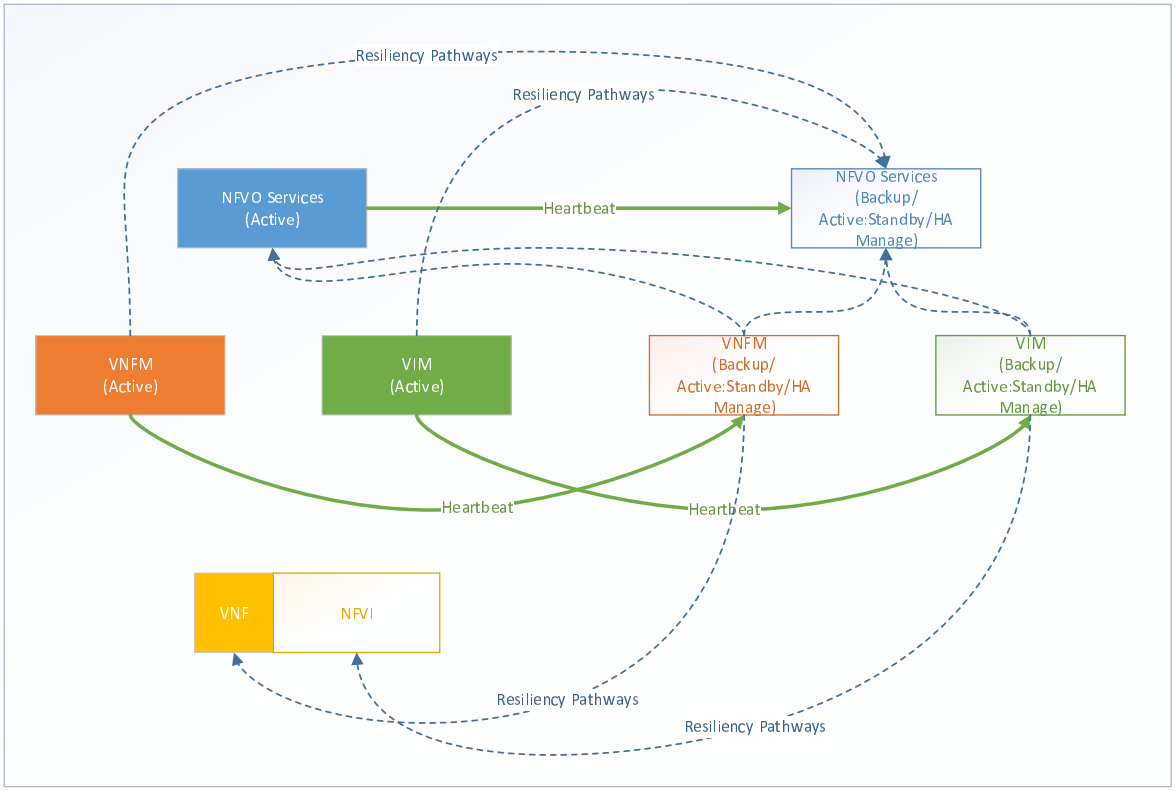

Figure 7.1: Heartbeat management

Security considerations

It is noteworthy that two security issues have to be taken into account:

- The heartbeat needs to include an integrity check in the form of a signed hash. This is to avoid spoofing state information pointer in the heartbeat.
- In order to avoid DoS attacks through heartbeat flooding, anti-replay measures need to be in place in the heartbeat generation portion of the MANO components.

Polling

Polling is a mechanism for determining aliveness. As applied to the critical functions in MANO components (i.e. NFVO, VNFM, VIM), HA manager in the MANO component polls periodically the monitored elements to determine their aliveness. In essence it is a pull model as compared to the push model of the Heartbeat.

7.4.2 Virtual links monitoring

In the previous clause, "heartbeat" and "polling" mechanisms are proposed for monitoring the availability:

(i) of a capability within a MANO component; or

(ii) of a MANO component.

However, the availability of internal virtual links within a MANO component and the availability of external links between MANO components also need to be monitored to identify the failure localization correctly, whether the loss of the "heartbeat" or "polling" messages is related to the failure of a link, of a capability or of a MANO component. In addition, network partition (e.g. failure condition based on servers not communicating and synchronizing their data to each other) might happen and cause the split-brain (e.g. data or availability inconsistencies originating from the maintenance of two separate data sets with overlap in scope), rendering the heartbeat or polling mechanisms useless. It is thus necessary to take network partition into consideration.

Supervising the availability of a virtual link is performed by monitoring its corresponding physical L2/L3 links. For this purpose, some kind of link failure detection mechanisms, e.g. OSPF LSA (link state advertisement), BFD (bidirectional forwarding detection), etc., might need to be implemented in the physical network. However, the failure detection and recovery of the lower level links have to be fast enough so that the upper layer failure detection mechanisms using "heartbeat" or "polling" can take recovery actions correctly. In addition, the detection and recovery of the network partition have to inform the HA function which manages the capability or MANO component affected by the network partition.

8 Recommendations

The recommendations proposed in this clause are based on the mapping results of the clause 6.2.3. They are related to the three functional components of NFV-MANO: NFVO, VNFM and VIM.

These recommendations propose to specify requirements on the NFVO, VNFM, and VIM in order to extend these functional blocks for supporting resiliency. Table 8.1 (resp. 8.2) displays the critical (resp. other) resiliency requirements for NFV-MANO components.

NOTE 1: The "Comment" column of tables 8.1 and 8.2 are provided for informative purposes, and is not part of the recommendations.

NOTE 2:

- *MANO-services* refer to the services provided by one, or more, of the MANO components (NFVO, VNFM, VIM).
- *NFV-NS services* refer to the services provided by the NS under study (e.g. NS IMS composed of the VNFs I/S-CSCF, P-CSCF, etc.).

Table 8.1: Critical resiliency requirements for NFV-MANO components

Number	Recommendation	Comment
It is recommended that a requirement set be specified on the		
Crrr.001	NFVO, VNFM and VIM functional blocks to support resiliency of VNFs, infrastructure, and MANO-services for multi-vendor environment	• NFV resiliency has to be designed from a system's point of view. The requirement 4.2.14 [i.2] has to be interpreted as "The resiliency mechanisms applicable to multiple NFV entities shall be designed for a multi-vendor environment". For example, failure recovery involves multiple entities, and needs to be designed for multi-vendor environment (priority or severity for failure reporting and recovery involving multiple entities, …).
Crrr.002	NFVO, VNFM and VIM functional blocks to support service availability levels (SAL) and grades of service, such service referencing to both MANO-services and NFV-NS services	• Support of SAL requirements in the VNFD is specified in [i.7] (VNF_PACK.META.017). • SAL can already be specified using VLD (NST_VLDF004 and attributes of the VirtualLinkDf information elements) [i.14].
Crrr.003	NFVO, VNFM and VIM functional blocks to support watchdog functionality to detect VNFs, infrastructure, or MANO-services unavailability	• Watchdog is used to detect the unavailability of a (critical) functionality.
Crrr.004	NFVO, VNFM and VIM functional blocks to support service continuity [i.1]	• Different services have different tolerable disruption times. Service continuity needs to be ensured during restoration following a failure, scaling and software (VNF, MANO, NFVI) modification.
Crrr.005	NFVO, VNFM and VIM functional blocks to support VNF migration	• When running maintenance operations or failures occur for NS elements (VNFs, …), migration of these elements may be required depending on the service continuity needed (see Crrr.004's comment). • VM migration might cause an interruption of the services carried on NS.
Crrr.006	NFVO, VIM and VNFM functional blocks to support unavailability avoidance of NFV-NS services in case of intentional actions (e.g. attacks)	• As malicious traffic is a possible source of congestion (see [i.2], Req. 7.3.9), this item is expected to be fulfilled by the security orchestrator as it is planned in [i.15].
Crrr.007	NFVO, VNFM and VIM to support the feature of degrading NFV-NS services	• MANO should support the concept of degraded service. In some circumstances (e.g. lack of resource following a disaster), resources allocated to VNFs with a low SAL could be revoked.
Crrr.008	NFVO and VNFM functional blocks to support VNF priority handling	• This concept could be inferred from the SAL (see Crrr.002 and Crrr.007).
Crrr.009	VNFM and VIM functional blocks to support error severity in VNFs, infrastructure, and MANO-services	• Different error severity levels are exploited to optimize the reaction to failures and the restoration process.
Crrr.010	NFVO, VNFM and VIM functional blocks to support overload control for VNFs, infrastructure, and MANO-services	• Overload control is to be understood as prevention/detection of overload conditions and protection of the VNFs, infrastructure, and MANO functional blocks elements from overload. • Performance monitoring of NFV-MANO is under study in [i.16] for overload situations.

Table 8.2: Other resiliency requirements for NFV-MANO components

Number	Recommendation	Comment
It is recommended that a requirement set be specified on the		
Orrr.001	VNFM and VIM functional blocks to support VNF pre-emption	• This concept could be inferred from the SAL (see Crrr.002 and Crrr.007).
Orrr.002	VNFM and VIM functional blocks to support failure prediction for VNFs, infrastructure, and MANO-services	• Failure prediction allows forecasting imminent failures while the NS is provided.
Orrr.003	NFVO, VNFM and VIM functional blocks to support the correlation of faults within the NFV system	• Distributed fault correlation processing helps to enhance reactions to incidents, e.g. through root cause analysis.

9 Security considerations

Some intentional actions such as DDoS may impact network service availability directly. This reinforces the need to specify appropriate security countermeasures, which is outside the scope of the present document.

Annex A:
MANO:
from initial needs to current capabilities

A.1 Brief history of NFV

Convinced that the communications industry should benefit from the virtualisation and cloud-based technologies which are well-oiled in the IT industry, and provide agility and higher efficiency, several operators have launched the NFV movement through a White Paper published in 2012 [i.10]. The core concept is to decouple network functions software from the underlying hardware and use industry standard high volume servers, switches and storage devices (COTS), creating thus an open and standardised infrastructure for a multi vendors environment. This new eco-system targets the emergence of more innovation through its openness to new actors, e.g. pure software entrants, small players. Leveraging the technologies mentioned above, it is expected that this new framework provides process automation and self-provisioning features to reduce operational load.

Apart from the NFV benefits enumerated in this White Paper (e.g. costs, power consumption and TTM reduction, network optimization, rapid innovation), some considerations, not yet designated as requirements at that time, were expressed:

- co-existence of legacy, i.e. envisioning hybrid networks composed of classical physical and virtual network appliances;
- compatibility with existing platforms (e.g. OSS, BSS) for re-usability purposes;
- support of multi-tenancy, e.g. to enable the share of resources across different administrative domains.

Non-functional aspects include:

(i) the desire to minimize the performance degradation (e.g. latency, throughput) due to the use of COTS;

(ii) the expectation of increased availability, e.g. thanks to on-demand VNFs creation facilitating healing needs, while ensuring the same level of security as for physical appliances.

Three main areas are thus present and unavoidable in the NFV's picture:

- a NFV infrastructure used to run the network services;
- dedicated VNFs that build software functions providing these network services;
- entities for management and orchestration.

The mechanisms implemented by this entity include aspects of life cycle management such as:

- network services creation, termination or update;
- automatic instantiation of VNFs;
- resources management, e.g. by assigning VNFs to the correct CPU core, memory and interfaces;
- dynamic VNFs scaling;
- re-initialisation of failed VNFs;
- VNF migration for maintenance or restoration purposes;
- inventory, usage, performance, resiliency and management.

NOTE: Although their contents are not included in this clause, the reader is invited to refer to the two additional White Papers [i.11] and [i.12] which were published further to [i.10].

A.2 MANO architectural framework

A.2.1 MANO main functionality

Following the decoupling of network functions software from the hardware infrastructure as stated in clause A.1, an entity, called MANO in what follows, is needed to orchestrate and manage:

(i) the software functions used to provide the network services (NSs); and

(ii) the infrastructure offering virtual resources for running these functions.

Network service lifecycle management can be viewed as the first (or top level) functionality for MANO [i.13]: NSs are instantiated/created after their on-boarding. It is noteworthy that the deployment and operational behaviour of NSs are described in deployment templates. During the life time of a NS, scaling it (i.e. increasing or reducing its capacity), updating it (i.e. changing its configuration such as the inter-VNF connectivity), and terminating it (e.g. by requesting the termination of its constituent VNF instances and freeing the resources) constitute the fundamental features. In order to link the different VNFs of a NS, a graph is needed: VNFFG or VNF forwarding graph; the management of links according to the graph (e.g. their creation, update, deletion), together with NS KPIs monitoring, are also part of the first functionality.

In addition to the management of NSs, taking care of the VNFs constitutes the second functionality of MANO. The operations it is in charge are similar to the ones described for NSs, i.e. on-boarding (including deployment templates to describe attributes and requirements to realize VNFs), instantiation, scaling, update, termination, and KPIs monitoring. It is noteworthy that at instantiation, NFVI resources are assigned to a VNF based on the requirements captured in the deployment template but also taking into consideration specific requirements, constraints, and policies (e.g. scaling policies) which were pre-provisioned or are accompanying the instantiation request. Furthermore, two types of VNFs scaling are explicitly considered:

(i) up/down (i.e. adding/removing resources such as CPU), or

(ii) out/in (i.e. adding/removing virtualised resources such as VMs).

The traditional fault, configuration, accounting, performance, and security management tasks (FCAPS) needed for the VNFs are outside the scope of MANO as these tasks are not modified due to virtualisation. They are assumed to be handled in the same way as for the non-virtualised versions of the network functions over interfaces defined by other SDOs.

MANO is finally expected to collect information from the hardware (compute, storage, and networking) and software (e.g. hypervisors) components which compose the infrastructure resources where VNFs are deployed. The set of features of this third management functionality includes the management of the (virtualised) resources and their availability/allocation/release, together with their fault/performance management. It is noteworthy that NFV provides an enormous advantage for allowing dynamic allocation of additional resources when needed, e.g. during load increase.

In addition, fault and performance management functionalities used, e.g. for SLA management purposes, cover the lifecycle of any NS instance or VNF instance. From several sources of faults, i.e. physical infrastructure (compute, storage, and networking), virtualised infrastructure (e.g. VM-related faults), and VNF instance related faults, different fault and performance related tasks are processed:

- measurement;
- notification;
- correlation/root cause analysis;
- resolution/corrective actions;
- actions triggering within the NFV framework or outside it, e.g. OSS.

Indeed, external and legacy entities such as operations and business support systems (OSS/BSS) need to be considered in conjunction with the functional components of the NFV framework, as the management of end-to-end services requires convergence on a common approach to presenting management services and information from both legacy network systems and NFV based ones. Common interface operations are also required for covering request/response patterns, as well as the behaviour and sequencing of those requests/responses for FCAPS management. Concerning information representation in end-to-end applications, common data types and common vocabularies are needed, e.g. specification of fault codes and location codes used across multiple resources. Automation, a key challenge, may be supported by an aligned approach between the OSS/BSS and the MANO through a dynamic policy-based management. Self-service operations at the service and product level need to respond with the speed and agility required by changing business needs.

The last item which is worth mentioning is the concept of administrative domain. Indeed, within NFV, the share of both physical and virtualised resources allow to define roles which may be separated or merged. An NFV system can thus be provided by a single or different organizations. Two types of administrative domains are proposed: infrastructure domain (ID) and tenant domain (TD). The first can be defined by criteria such as the type of resource (compute, storage, networking), while the second one can be viewed as, e.g. the type of network service. A TD may use infrastructure in a single ID or in multiple IDs. While residing in the TD, VNFs and NSs need NFVI resources from the IDs in order to materialize. Therefore, a TD exploits resources from one or more IDs using the ID orchestration functionality. Typical management tasks within a TD are VNF FCAPS, while tasks requiring the involvement of both the TD and the ID are VNFs on-boarding, instantiation and scaling.

A.2.2 MANO key components

Following the functionalities summarized in clause A.2.1, three main components of NFV-MANO are proposed: NFV Orchestrator (NFVO), VNF Manager (VNFM), and Virtualised Infrastructure Manager (VIM). An architectural framework based on these components shows how VNFs are hosted and operated in an NFV environment (figure A.1).

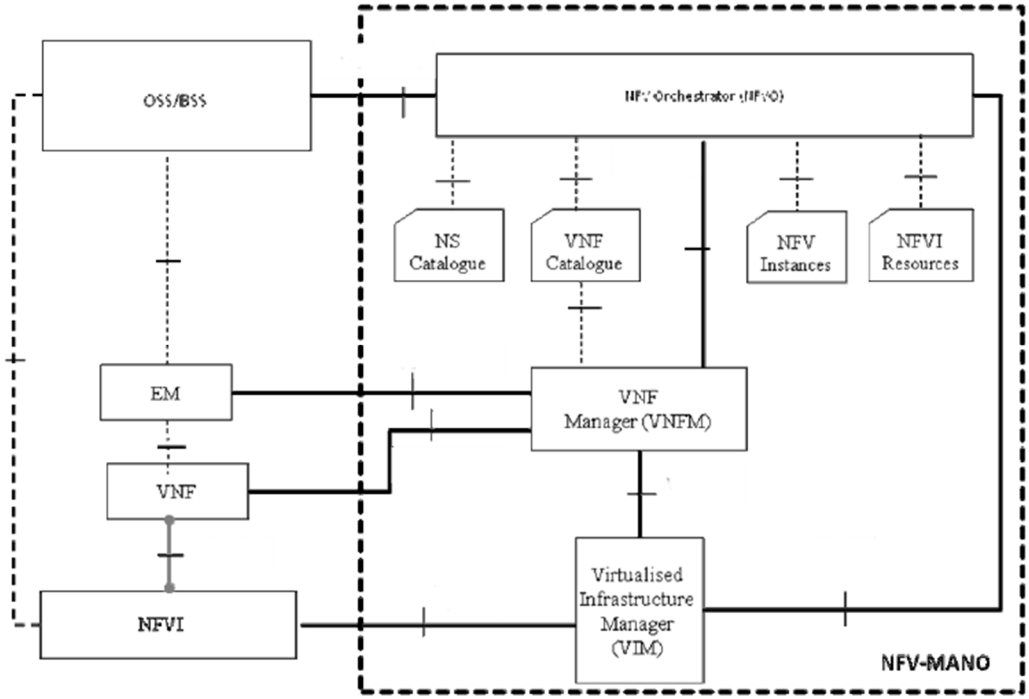

Figure A.1: NFV-MANO architectural framework

NFVO

This component is mainly dedicated to NSs lifecycle management. Orchestration of NFVI resources also falls within its responsibilities. Concerning the former, on-boarding NSs and their related VNFs packages (including their deployment templates), followed by NSs instantiation permits to launch the NSs. During operations, the NFVO's actions include:

(i) scaling and updating NSs;

(ii) collecting performance/resiliency measurements from the NSs;

(iii) managing the NS instances topology and automation (e.g. through triggering actions governed by policies such as affinity/anti-affinity and/or regulatory rules).

The other NFVO field of responsibility starts with the validation and authorization of NFVI resource requests from the VNFM(s), within one NFVI or across multiples ones. The NFVO manages the distribution, reservation and allocation of NFVI resources to NS instances, and VNF instances (in the case of VNF-related resource management in indirect mode). Policy management and enforcement for these two types of instances (e.g. placement optimization based on geography rules or resource usage), and collection of usage information for NFVI resources are the other NFVO's duties.

The functions that the NFVO is expected to provide correspond to the list of requirements specified in clause 6 of [i.3].

VNFM

Just as the NFVO takes care of NSs and their instances, the VNFM deals with VNFs and their instances. VNF packages contain all of the files and meta-data descriptors required to validate and instantiate VNFs. The tasks towards VNF instances, expected from a VNFM, are: instantiation, configuration, update, upgrade, scaling (e.g. out/in), performance and resiliency measurement/detection, healing (e.g. following faults analysis), security related measures (e.g. integrity management), termination.

The VNFs instantiation is done based on the VNF descriptors (VNFDs) which come along with VNF packages. It is noteworthy that VNFs are defined as formed of VNF components (VNFC) instances running separately on distinct containers such as VMs. These descriptors, defining the attributes and requirements needed to instantiate VNFs, are deployment templates containing, e.g. connectivity, interface and KPIs requirements used to establish virtual links within the NFVI between the VNFC instances (or between a VNF instance and the endpoint interface to the other network functions). NFVI resources are assigned to a VNF based on requirements captured in its VNFD, constraints and policies which are pre-provisioned, or are accompanying the request for instantiation. To describe specific deployments of a VNF, Deployment Flavour(s) is(are) described in the VNFD. A given DF can be associated with one or more instantiation levels, each level corresponding to a given level of resources to be instantiated within this DF in term of the number of VNFC instances to be created from each virtualisation deployment unit (VDU). A VDU describes the resources (computation, memory, network bandwidth, …) required to deploy a VNFC instance in a certain environment.

The functions that the VNFM is expected to provide correspond to the list of requirements specified in clause 7 of [i.3].

VIM

With its southbound interfaces connected to hypervisors and network controllers, it is used to control and manage the NFVI through tasks such as:

- orchestration of the allocation/upgrade/release of NFVI resources;
- association of virtualised resources to the physical compute, storage, networking resources;
- support of Network Forwarding Paths (create, query, update, delete), e.g. by creating/maintaining virtual links, virtual networks, sub-nets, and ports, as well as the management of security group policies to ensure network/traffic access control;
- management of related information of NFVI hardware resources and software resources (e.g. hypervisors), and discovery of the capabilities and features (e.g. related to usage optimization) of such resources;
- collection of performance and fault information of hardware, software (e.g. hypervisors), and virtualised (e.g. VMs) resources;
- management of software images.

The functions that the VIM is expected to provide correspond to the list of requirements specified in clause 8 of [i.3].

Repositories

Besides the three main components described above, the NFV-MANO shown in figure A.1 includes four repositories: NS Catalogue, VNF Catalogue, NFV Instances, and NFVI Resources. The NS Catalogue contains the on-boarded NSs, and supports the NFVO to create and manage NS deployment templates: VNF Forwarding Graph Descriptor (VNFFGD), NS Descriptor (NSD), Virtual Link Descriptor (VLD). As for the VNF Catalogue, it contains all the on-boarded VNF packages with their contents (VNFD, software images, …).

Records of VNF instances and NS instances are held in the NFV Instances repository. A VNF, resp. NS, instance is represented by a VNF Record (VNFR), resp. NS record (NSR). The records are updated during the instances lifecycle. Virtual link records (VLRs) and VNFFG records (VNFFGRs) are also part of this repository. Finally, the NFVI Resources repository hosts data about available/reserved/allocated NFVI resources as abstracted by the VIM, allowing their tracking against NS/VNF instances associated with them.

NOTE: Defined in [i.13], these repositories are not explicitly considered as such by the IFA WG. Nevertheless, a subset of the information contained in these repositories is reflected on the interfaces currently specified by the IFA WG.

EM and controllers

Finally, two points worth mentioning are the Element Manager (EM) and the network controllers although they are both functional blocks outside the MANO scope. The former, shown in figure A.1, is a legacy functionality responsible for VNF FCAPS tasks. The latter, not displayed in the figure as it is optional in the NFV framework, may be seen as an abstraction layer. Such controllers (e.g. SDN) allow to establish connectivity within an administrative domain, thus producing virtual networks (VNs), a service provided by the NFVI. At its lowest layer, a network controller manages devices, while at its highest one, it provides connectivity services to applications and abstraction of the underlying resources.

Annex B: Resiliency requirements issued from ETSI GS NFV-REL 001

Reference Number	Details
[Req.4.2.1]	The Virtualised Network Function (VNF) needs to ensure the availability of its part of the end-to-end service, just as in the case of a non-virtualised NF.
[Req.4.2.2]	The VNF designer needs to be able to define the requirements of the Network Function Virtualisation Infrastructure (NFVI), such as geo-redundancy requirements, resiliency requirements, etc. in the Network Service Descriptor (NSD) and VNF Descriptor (VNFD) passed to the NFV Management and Orchestration (NFV-MANO) function.
[Req.4.2.3]	The NSD and VNFD need to provide capabilities to define resiliency requirements.
[Req.4.2.4]	The NFV-MANO function shall provide the necessary mechanisms to recreate VNF automatically after a failure, such as a Virtual Machine (VM) failure.
[Req.4.2.5]	The NFV-MANO function shall support failure notification mechanisms at run time. The VNF can optionally request notification of certain types of failures, and NFV-MANO need to support such a notification mechanism.
[Req.4.2.6]	Failures in the NFVI shall be handled (i.e. detection and remediation) in the NFVI layer or the NFV-MANO (e.g. hardware failure, loss of connectivity, etc.).
[Req.4.2.7]	The NFVI shall provide the necessary functionality to enable high availability at the VNF level, such as failure notification and remediation.
[Req.4.2.8]	NFV frameworks shall ensure that not all services need to be "built to the peak", but Service Level Agreements (SLAs) can be defined and applied according to given resiliency classes.
[Req.4.2.9]	Storage and transfer of state information need to be provided by the NFVI, where the VNF defines the information to be stored and the NFVI provides the respective object store.
[Req.4.2.10]	The NFV-MANO functions need to support capacity limitations per instance as part of the deployment instructions of a VNF.
[Req.4.2.11]	In addition to the normal mode of service execution, service continuity shall be ensured in two situations, namely at session/service establishment and during relocation of a service.
[Req.4.2.12]	On the NFVI level, there should be an automated fail-over in the case of for example compute, memory, storage or connectivity failures.
[Req.4.2.13]	There is an overall requirement that a NFV framework shall not contain a single point of failure with the potential to endanger service continuity.
[Req.4.2.14]	All resiliency mechanisms shall be designed for a multi-vendor environment, where for example the NFVI, NFV-MANO, and VNFs may be supplied by different vendors.
[Req.4.2.15]	Resiliency related information shall always be explicitly specified and communicated using the reference interfaces (including policies/templates) of the NFV framework.
[Req.5.4.1]	When an anomaly event, which causes hardware/software failure or resource shortage/outage, occurs, the corresponding VNF should be able to be migrated (relocated and restored) with preserving its configuration (e.g. IP address or MAC address) in order to provide service continuity.
[Req.5.4.2]	When an anomaly event occurs and the corresponding VNF needs to be migrated, the acceptable disruption time to recover should be taken into consideration (e.g. by NFV-MANO).
[Req.5.4.3]	In order to provide seamless service continuity, the disruption time shall be within the range of the acceptable value defined in the SLA.
[Req.5.4.4]	When a VNF is migrated to another VM or hardware, the communication between the VNF and others shall be maintained regardless of its location.
[Req.5.4.5]	Deployment conditions and/or constraints (e.g. maximum acceptable end-to-end propagation delay) for resiliency shall be measurable and should be described in the VNFD.
[Req.5.4.6]	When a VNF needs to be migrated to another VM or hardware due to anomaly event, the NFVMANO should be able to access the VNFD to retrieve the deployment conditions and/or constraints of the VNF.
[Req.5.4.7]	NFV-MANO should be able to handle priorities of the VNFs in order to mitigate conflicts in resource allocation.
[Req.5.4.8]	Permission to pre-empt a VNF should be described in the SLA in order to be able to restore a higher priority VNF even by suspending a running but lower priority one when there are not enough resources for both.
[Req.5.4.9]	Replication of a VNF and distribution of the load to those VNFs should be supported.
[Req.5.4.10]	VIM should be able to monitor the used and available resources of the hardware infrastructure and to balance the resource usage among VNFs as appropriate.
[Req.5.4.11]	The level (or class) of resiliency and High Availability (HA) for a VNF should be possible to be described in the VNFD, policies, templates, etc.
[Req.7.3.1]	The NFVI and NFV-MANO shall support multiple levels of service availability.

Reference Number	Details
[Req.7.3.2]	Within each service availability level, the NFVI and NFV-MANO shall support multiple grades of service depending on the service (voice, video, web-browsing, etc.)
[Req.7.3.3]	The NFVI and NFV-MANO shall support options for including various service types and possible grades within a service availability level depending on the SLA between a service provider and customer.
[Req.7.3.4]	It shall be possible to continue to provide the service with reduced or limited capability under abnormal network conditions (e.g. a video call downgraded to voice call with still images).
[Req.7.3.5]	The VNFs shall implement service session indicators and packet markings ([1], [2], [3], [4]) that enable service admission control and restoration functions as applicable to the services and the NFVI & NFV-MANO shall support the capabilities to allocate resources appropriately.
[Req.7.3.6]	As the service's availability is related to its failure gravity/severity, i.e. a product of (probability of failure x impact of failure), both dimensions should be regarded to optimize cost of Service Availability.
[Req.7.3.7]	Under failure conditions, VNFs should support service recovery within time intervals specified in table 4 (clause 7.4.2 of [5]).
[Req.7.3.8]	Under failure or excessive load conditions, it shall be possible to support migrating or scaling out the VNFs onsite (on the same or different VNF infrastructure) and/or offsite. The approach shall depend on the type of failure and support available for other failure recovery mechanisms (e.g. VNF redundancy).
[Req.7.3.9]	If a congestion situation is caused by malicious traffic (e.g. DDoS attack), it should be possible to identify such traffic and take appropriate measures to maintain service availability, while at the same time avoiding the consumption of excessive resources.
[Req.7.4.1]	The NFV framework should include a network Service Availability calculation function which is based on key system operational indicators to be available from VNF and NFVI, e.g. service interruption alarm, hardware failure notification, etc.
[Req.9.2.1]	VIM may assign dedicated resources (e.g. CPU core, memory, disk storage and IO) to a VM for containment of VM failures.
[Req.9.2.2]	The hypervisor should implement flexible resource allocation mechanisms to enable the VIM to set policies to ensure failure isolation under a variety of configurations and workloads.
[Req.9.2.3]	The configured shared resource policy in the VDU should define the maximum and minimum amount of resources for each VM to help containment of a VM failure (including containment of the effect of the security attack), in order to prevent all shared virtual resources (e.g. network IO and disk IO resources) from being consumed totally by a single VM. When multiple VMs share the same virtual resources, the sum of the maximum resources assigned to those VMs shall not be more than 100 % of the available resources.
[Req.9.3.1]	The real-time resource usage such as the disk usage, CPU Load, memory usage, network IO and virtual IO usage and their loss rate, and available vCPUs, virtual memory, etc. shall be provided to VIM at configurable intervals by entities of infrastructure. It should also be possible to configure the thresholds or watermarks for the event notification instead of configuring the reporting interval.
[Req.9.3.2]	Each entity of infrastructure shall provide the open interfaces for communicating the performance and the consumption resource and for allowing polling of its working state and resource usage.
[Req.9.3.3]	The failure prediction framework should include the functionality of the false alarm filtering to avoid triggering unnecessary prevention procedure for anomaly alarming.
[Req.9.3.4]	The failure prediction framework should include trend identification, period or seasonal variations, and randomness analysis of the collected data on resource usage (e.g. memory, file descriptors, sockets, database connections) to predict the progression of the operated NFV system to an unhealthy state, e.g. resource exhausting.
[Req.9.3.5]	The failure prediction framework should be able to diagnose or verify which entity is suspected to be progressing towards a failure and which VNFs might be affected due to the predicted anomaly.
[Req.9.3.6]	The entities of VNF and its supported infrastructure should have their own self-diagnostic functionality in order to provide their health information to the VIM.
[Req.9.3.7]	The log report associated with a NFVI resource failure or an error detected by a hardware component, software module, hypervisor, VM, or the network should include the error severity.
[Req.9.3.8]	The log report should include an indication of the failure cause.
[Req.9.4.1]	The resource (CPU, memory, and IO) load of a VM and server (hosting hypervisor) should not stay in a state in which one or more internal processes or messaging has been slowing down, stuck or dropped (how to detect this state is an implementation issue).
[Req.9.4.2]	The overload control in VNF should be implemented in a virtual environment even though elastic resource management could be applied for capacity scaling. The load in guest OS might not the unique parameter for indicating VNF load situation because the performance and load in hypervisor might have an impact on the VNF performance as well.

Reference Number	Details
[Req.9.4.3]	The recommended guidelines and requirements of [6] for the overload control should be considered in the implementation of overload control.
[Req.9.5.1]	The VMs that host the VNFCs implementing the same functionality should be deployed following anti-affinity policies to prevent a single point of failure (additional optimization may be needed in making the geographical distribution decision with the consideration for factors such as state synchronization, legal constraints, etc.).
[Req.9.5.2]	The VNFs with the same functionality should be deployed in independent NFVI fault domains to prevent a single point of failure for the VNF. In order to support disaster recovery for a certain critical functionality, the NFVI resources needed by the VNF should be located in different geographic locations; therefore, the implementation of NFV should allow a geographically redundant deployment.
[Req.9.5.3]	The transport network including the virtual network should provide alternative paths for accessing the same VNF instance to prevent a single point of failure.
[Req.10.8.1]	In an NFV context, it is the responsibility of the hypervisor (or the underlying Host together with its hypervisor) to perform the hardware failure detection which may have been previously performed by the NF itself (or the NF's underlying OS + middleware). The intent is to provide parity with the level of failure detection previously performed on the NF prior to virtualisation.
[Req.10.8.2]	VNFs may require visibility into specific classes of hardware failures. Such visibility may enable them to achieve committed SLAs by reacting to failures more quickly through existing legacy mechanisms than via other means. The NFV-MANO system should enable such visibility via the generation of associated virtual failures.
[Req.10.8.3]	Mechanisms such as VNFD or policies shall be supported to be used by a VNF to register for requesting specific hardware failure notifications from the NFV-MANO when they are detected.
[Req.10.8.4]	The hypervisor (or the underlying host together with its hypervisor) shall report all failures detected to the VIM for subsequent processing, decision making, and/or logging purposes.
[Req.10.8.5]	NFV-MANO functions shall monitor the resources provided to a VNF and shall, in case of a failure in NFVI, take the necessary actions to ensure the continuation of the affected service. The availability requirements stated by the VNF in the VNFD shall be taken into account.
[Req.10.8.6]	In the presence of one or more failures in a system, these failures should be identified first locally at each layer and then across subsystems.
[Req.10.8.7]	Fault correlation processing shall be kept distributed as much as possible and avoid propagating large number of failure notifications to a centralized entity by sending locally correlated reports only, to avoid bottlenecks in the system.
[Req.10.8.8]	The fault correlation function should classify and categorize failure reports in the system, e.g. according to global severity.
[Req.10.8.9]	Correlated failure reports should be communicated via a standard failure reporting mechanism (e.g. API) using a common data model to other layers within the system and/or to external correlation engine.
[Req.10.8.10]	Watchdog functionality should be available in a virtualised network environment. Its key characteristics include the reliability of the trigger mechanism and low latency associated with notification.
[Req.10.8.11]	The VIM shall check the NVF infrastructure actively to detect failures even if a component is currently unused.
[Req.10.8.12]	The VNF shall detect and correct application failures within its own context without relying on external entities in the NFV end-to-end architecture.
[Req.10.8.13]	The VNF shall inform the VNFM in the event that it cannot remediate a failure that results in a degradation of the overall capacity of the VNF.
[Req.10.8.14]	The VNFM shall request additional resources in the event that the VNF encounters a failure that results in a degradation of processing capacity of the VNF.
[Req.10.8.15]	The NVFI layer shall provide indication of hardware and environmental events to the VIM for the purposes of VIM proactively migrating VNFs away from the faulty hardware. For examples of the types of events and the associated metrics, see clause 7.4 ("Hardware Resource Metrics of the Compute Domain") of [7].
[Req.10.8.16]	The NFV-MANO architectural framework shall take corrective action in the event a failure is reported from the NFVI layer. This includes actions such as: - VNF migration for VNFs that provide support for live migration. - VNF capacity reduction in the event that switching capacity has been diminished. - When applicable, removing the failing hardware entity from service and flagging it as faulty and unavailable for consideration for instantiating VNF services.

Reference Number	Details
[Req.10.8.17]	The VNFM may provide an interface that allows VNFs to register for health checking. The registration is application dependent and not explicitly required by the specification.
[Req.10.8.18]	If a VNF failure is detected, the NFVO may also take corrective actions on other VNFs in the same service chain.
[Req.10.8.19]	The NFV-MANO should support a means (e.g. via VNFD, policy) to indicate whether or not a VNF supports an externally initiated health check mechanism, together with the parameters necessary to configure such a health check (e.g. invocation method, execution frequency, response timeout, recovery action).
[Req.10.8.20]	The VNFM should support the ability to restart a VNF that has exceeded its health check response timeout.
[Req.10.8.21]	The NFV-MANO system shall support being deployed in a redundant manner that eliminates any single point of failure.
[Req.10.8.22]	The NFV-MANO system should support being deployed in a geographically distributed configuration to protect against site failures.
[Req.10.8.23]	A failure at the NFV-MANO system shall not affect any existing VNF instances. Any outstanding requests from the VNFs towards the NFV-MANO system shall time out gracefully and not change the existing service level of the VNFs.
[Req.10.8.24]	The NFV-MANO system shall be able to recover the state of the environment by means of restoring persistent storage, as well as auditing the environment to determine the true state of the environment. This shall complete without interruption to the in service VNF instances.
[Req.10.8.25]	Any operation of the NFV-MANO system that modifies internal state data within the NFVMANO should be atomic, and the integrity of the state data should be verifiable. This ensures the integrity of the NFV-MANO system should failures occur during orchestration operations.

Annex C:
Authors & contributors

The following people have contributed to the present document:

Rapporteur:
Chidung Lac, Orange

Other contributors:
Randee Adams, Nokia

Pradeepsunder Ganesh, Intel

Olivier Le Grand, Orange

Tommy Lindgren, Ericsson

Astrid Mann, Huawei

Shaoji Ni, Huawei

Percy Tarapore, AT&T

Maria Toeroe, Ericsson

Annex D:
Change history

Date	Version	Information about changes
April 2016	0.0.1	First version of the ToC and the scope
Nov 2016	0.0.2	Addition of clause 5
Dec 2016	0.0.3	Addition of clause 6 (#5 starting from V0.0.5)
Jan 2017	0.0.4	Addition of clause 7 (#6 starting from V0.0.5) + slight modifications in clause 6 (#5 starting from V0.0.5)
March 2017	0.0.5	Modification of the ToC (clause 5 ("MANO: from initial needs to current capabilities") moved to annex) Modification of the clause 2 ("References") Addition of clause 3 ("Abbreviations") Addition of clause 4 ("Introduction") Addition of clause 9 ("Security considerations")
April 2017	0.0.6	Addition of clause 7 ("Resiliency mechanisms applied to MANO's critical capabilities") Addition of clause 8 ("Recommendations and guidelines") Slight modifications in clauses 2, 3, 5.2.1, 6.1 and annex A
June 2017	0.0.7	General revision of all clauses
July 2017	0.0.8	Editorial modification of the scope + clause 8 revision

History

Document history		
V1.1.1	September 2017	Publication