ETSI GS NFV-IFA 011 V4.4.1 (2023-03)

Network Functions Virtualisation (NFV) Release 4; Management and Orchestration; VNF Descriptor and Packaging Specification

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
RGS/NFV-IFA011ed441

Keywords
management, MANO, NFV, orchestration,
virtualisation

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 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are 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 Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, 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.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

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

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**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 provides requirements for the structure and format of a VNF Package to describe the VNF properties and associated resource requirements in an interoperable template.

The focus is on VNF packaging, meta-model descriptors (e.g. VNFD) and package integrity and security considerations.

The present document also specifies requirements for the structure and contents of VNF Snapshot Package.

2 References

2.1 Normative references

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

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

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

The following referenced documents are necessary for the application of the present document.

[1] Void.

[2] Hash Function Textual Names registry at IANA.

[3] ISO/IEC 9899: "Information Technology -- Programming languages -- C".

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-IFA 002: "Network Functions Virtualisation (NFV); Acceleration Technologies; VNF Interfaces Specification".

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

[i.3] ETSI GS NFV-IFA 007: "Network Functions Virtualisation (NFV); Management and Orchestration; Or-Vnfm reference point - Interface and Information Model Specification".

[i.4] ETSI GS NFV-IFA 008: "Network Functions Virtualisation (NFV); Management and Orchestration; Ve-Vnfm reference point - Interface and Information Model Specification".

[i.5] ISO/IEC 9646-7: "Information technology -- Open Systems Interconnection -- Conformance testing methodology and framework -- Part 7: Implementation Conformance Statements".

[i.6] Void.

[i.7] Assigned Internet Protocol Numbers.

[i.8] ETSI GS NFV-IFA 014: "Network Functions Virtualisation (NFV); Management and Orchestration; Network Service Templates Specification".

[i.9] IETF RFC 4090: "Fast Reroute Extensions to RSVP-TE for LSP Tunnels".

[i.10] Void.

[i.11] ETSI GR NFV 003: "Network Functions Virtualisation (NFV); Terminology for main concepts in NFV".

[i.12] ETSI GS NFV-IFA 040: "Network Functions Virtualisation (NFV) Release 4; Management and Orchestration; Requirements for service interfaces and object model for OS container management and orchestration specification".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI GR NFV 003 [i.11] apply.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in ETSI GR NFV 003 [i.11] and the following apply:

ARM	Advanced RISC Machine
CDN	Content Delivery Network
CP	Connection Point
CPD	Connection Point Descriptor
CPU	Central Processing Unit
DF	Deployment Flavour
DSL	Domain Specific Language
EM	Element Manager
GS	Group Specification
IFA	Infrastructure and Architecture Working Group
IP	Internet Protocol
ISG	Industry Specification Group
LAN	Local Area Network
LCM	Life Cycle Management
MAC	Media Access Control
MPLS	MultiProtocol Label Switching
NFV	Network Functions Virtualisation
NFVI	Network Functions Virtualisation Infrastructure
NFVO	Network Functions Virtualisation Orchestrator
NS	Network Service
PM	Performance Management
QA	Quality Assurance
QoS	Quality of Service
RAM	Random Access Memory
RDMA	Remote Direct Memory Access
SAL	Service Availability Level
SW	Software
UML	Unified Modelling Language

URL	Uniform Resource Locator
VDU	Virtualisation Deployment Unit
VIM	Virtualised Infrastructure Manager
VL	Virtual Link
VLD	Virtual Link Descriptor
VM	Virtual Machine
VNF	Virtualised Network Function
VNFC	Virtualised Network Function Component
VNFD	Virtualised Network Function Descriptor
VNFM	Virtualised Network Function Manager

4 General description

4.1 Introduction

The present document develops specifications for packaging of VNFs to be delivered to service providers, focusing on the holistic end-to-end view of the VNF Package lifecycle, from design to runtime, capturing development as well as operational views. The present document provides an analysis of end-to-end VNF Package lifecycle management operations based on use-cases and NFV Architectural Framework functional blocks.

A VNF Package contains all of the required files and meta-data descriptors required to validate and instantiate a VNF.

Standardized meta-data descriptors are required to:

- describe the NFV infrastructure resource requirements for a VNF in a service provider environment;
- describe design constraints and other dependencies in order for the VNF to successfully install, instantiate and terminate; and
- describe VNF operational behaviour including VNF lifecycle events (e.g. scaling, upgrading).

Standardized packaging and validation of VNFs is required to:

- provide a consistent, documented method for VNF providers to package VNFs;
- harmonize the service provider on-boarding process for VNFs coming from different VNF providers;
- ensure integrity, trust and auditability of a VNF Package;
- allow for a flexible and extensible VNF packaging structure that accommodates a wide variety of NFV infrastructure scenarios; and
- allow the packaged VNF-related meta-data to be interpreted and the packaged VNF to be instantiated in a wide variety of orchestration systems irrespective of technology choice or infrastructure environment.

4.2 Objectives

The present document delivers:

- A description of a set of use cases involving the handling of VNF Packages.
- A set of functional requirements to be fulfilled when packaging a VNF.
- A specification of the information elements and attributes applicable to the VNFD.

4.3 Conventions

The attributes of the VNFD and associated information elements are described in the tables provided in clause 7. Each table has 5 columns, with the following significance:

- The "Attribute" column provides the attribute name.
- The "Qualifier" column indicates whether the support of the attribute is mandatory, optional or conditional.
- The "Cardinality" column contains the minimum and maximum cardinality of this information element (e.g. 1, 2, 0..N, 1..N). A cardinality range starting with 0 indicates that the attribute need not always be included.
- The "Content" column provides information on the type of the attribute values. It can be the name of an Information Element, a primitive type (Identifier, DateTime, etc.) or a generic UML type (String, Integer, etc.). If a cell in the "Content" column is marked as "Not specified", this means that the specification of the type is left to the data model design stage.
- The "Description column" provides a brief explanatory description and additional constraints.

The following notations, defined in ISO/IEC 9646-7 [i.5], are used for the qualifier column:

- M mandatory - the attribute shall be supported.
- O optional - the attribute may, but need not to, be supported.
- CM conditional mandatory - the attribute shall be supported under certain conditions. If the specified conditions are met then the attribute shall be supported. These conditions are specified in the Description column.
- CO conditional optional - the attribute may, but need not to, be supported under certain conditions. These conditions are specified in the Description column.

A Mandatory qualifier would imply that NFVO/VNFM shall understand/parse the particular element but the presence (inclusion in an occurrence of a VNFD) of the element is dictated by Cardinality. The lower bound of "1.." cardinality would imply that the attribute shall be present in the VNFD.

The following notations are used for the content column of information elements, input parameters, notifications, etc.:

- Parameters are of type "Identifier" when referring to an identifier of an actual object.
- For a "true" identifier identifying an object (information element or structure) the content type "Identifier" and the description "Identifier of this <object_name> <notification/information element/…>" is used.

EXAMPLE: Identifier "resourceId" of the "NetworkSubnet information element" shall have the description "Identifier of this NetworkSubnet information element".

- Object(s) are referenced by their identifier using the syntax "Identifier (Reference to <object_name1> [, <object_name2>…][, or <object_nameN>])".
- Names for attributes and parameters of type Identifier shall be of the following pattern: <name>Id.

4.4 Levels of NFV Entities

For NFV management, there are four levels of entities, i.e.:

- Descriptors - general type definitions for entities such as VNFs and VLs, e.g. VNFD and VLD.
- Descriptor objects - an instance of a descriptor, e.g. an instance of a VNFD (not an instance of a VNF instantiated according to this VNFD):
  - A descriptor object may provide (among other things) value ranges and default values for the attributes in the associated NFV entity class.

  - In the present document, the creation of subclasses of generic descriptors (e.g. VNFD_x as a subclass of VNFD) has been avoided, since this approach would create a proliferation of descriptor classes.

- NFV Entity Classes - these are classes that represent various NFV entities such as VNF and VL. There is one-to-one mapping between a descriptor object and an NFV entity class. An example of an NFV Entity Class is CDN Cache VNF.

- NFV Entity Instances - these are instances of a given NFV entity class. An NFV entity instance is used to represent the current state and attribute values for a given NFV entity. Each NFV entity instance is bound by the associated descriptor object, e.g. value ranges and default values for attributes. An example of an NFV Entity Instance is a CDN Cache VNF instance.

Each level puts constraints on the subsequent levels.

Information in a lower level does not appear in a higher level, e.g. NFV entity instance information does not appear in the associated NFV entity class, descriptor object or descriptor.

For example:

- A VNFD has parameters such as virtualisationDeploymentUnit, intVirtualLinkDesc, extConnectionPointDesc and deploymentFlavour. These same parameters apply to every type of VNF.

- For a given type of VNF (e.g. a firewall), one would create an instance of the VNFD and populate the various VNFD parameters with values specific to the given type of firewall: specific VDU instances describing the resource requirements for this VNFD instance, VLD instances describing the various types of VL needed, specific Deployment Flavour (DF), etc.

- Next, one defines the class for the given VNF firewall. The class includes the attributes that are seen across the given reference point.

- Finally, one can instantiate one or more VNF firewall by populating the various attributes in the VNF class with actual values.

5 VNF Packaging use-cases (informative)

5.1 General

The following use cases describe the steps involving the VNF Package as it transitions from the VNF Provider to the Service Provider. They capture the generic processes as well as the actions required to be performed by actors playing different roles in order to identify the requirements for the standard packaging format.

All the use cases presented in this clause are informative.

For the purpose of the use cases, the roles identified in table 5.1-1 have been identified.

Table 5.1-1: List of roles

Role	Description
VNF Provider	The role providing the VNF. Actors that can play this role include, but are not limited to, vendor, integrator or in-house developer.
Supply Chain Specialist	Service provider function responsible for recommending or identifying VNFs required for desired services.
Service Designer	Service provider function responsible for defining and providing requirements (functional and non-functional) for required services. Also responsible for creating services to be deployed by the service provider.
Service Acceptance Specialist	Service provider function responsible to validate, certificate and on-board VNFs.
Service Deployment Manager	Service provider function responsible for managing the deployment (e.g. instantiation, update) of the VNFs and VLs validated by the Service Acceptance Specialist.

5.2 VNF Package bundling for distribution

A VNF is, from a delivery point of view, a software application so most of the general principles and processes associated with the software development lifecycle apply. After a VNF provider completes the development and functional testing for the VNF it needs to bundle all the necessary binaries and corresponding metadata for distribution to potential customers.

Roles

#	Role
1	VNF Provider

Pre-conditions

#	Pre-conditions	Comment
1	Functional Testing was performed and the version of the VNF has been identified	

Post-conditions

#	Post-conditions	Comment
1	A versioned single file package	

Base Flow

#	Role	Action/Description
1	VNF Provider	Using their own software development lifecycle tools and procedures, retrieve all the software components associated with the version to be built. This includes but not limited to own developed code, configuration files as well as third party components with their code, as well as build scripts and optionally license terms information (human readable file).
2	VNF Provider	Capture the release notes including clear description of the functionality the release delivers, any external dependencies, known bugs fixed relative to the prior releases as well as known issues in specific configurations.
3	VNF Provider	Bundle the release, sign the package and place it in a distribution repository.

5.3 VNF Package testing

The VNF Package testing encompasses steps to guarantee that the package adheres to the standard structure and contains the mandatory metadata required in order to be considered compliant with the industry format.

Roles

#	Role
1	VNF Provider

Pre-conditions

#	Pre-conditions	Comment
1	Versioned Package is signed and available for distribution	

Post-conditions

#	Post-conditions	Comment
1	Package is flagged as Validated	

Base Flow

#	Role	Action/Description
1	VNF Provider	Using parsing tools to perform a final test on the package in order to make sure that: • VNF Package signature can be validated. • VNF Package can be unbundled. • VNF Package has the right structure (files, directories) as expected by onboarding tools.

5.4 VNF pre procurement

Prior to acquiring the VNFs, the Service Provider will match the VNF against their needs allowing them to compare different offers from different suppliers.

Roles

#	Role
1	Supply Chain Specialist
2	Service Designer

Pre-conditions

#	Pre-conditions	Comment
1	Supply Chain Specialist has received clear functional and non-functional requirements from Service Designers	
2	Supply Chain Specialist obtained versioned package from VNF Provider	

Post-conditions

#	Post-conditions	Comment
1	Recommendation for purchase	

Base Flow

#	Role	Action/Description
1	Supply Chain Specialist	Identifies and quantifies the VNF attributes against the service requirements by retrieving VNF metadata describing the scalability, reliability, manageability and security attributes of the package.

5.5 VNF Package validation and certification

A VNF Package is composed of several components like e.g. VNFD, software images, scripts, etc. During the on-boarding of the VNF Package, a validation of the package is performed. The validation is a procedure that verifies the integrity of the VNF Package.

A package is certified by performing acceptance testing and full functional testing against the VNF including configuration, management and service assurance.

Roles

#	Role
1	Service Acceptance Specialist

Pre-conditions

#	Pre-conditions	Comment
1	VNF Package is available for onboarding	

Post-conditions

#	Post-conditions	Comment
1	VNF Package is validated	
2	VNF Package is marked as certified	

Base Flow

#	Role	Action/Description
1	Service Acceptance Specialist	Validate the package signature, origin, contents and structure.
2	Service Acceptance Specialist	Perform a full onboard, setup, install in a QA environment and certify the VNF for functionality as well as authenticity, integrity and packaging compliance.

5.6 VNF install

VNF is installed and ready to be configured and used for network services.

Roles

#	Role
1	Service Deployment Manager

Pre-conditions

#	Pre-conditions	Comment
1	VNF is on-boarded and available for Service Orchestration	

Post-conditions

#	Post-conditions	Comment
1	VNF is installed and ready to be configured for use in network services	

Base Flow

#	Role	Action/Description
1	Service Deployment Manager	Identify the desired VNFs, configure and instantiate them according to the deployment policies. VNF configuration is based on parameterization captured at design time, included in the VNF Package, and complemented during VNF instantiation.

5.7 Keeping NFV management and orchestration in sync about a VNF application software modification

For currently deployed VNFs on-boarding of new versions will need the ability to keep track of multi version, multi environment multi instance and allow the service provider team to perform updates/upgrades with clear expectations of service continuity based on metadata information including component dependencies.

The use case below focuses on updating the information about a VNF instance stored in NFV management and orchestration as a result of a VNF application software modification performed through service provider's management system, wherein such a process only comprises modifying the VNF's application software without requiring a change of the VNF's underlying virtualised resources or internal VNF component (VNFC) topology/composition (see figure 5.7-1). Examples of VNF application software modification are: update, upgrade, and downgrade. Such a modification may be performed without requiring the termination of the VNF instance with the prior VNF application software version. Consequently, the relevant VNF Package is replaced by a different VNF Package which includes the VNF application software used in the modification.

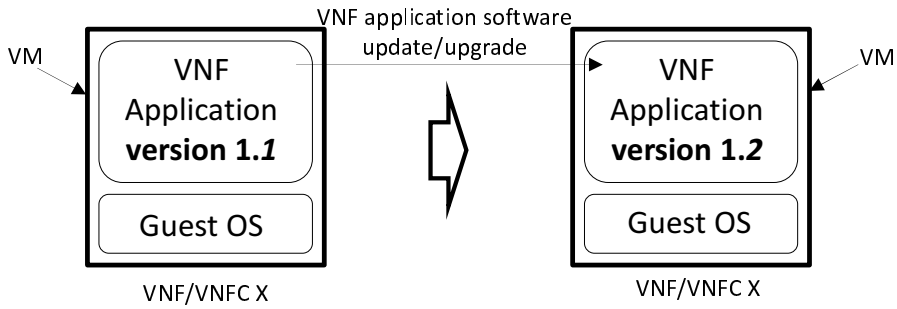

Figure 5.7-1: Example of VNF application software modification

Roles

#	Role
1	VNF Provider, Service Acceptance Specialist, Service Deployment Manager

Pre-conditions

#	Pre-conditions	Comment
1	Prior version of VNF already instantiated and in use.	
2	A VNF application software version to be used for the modification of a VNF instance has been certified.	

Post-conditions

#	Post-conditions	Comment
1	The VNF instance with the modified application software is available.	
2	The VNF Package with the VNF application software used in the modification is on-boarded.	
3	The VNF instance information refers to the VNF Package with the VNF application software used in the modification.	

Base Flow

#	Role	Action/Description
1	VNF Provider	Provide the VNF Package including the VNF application software to be used in the modification.
2	Service Acceptance Specialist	On-board the VNF Package of step 1 to the NFVO.
3	Service Deployment Manager	Perform the modification of the VNF instance's application software through Service Provider's management system.
4	Service Deployment Manager	Modify the VNF instance information in the NFVO/VNFM to refer to the VNF Package that includes the VNF application software used in the modification.

5.8 VNF configurable parameter provisioning

The VNFD is a static description file, not a dynamic configuration file. The metadata description in the VNFD is not changed during the whole VNF lifecycle. Some VNF parameters described in the VNFD can be declared to be configurable during the VNF design phase, and further be configured by the VNFM during or after VNF instantiation. This use case provides a scenario where the VNF configurable parameters described in the VNFD are provisioned.

Roles

#	Role
1	Service Acceptance Specialist, Service Deployment Manager

Pre-conditions

#	Pre-conditions	Comment
1	The description of the VNF configurable parameters that is described or declared in the VNFD has been encapsulated in the VNFD during the VNF design phase.	

Post-conditions

#	Post-conditions	Comment
1	The VNF configurable parameters in the VNFD are provisioned (configured with a real value) after VNF instantiation, and can also be re-configured at any time of VNF lifecycle.	

Base Flow

#	Role	Action/Description
1	Service Acceptance Specialist	The NFVO on-boards the VNF Package and stores the VNFD.
2	Service Deployment Manager	The VNFM accesses to the VNFD, reads the description of each VNF parameter and determines whether it is configurable. See note 1.
3	Service Deployment Manager	For each configurable VNF parameter in the VNFD, based on the interaction with the NFVO, the VNFM configures the value of VNF parameter during VNF instantiation (i.e. when the VNF is deployed). See note 2.
NOTE 1: VNF configurable parameters in the VNFD (e.g. the IP address of element manager for the VNF) belong to virtualisation-related configuration parameters of the VNF as specified in ETSI GS NFV-IFA 008 [i.4].		
NOTE 2: This configuration step is a part of VNF instantiation instead of VNF update.		

6 Functional requirements for VNF Packaging

6.1 Generic Functional Requirements

Table 6.1-1 specifies generic functional requirements applicable to VNF Packaging.

Table 6.1-1: Generic functional requirements for VNF Packaging

Numbering	Requirement Description	Comments
VNF_PACK.GEN.001	The VNF Package contents, including the VNF descriptor, VNF Binaries, configuration, scripts and software images, as well as manifest file, checksum, etc. as appropriate constitutes a single delivery unit from a distribution perspective. Any changes to the constituency of this unit shall be considered as a change to the whole and therefore shall be versioned, tracked and inventoried as one.	

6.2 Functional requirements for VNF Packaging specification

6.2.1 Requirements for the structure of a VNF Package

Table 6.2.1-1 specifies requirements applicable to the structure of a VNF Package.

Table 6.2.1-1: Requirements for the structure of a VNF Package

Numbering	Requirement Description	Comments
VNF_PACK.STRUCT.001	The VNF Package shall be assembled in one file.	
VNF_PACK.STRUCT.002	The VNF Package shall be digitally signed by the VNF Provider.	
VNF_PACK.STRUCT.003	The VNF Package should contain files for one VNF and its corresponding metadata.	
VNF_PACK.STRUCT.004	The VNF Package shall enable including VNF specific files organized according to the design of the VNF, or referencing these files if they are external to the package. See note.	
VNF_PACK.STRUCT.005	The VNF Package shall provide means to address individually the files which it contains and/or which it references.	
VNF_PACK.STRUCT.006	If an external reference (e.g. URL) is used, file integrity information (such as checksum/signature) shall be specified to guarantee the integrity of the referenced file, so it cannot be substituted with a different file by the same name.	
NOTE: This can include e.g. software images and additional specific files to run and manage the VNF, supplied by the VNF provider.		

6.2.2 Requirements for the description of VNF Package content

Table 6.2.2-1 specifies requirements applicable to the content of a VNF Package.

Table 6.2.2-1: Requirements for the description of VNF Package content

Numbering	Requirement Description	Comments
VNF_PACK.DESC.001	The VNF Package may contain the license terms information (a human readable file) under which the packaged VNF is released.	
VNF_PACK.DESC.002	The VNF Package may contain other license terms information (human readable file(s)) corresponding with all the components included in the package if different than the one of the VNF.	
iVNF_PACK.DESC.003	The VNF Package shall contain a Change Log. Change log captures the changes from one version to another including but not limited to features added/removed, issues fixed as well as known issues not resolved.	
VNF_PACK.DESC.004	VNF Package shall contain or reference one or more software images (VM images or OS container images).	
VNF_PACK.DESC.005	The VNF Package may contain or reference at most one software image per VDU in case the VNFC is realized as a virtual machine or one software image per OS container descriptor in case the VNFC is realized by one or a set of OS containers.	In case different virtualisation environments require different SW images of a VNFC they will be delivered in separate VNF Packages.
VNF_PACK.DESC.006	The VNF Package shall provide a mechanism to describe the package and its contents including, not limited to, version of the package, provider of the package and identification of the included metadata/artifacts.	
VNF_PACK.DESC.007	The VNF Package shall contain VNFD metadata.	
VNF_PACK.DESC.008	VNFD metadata shall not be modified once the package is assembled.	
VNF_PACK.DESC.009	VNFD metadata shall be placed in a well-known location within the VNF Package in order for the compliant parsers to find and extract.	
VNF_PACK.DESC 010	The VNF package shall enable including information supporting VNF testing.	This information may include test scripts and/or dependencies on an external test system.
VNF_PACK.DESC 011	The VNF Package shall allow to store in the package sets of related artifacts for use by functional blocks beyond NFV-MANO, and to assign a globally unique identifier to each set in an SDO-independent and vendor-independent manner.	
VNF_PACK.DESC.012	The VNF Package shall contain one or more Managed Container Infrastructure Object Packages (MCIOP) representing aggregated containerized workload structures, when the VNF is realized by a set of OS containers.	

6.2.3 Requirements for VNF Identification

Proper VNF Identification is required across the VNF lifecycle from development to retirement/decommission.

Table 6.2.3-1 specifies requirements applicable to the VNF identification.

Table 6.2.3-1: Requirements for the VNF Identification

Numbering	Requirement Description	Comments
VNF_PACK.ID.001	There shall be a way to identify the version of the VNF Package Specification associated with a particular VNF.	This should guarantee compliance with the present document and allow systems parsing the metadata in the template to associate data elements with schema definition for compatibility reasons.
VNF_PACK.ID.002	VNF Package shall be globally uniquely identifiable. The globally unique identifier for the VNF Package shall be used to uniquely identify the VNFD and the VNF included in the package.	The unique identification is needed by the service provider for onboarding, operations and in order to properly associate subsequent upgrades, patches and fixes delivered to the service provider.
VNF_PACK.ID.003	VNF Package Identification Metadata shall contain: • VNF Provider. • VNF Product name. • VNF Release Date/Time. • VNF Package Version (version of the VNF release).	This is similar to current asset management practices for physical equipment by Make, Model and version.
VNF_PACK.ID.004	VNF Product Name and VNF Provider shall not be changed throughout the lifespan of the VNF. This is to aid with correlation between different versions of a VNF with the same code base.	VNF lifespan is defined and set by the VNF Provider on a case by case basis considering the product management, portfolio roadmap or any other commercially related factors.

6.2.4 Requirements for security and integrity of a VNF Package

Table 6.2.4-1 specifies the requirements applicable to the security and integrity of a VNF Package.

Table 6.2.4-1: Requirements for security and integrity of a VNF Package

Numbering	Requirement Description	Comments
VNF_PACK.SEC.001	The digest and the public key of the entity signing VNF Package shall be included in the package along with the corresponding certificate.	
VNF_PACK.SEC.002	For each signed artifact, corresponding public key, algorithm and certificate used shall be stored in a well-known location within the VNF Package.	
VNF_PACK.SEC.003	Security sensitive artifacts shall be encrypted. Encryption keys for these artifacts should be different than the VNF Package key to allow for better access control within the provider environment.	
VNF_PACK.SEC.004	Each artifact in the VNF Package shall be signed by the VNF provider.	

6.2.5 Requirements for VNFD Metadata

Table 6.2.5-1 specifies requirements applicable to VNFD metadata.

Table 6.2.5-1: Requirements for VNFD Metadata

Numbering	Requirement Description	Comments
VNF_PACK.META.001	The VNFD shall support a description of deployment policies.	
VNF_PACK.META.002	The VNFD shall support a description of required virtualisation containers in terms of e.g. amount, characteristics and capabilities for virtual CPUs and virtual RAM and virtual disks.	
VNF_PACK.META.003	The description of a virtualisation container in the VNFD shall support a description of attached additional virtual devices and their characteristics and capabilities.	The description of additional virtual devices may include, but is not limited to, virtual CDROM drives, virtual NICs and special configuration drives.
VNF_PACK.META.004	The description of a virtualisation container in the VNFD shall support a description of acceleration capabilities and characteristics.	The description of acceleration capabilities may include, but is not limited to, crypto, video transcoding, or RDMA.
VNF_PACK.META.005	The VNFD shall support a description of the minimum and maximum number of instances of each particular virtualisation container that conform to the VNF.	
VNF_PACK.META.006	The VNFD shall support a description of the VNF internal connectivity, including the connectivity between virtualisation containers, and associated connectivity resource requirements.	
VNF_PACK.META.007	The VNFD shall support a description of one or more DFs to choose a particular variant of the VNF to be instantiated.	
VNF_PACK.META.008	The VNFD shall support a description of parameters to be monitored for the VNF after instantiation.	
VNF_PACK.META.009	The VNFD shall support a description of parameters which can be configured for the VNF and whether the parameters can be configured after VNF instantiation.	The parameters may be combined with default values.
VNF_PACK.META.010	The VNFD shall support a description of lifecycle events and related actions which can be performed for the VNF.	
VNF_PACK.META.011	The VNFD shall support a description of metadata about the VNF product.	The metadata shall include, but is not limited to, name, version, unique identifier and provider name of the VNF.
VNF_PACK.META.012	The VNFD shall support a description of metadata about placement of virtualisation containers relative to each other.	Placement may include, but is not limited to, affinity or anti-affinity.
VNF_PACK.META.013	The VNFD shall support a description of the supported VNF instance scaling.	
VNF_PACK.META.014	The VNFD shall support a description of rules for auto-scaling describing which actions shall be executed if a condition involving monitoring parameters and/or VNF Indicators is satisfied.	An action may be the trigger of a lifecycle event or an alarm.
VNF_PACK.META.015	The VNFD shall support a description of metadata to determine if an EM is used for the VNF and parameters describing how to connect to the EM.	Deployment specific information e.g. the IP address of the EM may be specified using instantiation specific parameters (see VNF_PACK.META.018).
VNF_PACK.META.016	The VNFD shall support a description of metadata about dependencies between virtualisation containers.	Dependencies may include, but is not limited to existence of a dependency.

Numbering	Requirement Description	Comments
VNF_PACK.META.017	The VNFD shall support a description of Service Availability Level (SAL) requirements for virtual resources on the underlying NFVI.	SAL requirements may be described for a VNF as well as for individual VDUs.
VNF_PACK.META.018	The VNFD shall support a description of parameters whose values have to be specified as input to the instantiation process.	
VNF_PACK.META.019	The VNFD shall support metadata related to network addresses to be assigned to Connection Point(s) (CP).	For example the metadata for layer 3 network addresses can include IP address type, range, and allocation scheme.
VNF_PACK.META.020	The VNFD shall support the description of VNF indicators.	See note.
VNF_PACK.META.021	The VNFD shall support a description of external CP supported by the VNF enabling connectivity with one or more external entities.	
VNF_PACK.META.022	The description of a virtualisation container in a VNFD shall support a description of meta data about software image(s).	
VNF_PACK.META.023	The VNFD shall provide the possibility to reference information elements via URLs e.g. to external files provided by the VNF provider.	
VNF_PACK.META.024	The VNFD shall provide a reference to the VNFM(s) compatible with the VNF described in the VNFD.	
VNF_PACK.META.025	The VNFD shall support a description of the security rules to filter the ingress/egress packets related to the VNF.	The filtering rules include, but are not limited to the packet direction, TCP/UDP port range, IP protocol, etc.
VNF_PACK.META.026	The VNFD shall support associating the security rules to the relevant VNF connection points.	
VNF_PACK.META.027	The VNFD shall support a description of the information for changing the current VNF Package applicable to a VNF.	
VNF_PACK.META.028	The VNFD shall support the possibility to reference one or more MCIOP(s) used in containerized workload management.	The reference to the MCIOP(s) is used as input for the VNF LCM operations.
VNF_PACK.META.029	The VNFD shall support the possibility to reference OS container images used in OS container image management.	
NOTE: VNF Indicators are information supplied by the VNF or the EM to provide some indication on the VNF behaviour. VNFM can use these indicators in conjunction with e.g. monitoring parameters to perform auto-scaling decisions or to trigger a VNF LCM script. These indicators are applicable at both the VNF level (e.g. global indicators) and the deployment flavour level of a certain VNFD (e.g. local indicators). The values of local indicators complement the values of global indicators.		
DISCLAIMER: Not all listed requirements are supported by the information elements specified in clause 7.		

6.2.6 Requirements for LCM scripts

6.2.6.1 General

Table 6.2.6.1-1 specifies requirements for Life Cycle Management (LCM) scripts.

Table 6.2.6.1-1: Requirements for LCM scripts

Numbering	Requirement Description	Comments
VNF_PACK.LCM.001	LCM scripts embedded in the VNF Package and to be used in the LCM execution environments provided by generic VNF Managers shall be specified using a Domain Specific Language (DSL) that fulfils the requirements specified in the following clauses.	See note.
NOTE: The specification of a DSL fulfilling the requirements specified in the following clauses is outside the scope of the present document.		

6.2.6.2 Requirements for DSL

Table 6.2.6.2-1 specifies requirements that shall be fulfilled by the DSL used to specify lifecycle management scripts embedded in the VNF Package.

Table 6.2.6.2-1: DSL requirements for LCM scripts

Numbering	Requirement Description	Comments
VNF_PACK.LCMDSL.001	The DSL shall support arithmetic, comparison and logical operators defined in ISO/IEC 9899 [3].	
VNF_PACK.LCMDSL.002	The DSL shall support expressing policy rules associating conditions with actions.	
VNF_PACK.LCMDSL.003	The DSL shall enable expressing a condition that is the receipt of a request invoking one of the operations of the VNF Lifecycle Management interface.	
VNF_PACK.LCMDSL.004	The DSL shall enable expressing a condition that is the receipt of a notification.	
VNF_PACK.LCMDSL.005	The DSL shall enable expressing conditions on the values of the parameters of an operation request.	
VNF_PACK.LCMDSL.006	The DSL shall enable using extended regular expressions to express conditions on the values of the parameters of an operation request. See example.	
VNF_PACK.LCMDSL.007	The DSL shall enable expressing as a condition the detection that the value of an internal variable used by the script is equal, greater or less than a threshold defined by the script.	
VNF_PACK.LCMDSL.008	The DSL shall enable expressing actions leading to setting, incrementing and decreasing internal variables.	
VNF_PACK.LCMDSL.009	The DSL shall enable expressing actions leading to: • invoke an operation of the Software Image Management interface; • invoke an operation of the Virtualised Resources Information Management interface; • invoke an operation of the Virtualised Resources Management interface; • invoke an operation of the Virtualised Resources Change Notification interface; • invoke an operation of the Virtualised Resources Reservation Management interface; • invoke an operation of the Virtualised Resources Performance Management interface; • invoke an operation of the Virtualised Resources Fault Management interface; • invoke an operation of the VNF Configuration interface; • invoke an operation of the VNF Indicator interface; • invoke an operation of the VNF Lifecycle Operation Granting interface; • invoke an operation of the LCM Coordination interface; • invoke an operation of the OS container workload management service interface; • invoke an operation of the OS container compute management service interface; • invoke an operation of the OS container storage management service interface; • invoke an operation of the OS container network management service interface; • invoke an operation of the OS container configuration management service interface; and • invoke an operation of the OS container image management service interface. See note 2.	

Numbering	Requirement Description	Comments
VNF_PACK.LCMDSL.010	The DSL shall enable mapping LCM script variables on to: • parameters of the VNFD; • parameters of operation requests and results.	
VNF_PACK.LCMDSL.011	The DSL shall enable a LCM script to access arbitrary artifacts in the VNF Package.	
NOTE 1: The DSL does not provide means to specify where to send the operation request. The VNFM script execution environment will determine where to send the operation request based on local policies and/or information received from the NFVO. NOTE 2: Operations that can be invoked correspond to operations specified in ETSI GS NFV-IFA 006 [i.2], ETSI GS NFV-IFA 007 [i.3], ETSI GS NFV-IFA 008 [i.4] and ETSI GS NFV-IFA 040 [i.12], where the VNFM acts as request consumer.		
EXAMPLE: Assuming the case that virtualised container instances have an attribute "name" and there are two instances named "boba" and "bobb", while listing virtualised container instances information, usage of the regular expression "bob." would request the producer to return information from instances named "boba" and "bobb".		

7 Virtualised Network Function information elements

7.1 VNF Descriptor (VNFD)

7.1.1 Introduction

The clauses below define the information elements related to the VNFD. A UML representation of the VNFD high-level structure is shown in figure 7.1.1-1.

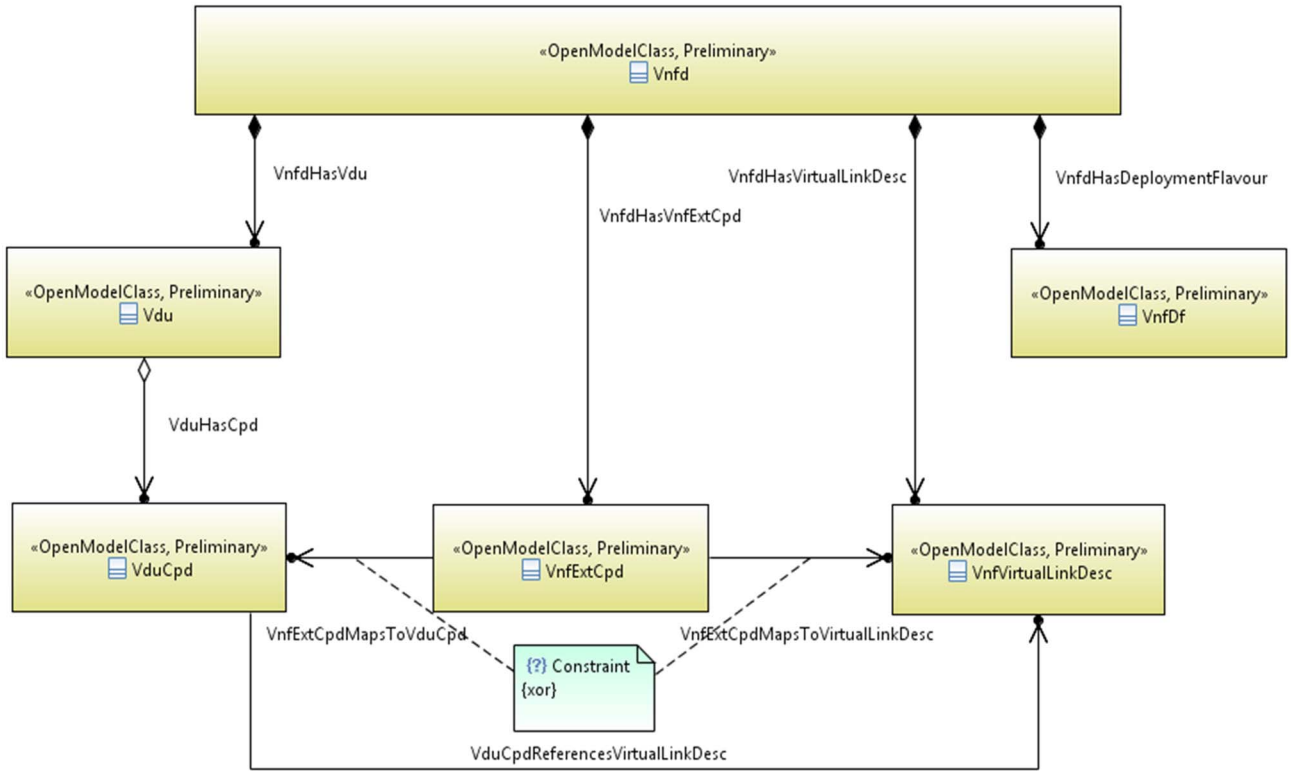

Figure 7.1.1-1: VNFD high-level structure

7.1.2 Vnfd information element

7.1.2.1 Description

A VNFD is a deployment template which describes a VNF in terms of deployment and operational behaviour requirements. It also contains connectivity, interface and virtualised resource requirements.

7.1.2.2 Attributes

The attributes of the Vnfd information element shall follow the indications provided in table 7.1.2.2-1.

Table 7.1.2.2-1: Attributes of the Vnfd information element

Attribute	Qualifier	Cardinality	Content	Description
vnfdId	M	1	Identifier	Identifier of this Vnfd information element. This attribute shall be globally unique. The format will be defined in the data model specification phase. See note 1.
vnfdExtInvariantId	M	0..1	Identifier	Identifies a VNFD in a version independent manner. This attribute is invariant across versions of the VNFD that fulfil certain conditions related to the external connectivity and management of the VNF. See note 8.
vnfProvider	M	1	String	Provider of the VNF and of the VNFD.
vnfProductName	M	1	String	Name to identify the VNF Product. Invariant for the VNF Product lifetime.
vnfSoftwareVersion	M	1	Version	Software version of the VNF. This is changed when there is any change to the software that is included in the VNF Package.
vnfdVersion	M	1	Version	Specifies the version of the VNFD.
vnfProductInfoName	M	0..1	String	Human readable name for the VNF Product. Can change during the VNF Product lifetime.
vnfProductInfoDescription	M	0..1	String	Human readable description of the VNF Product. Can change during the VNF Product lifetime.
vnfmInfo	M	1..N	String	Specifies VNFM(s) compatible with the VNF described in this version of the VNFD.
localizationLanguage	M	0..N	Not specified	Information about localization languages of the VNF (includes e.g. strings in the VNFD). See note 4.
defaultLocalizationLanguage	M	0..1	Not specified	Default localization language that is instantiated if no information about selected localization language is available. Shall be present if "localizationLanguage" is present and shall be absent otherwise.
vdu	M	1..N	Vdu	Virtualisation Deployment Unit. See clause 7.1.6.
virtualComputeDesc	M	0..N	VirtualComputeDesc	Defines descriptors of virtual compute resources to be used by the VNF when each of the VNFC instances of the VNF is intended to be deployed in a single VM. See note 6. See clause 7.1.9.2.2.
virtualStorageDesc	M	0..N	VirtualStorageDesc	Defines descriptors of virtual storage resources to be used by the VNF. See clause 7.1.9.4.2.

Attribute	Qualifier	Cardinality	Content	Description
osContainerDesc	M	0..N	OsContainerDesc	Defines descriptors of container compute resources to be used by the VNF when the VDUs of the VNF are realized by a set of OS Containers sharing the same host and same networking namespace. See note 6. See clause 7.1.6.13.
swImageDesc	M	0..N	SwImageDesc	Defines descriptors of software images to be used by the VNF. See clause 7.1.6.5. See note 5.
intVirtualLinkDesc	M	0..N	VnfVirtualLinkDesc	Represents the type of network connectivity mandated by the VNF provider between two or more CPs which includes at least one internal CP. See clause 7.1.7.
securityGroupRule	M	0..N	SecurityGroupRule	Defines security group rules to be used by the VNF. See clause 7.1.6.9.
vnfExtCpd	M	1..N	VnfExtCpd	Describes external interface(s) exposed by this VNF enabling connection with a VL. See clause 7.1.3.
vipCpd	M	0..N	VipCpd	Describes virtual IP addresses to be shared among instances of connection points. See clause 7.1.17.
virtualCpd	M	0..N	VirtualCpd	Describes a virtual connection point allowing to access a set of VNFC instances (based on their respective VDUs). See note 7.
deploymentFlavour	M	1..N	VnfDf	Describes specific DF(s) of a VNF with specific requirements for capacity and performance. See clause 7.1.8.
configurableProperties	M	0..1	VnfConfigurableProperties	Describes the configurable properties of the VNF (e.g. related to auto scaling and auto healing). See clause 7.1.12.
modifiableAttributes	M	0..1	VnfInfoModifiableAttributes	Describes the modifiable attributes of the VNF. See clause 7.1.14.
lifeCycleManagementScript	M	0..N	LifeCycleManagementScript	Includes a list of events and corresponding management scripts performed for the VNF. See clause 7.1.13.
vnfIndicator	M	0..N	VnfIndicator	Declares the VNF indicators that are supported by this VNF.
autoScale	M	0..N	Rule	Rule that determines when a scaling action needs to be triggered on a VNF instance e.g. based on certain VNF indicator values or VNF indicator value changes or a combination of VNF indicator value(s) and monitoring parameter(s). See notes 2 and 3.
vnfPackageChangeInfo	M	0..N	VnfPackageChangeInfo	Information used for performing the change of the current VNF Package. More than one VNF Package Change Info construct is possible to cater the possibility that changes of the current VNF Package can be performed for different source VNFDs.
lcmOperationCoordination	M	0..N	VnfLcmOperationCoordination	Provides information used for the coordination in VNF LCM operations.
mciopId	M	0..N	Identifier	Identifies the MCIOP(s) in the VNF package, used in containerized workload management, when the VNF is realized by a set of OS containers.

Attribute	Qualifier	Cardinality	Content	Description
NOTE 1: The VNFD Identifier shall be used as the unique identifier of the VNF Package that contains this VNFD. Any modification of the content of the VNFD or any other modification of the VNF Package shall result in a new VNFD Identifier. NOTE 2: Monitoring parameters are specified as part of VNF flavour, VDU and VL descriptions. NOTE 3: The rule (conditions and actions) can be expressed as a script. NOTE 4: This allows to provide one or more localization languages to support selecting a specific localization language at VNF instantiation time. NOTE 5: This shall be used to describe both the software image loaded on the virtualisation container used to realize a VDU and the software images to be stored on VirtualStorage resources (e.g. volumes) attached to a virtualisation container. NOTE 6: Either the virtualComputeDesc or the osContainerDesc shall contain at least one element. NOTE 7: If the VNF is realized only by one or a set of OS containers, it describes a service implemented by one or a set of OS containers and exposed on the primary container cluster external network in terms of information used to address it from the remote point and the VDU(s) used to deploy the set of containers that implement the service. NOTE 8: Different versions of a VNFD have different vnfdIds but can have the same vnfdExtInvariantId. Different versions of the VNFD with the same vnfdExtInvariantId shall expose: • same external connectivity, i.e. same number of vnfExtCpds and same identifiers • same VNFD attributes used in an NSD when referring to this VNFD: o VNF deployment flavours, VNF instantiation levels: in both cases the identifiers and the number of them shall be the same, but the content of e.g. a particular deployment flavour can change. o VNF indicators: same identifiers and possible values. o ScaleInfo: same scalingAspects identifiers and same levels per scalingAspect. This condition implies that VNFDs with the same vnfdExtInvariantId preserve external invariancy. Therefore, fulfilling this condition allows to use a different version of a VNFD in an NS instance without modification of the NSD on which the NS instance is based. The use of a different version is ultimately under the control of the service provider and it should consider if the NSD fulfils the requirements of the VnfExtCpds (e.g. bitrate, IP version, etc.).				

7.1.3 Information elements related to VnfExtCpd

7.1.3.1 Introduction

The clauses below define the information elements related to the VnfExtCpd.

7.1.3.2 VnfExtCpd information element

7.1.3.2.1 Description

A VnfExtCpd is a type of Cpd and describes an external interface, also known as external CP, exposed by this VNF enabling connection with a VL.

A VnfExtCpd inherits from the Cpd Class (see clause 7.1.6.3). All attributes of the Cpd are also attributes of the VnfExtCpd.

When the VnfExtCpd is mapped to a VduCpd and no floating IP address is used, the values of the attributes inherited by them from the Cpd IE shall be identical for both of these information elements.

When the VnfExtCpd is mapped to a VipCpd and the VduCpd(s) referred from the VipCpd are also exposed as VnfExtCpd(s), the VnfExtCpd mapped to the VipCpd and the VnfExtCpd(s) mapped to the VduCpd(s) shall be related, by means of references, to the same external virtual link descriptor.

NOTE: To determine which VnfExtCpd (those mapped to VipCpd and mapped to a VduCpd) need to connect to the same external virtual link descriptor, the NSD designer would find which VduCpd are "referred" in the VipCpd, and then find from the array of VnfExtCpd which ones are mapped to the "referred" VduCpd.

7.1.3.2.2 Attributes

The attributes of the VnfExtCpd information element shall follow the indications provided in table 7.1.3.2.2-1.

Table 7.1.3.2.2-1: Attributes of the VnfExtCpd information element

Attribute	Qualifier	Cardinality	Content	Description
intVirtualLinkDesc	M	0..1	Identifier (Reference to VnfVirtualLinkDesc)	References the internal Virtual Link Descriptor (VLD) to which CPs instantiated from this external CP Descriptor (CPD) connect. See notes 3 and 4.
intCpd	M	0..1	Identifier (Reference to VduCpd)	References the internal VDU CPD which is used to instantiate internal CPs. These internal CPs are, in turn, exposed as external CPs defined by this external CPD. See note 4.
vipCpd	M	0..1	Identifier (Reference to VipCpd)	References the VIP CPD which is used to instantiate CPs to hold virtual IP addresses. These CPs are, in turn, exposed as external CPs defined by this external CPD. See note 4.
virtualCpd	M	0..1	Identifier (Reference to VirtualCpd)	References the Virtual CPD which is used to describe a virtual connection point allowing to access a set of VNFC instances (based on their respective VDUs). See note 4.
virtualNetworkInterfaceRequirements	M	0..N	VirtualNetworkInterfaceRequirements	Specifies requirements on a virtual network interface realizing the CPs instantiated from this CPD. See notes 1 and 5.
(inherited attributes)				All attributes inherited from Cpd. See note 2.

NOTE 1: In case of referencing an intCpd via its identifier, the virtualNetworkInterfaceRequirements attribute of the referenced intCpd applies.
NOTE 2: For CPs exposed by VNFs realized only by one or set of OS containers and used by the OS containers to connect to the primary container cluster external network, the ability to configure virtualised resources based on cpRole and trunkMode attributes might not be supported by all container technologies.
NOTE 3: For CPs exposed by VNFs realized only by one or a set of OS containers, the ability to configure virtualised resources based on this attribute might not be supported by all container technologies.
NOTE 4: One and only one of the following attributes shall be present: intVirtualLinkDesc or intCpd or vipCpd or virtualCpd.
NOTE 5: A VNFD conformant to the present document shall not specify "virtualNetworkInterfaceRequirements" in VnfExtCpds corresponding to primary container cluster network interfaces.

7.1.3.3 AddressData information element

7.1.3.3.1 Description

The AddressData information element supports providing information about the addressing scheme and parameters applicable to a CP.

7.1.3.3.2 Attributes

The attributes of the AddressData information element shall follow the indications provided in table 7.1.3.3.2-1.

Table 7.1.3.3.2-1: Attributes of the AddressData information element

Attribute	Qualifier	Cardinality	Content	Description
addressType	M	1	Enum	Describes the type of the address to be assigned to the CP instantiated from the parent CPD. VALUES: • MAC address • IP address • etc. The content type shall be aligned with the address type supported by the layerProtocol attribute of the parent CPD.

Attribute	Qualifier	Cardinality	Content	Description
l2AddressData	M	0..1	L2AddressData	Provides the information on the MAC addresses to be assigned to the CP(s) instantiated from the parent CPD. Shall be present when the addressType is MAC address.
l3AddressData	M	0..1	L3AddressData	Provides the information on the IP addresses to be assigned to the CP instantiated from the parent CPD. Shall be present when the addressType is IP address. See clause 7.1.3.4.

7.1.3.4 L3AddressData information element

7.1.3.4.1 Description

The L3AddressData information element supports providing information about Layer 3 level addressing scheme and parameters applicable to a CP.

7.1.3.4.2 Attributes

The attributes of the L3AddressData information element shall follow the indications provided in table 7.1.3.4.2-1.

Table 7.1.3.4.2-1: Attributes of the L3AddressData information element

Attribute	Qualifier	Cardinality	Content	Description
ipAddressAssignment	M	1	Boolean	Specify which mode is used for the IP address assignment. If it is set to True and this flag is not used in the context of the VirtualCpd information element, IP configuration information shall be provided for the VNF by a management entity using the NFV MANO interfaces towards the VNFM. If it is set to True and this flag is used in the context of the VirtualCpd information element, IP configuration information should be provided for the VNF by a management entity using the NFV MANO interfaces towards the VNFM. If it is not provided, the CISM assigns an IP address. See note 5. If it is set to False, the value of the "ipAddressAssignmentSubtype" attribute defines the method of IP address assignment. Shall be present if the "fixedIpAddress" attribute is not present and should be absent otherwise. See note 3.

Attribute	Qualifier	Cardinality	Content	Description
ipAddressAssignmentSubtype	M	0..1	Enum	Method of IP address assignment in case the IP configuration is not provided using the NFV MANO interfaces towards the VNFM. Shall be present in case the "ipAddressAssignment attribute is set to "False" and shall be absent otherwise. VALUES: • DYNAMIC: the VNF gets an IP address that is dynamically assigned by the NFVI/VIM/CISM without receiving IP configuration information from the MANO interfaces • VNF_PKG: an IP address defined by the VNF provider is assigned by means included as part of the VNF package (e.g. LCM script) • EXTERNAL: an IP address is provided by an external management entity (such as EM) directly towards the VNF.
floatingIpActivated	M	0..1	Boolean	Specify if the floating IP scheme is activated on the CP or not. See notes 3, 4 and 6.
ipAddressType	M	0..1	Enum	Define address type. VALUES: • IPV4 • IPV6 See notes 1 and 3.
numberOfIpAddress	M	0..1	Integer	Minimum number of IP addresses to be assigned based on this L3AddressData information element. See note 3.
fixedIpAddress	M	0..N	String	Fixed IP addresses to be assigned to the internal CP instance. This attribute enables the VNF provider to define fixed IP addresses for internal CP instances to be assigned by the VNFM or the NFVO. See notes 2 and 3.

NOTE 1: The address type should be aligned with the address type supported by the layerProtocol attribute of the Cpd.
NOTE 2: This attribute is only permitted for Cpds without external connectivity, i.e. connectivity outside the VNF. If included, it shall be compatible with the values of the l3ProtocolData of the intVirtualLinkDesc referred to by the VduCpd, if l3ProtocolData is included in the VnfVirtualLinkDesc.
NOTE 3: If the fixedIpAddress attribute is included:
- the ipAddressAssignment attribute should not be present. If it is present in this context, its value has no meaning and shall be ignored when processing the VNFD. Using the ipAddressAssignment attribute in this context is deprecated and only provided for backward compatibility; implementations need to be aware that support can be removed in subsequent versions of the present document;
- the value of the floatingIpActivated attribute shall be set to false;
- the value of the ipAddressType attribute, if included, shall be set consistently with the fixedIpAddress;
- the value of the numberOfIpAddress attribute, if included, shall be set consistently with the cardinality of the fixedIpAddress.

NOTE 4: This attribute is only relevant when used in a VnfExtCpd. It shall be omitted or set to false otherwise.
NOTE 5: For VirtualCps exposed by a VNF component realized by one or more OS containers, if the used container technology does not support the capability to set a defined IP address and the "ipAddressAssignment" flag is set to "true", IP configuration information should not be provided for the VNF by a management entity using the NFV MANO interfaces towards the VNFM. If it is provided nevertheless, the CISM might not be able to assign that IP address to the VirtualCp instance.
NOTE 6: For CPs of a VNFC realized by one or a set of OS containers the ability to configure a floating IP address might not be supported by all container technologies.

7.1.3.5 L2AddressData information element

7.1.3.5.1 Description

The L2AddressData information element supports providing information about Layer 2 level addressing applicable to a CP.

7.1.3.5.2 Attributes

The attributes of the L2AddressData information element shall follow the indications provided in table 7.1.3.5.2-1.

Table 7.1.3.5.2-1: Attributes of the L2AddressData information element

Attribute	Qualifier	Cardinality	Content	Description
macAddressAssignment	M	1	Boolean	Specify which mode is used for the MAC address assignment. If it is set to True, a MAC address is expected to be provided by a management entity via the NFV MANO interfaces towards the VNFM using attributes standardized for this purpose in the NFV-MANO information model and is further transferred from the VNFM to the VIM/CISM. A MAC address will be automatically assigned by the VIM/NFVI/CISM as fallback if not provided. If it is set to False, a MAC address is expected to be assigned by means specific to the VNF itself (e.g. by an LCM script, by the EM) and is further transferred from the VNFM to the VIM/CISM. A MAC address will be automatically assigned by the VIM/NFVI/CISM as fallback if not provided to the VIM/CISM.

7.1.4 Void

7.1.5 Information elements related to the configuration of VNF lifecycle management operations

7.1.5.1 Introduction

This clause defines information elements which represent information to configure lifecycle management operations as specified in ETSI GS NFV-IFA 007 [i.3] and ETSI GS NFV-IFA 008 [i.4].

7.1.5.2 VnfLcmOperationsConfiguration information element

7.1.5.2.1 Description

This information element is a container for all attributes that affect the invocation of the VNF Lifecycle Management operations, structured by operation.

7.1.5.2.2 Attributes

The VnfLcmOperationsConfiguration information element shall follow the indications provided in table 7.1.5.2.2-1.

Table 7.1.5.2.2-1: Attributes of the VnfLcmOperationsConfiguration information element

Attribute	Qualifier	Cardinality	Content	Description
instantiateVnfOpConfig	M	0..1	InstantiateVnfOpConfig	Configuration parameters for the InstantiateVnf operation.
scaleVnfOpConfig	M	0..1	ScaleVnfOpConfig	Configuration parameters for the ScaleVnf operation.
scaleVnfToLevelOpConfig	M	0..1	ScaleVnfToLevelOpConfig	Configuration parameters for the ScaleVnfToLevel operation.
changeVnfFlavourOpConfig	M	0..1	ChangeVnfFlavourOpConfig	Configuration parameters for the ChangeVnfFlavour operation.
healVnfOpConfig	M	0..1	HealVnfOpConfig	Configuration parameters for the HealVnf operation.
terminateVnfOpConfig	M	0..1	TerminateVnfOpConfig	Configuration parameters for the TerminateVnf operation.
operateVnfOpConfig	M	0..1	OperateVnfOpConfig	Configuration parameters for the OperateVnf operation.
changeExtVnfConnectivityOpConfig	M	0..1	ChangeExtVnfConnectivity OpConfig	Configuration parameters for the ChangeExtVnfConnectivity operation.
createSnapshotVnfOpConfig	M	0..1	CreateSnapshotVnfOpConfig	Configuration parameters for the Create VNF Snapshot operation.
revertToSnapshotVnfOpConfig	M	0..1	RevertToSnapshotVnfOpConfig	Configuration parameters for the Revert-To VNF Snapshot operation.
changeCurrentVnfPackageOpConfig	M	0..N	ChangeCurrentVnfPackage OpConfig	Configuration parameters for the ChangeCurrentVnfPackage operation.

7.1.5.3 InstantiateVnfOpConfig information element

7.1.5.3.1 Description

This information element defines attributes that affect the invocation of the InstantiateVnf operation.

7.1.5.3.2 Attributes

The InstantiateVnfOpConfig information element shall follow the indications provided in table 7.1.5.3.2-1.

Table 7.1.5.3.2-1: Attributes of the InstantiateVnfOpConfig information element

Attribute	Qualifier	Cardinality	Content	Description
parameter	M	0..N	Not specified	Array of KVP requirements for VNF-specific parameters to be passed when invoking the InstantiateVnf operation. See note.
targetScaleLevelsSupported	M	0..1	Boolean	Signals whether target scale levels are supported by this VNF during instantiation. Default is FALSE, i.e. "not supported".
NOTE: It is assumed that the KVP requirements will be implicitly used to define the value type.				

7.1.5.4 ScaleVnfOpConfig information element

7.1.5.4.1 Description

This information element defines attributes that affect the invocation of the ScaleVnf operation.

7.1.5.4.2 Attributes

The ScaleVnfOpConfig information element shall follow the indications provided in table 7.1.5.4.2-1.

Table 7.1.5.4.2-1: Attributes of the ScaleVnfOpConfig information element

Attribute	Qualifier	Cardinality	Content	Description
parameter	M	0..N	Not specified	Array of KVP requirements for VNF-specific parameters to be passed when invoking the ScaleVnf operation. See note.
scalingByMoreThanOneStepSupported	M	0..1	Boolean	Signals whether passing a value larger than one in the numberOfSteps parameter of the ScaleVnf operation is supported by this VNF. Default is FALSE, i.e. "not supported".
NOTE: It is assumed that the KVP requirements will be implicitly used to define the value type.				

7.1.5.5 ScaleVnfToLevelOpConfig information element

7.1.5.5.1 Description

This information element defines attributes that affect the invocation of the ScaleVnfToLevel operation.

7.1.5.5.2 Attributes

The ScaleVnfToLevelOpConfig information element shall follow the indications provided in table 7.1.5.5.2-1.

Table 7.1.5.5.2-1: Attributes of the ScaleVnfToLevelOpConfig information element

Attribute	Qualifier	Cardinality	Content	Description
parameter	M	0..N	Not specified	Array of KVP requirements for VNF-specific parameters to be passed when invoking the ScaleVnfToLevel operation. See note.
arbitraryTargetLevelsSupported	M	1	Boolean	Signals whether scaling according to the parameter "scaleInfo" is supported by this VNF.
NOTE: It is assumed that the KVP requirements will be implicitly used to define the value type.				

7.1.5.6 HealVnfOpConfig information element

7.1.5.6.1 Description

This information element defines attributes that affect the invocation of the HealVnf operation.

7.1.5.6.2 Attributes

The HealVnfOpConfig information element shall follow the indications provided in table 7.1.5.6.2-1.

Table 7.1.5.6.2-1: Attributes of the HealVnfOpConfig information element

Attribute	Qualifier	Cardinality	Content	Description
parameter	M	0..N	Not specified	Array of KVP requirements for VNF-specific parameters to be passed when invoking the HealVnf operation. See note.
cause	M	0..N	String	Supported "cause" parameter values.
NOTE: It is assumed that the KVP requirements will be implicitly used to define the value type.				

7.1.5.7 TerminateVnfOpConfig information element

7.1.5.7.1 Description

This information element defines attributes that affect the invocation of the TerminateVnf operation.

7.1.5.7.2 Attributes

The TerminateVnfOpConfig information element shall follow the indications provided in table 7.1.5.7.2-1.

Table 7.1.5.7.2-1: Attributes of the TerminateVnfOpConfig information element

Attribute	Qualifier	Cardinality	Content	Description
minGracefulTerminationTimeout	M	1	Number	Minimum timeout value for graceful termination of a VNF instance.
maxRecommendedGracefulTerminationTimeout	M	0..1	Number	Maximum recommended timeout value that can be needed to gracefully terminate a VNF instance of a particular type under certain conditions, such as maximum load condition. This is provided by VNF provider as information for the operator facilitating the selection of optimal timeout value. This value is not used as constraint.
parameter	M	0..N	Not specified	Array of KVP requirements for VNF-specific parameters to be passed when invoking the TerminateVnf operation. See note.
NOTE: It is assumed that the KVP requirements will be implicitly used to define the value type.				

7.1.5.8 OperateVnfOpConfig information element

7.1.5.8.1 Description

This information element defines attributes that affect the invocation of the OperateVnf operation.

7.1.5.8.2 Attributes

The OperateVnfOpConfig information element shall follow the indications provided in table 7.1.5.8.2-1.

Table 7.1.5.8.2-1: Attributes of the OperateVnfOpConfig information element

Attribute	Qualifier	Cardinality	Content	Description
minGracefulStopTimeout	M	1	Number	Minimum timeout value for graceful stop of a VNF instance.
maxRecommendedGracefulStopTimeout	M	0..1	Number	Maximum recommended timeout value that can be needed to gracefully stop a VNF instance of a particular type under certain conditions, such as maximum load condition. This is provided by VNF provider as information for the operator facilitating the selection of optimal timeout value. This value is not used as constraint.
parameter	M	0..N	Not specified	Array of KVP requirements for VNF-specific parameters to be passed when invoking the OperateVnf operation. See note.
NOTE: It is assumed that the KVP requirements will be implicitly used to define the value type.				

7.1.5.9 ChangeVnfFlavourOpConfig information element

7.1.5.9.1 Description

This information element defines attributes that affect the invocation of the ChangeVnfFlavour operation.

7.1.5.9.2 Attributes

The ChangeVnfFlavourOpConfig information element shall follow the indications provided in table 7.1.5.9.2-1.

Table 7.1.5.9.2-1: Attributes of the ChangeVnfFlavourOpConfig information element

Attribute	Qualifier	Cardinality	Content	Description
parameter	M	0..N	Not specified	Array of KVP requirements for VNF-specific parameters to be passed when invoking the ChangeVnfFlavour operation. See note.
NOTE: It is assumed that the KVP requirements will be implicitly used to define the value type.				

7.1.5.10 ChangeExtVnfConnectivityOpConfig information element

7.1.5.10.1 Description

This information element defines attributes that affect the invocation of the ChangeExtVnfConnectivity operation.

7.1.5.10.2 Attributes

The ChangeExtVnfConnectivityOpConfig information element shall follow the indications provided in table 7.1.5.10.2-1.

Table 7.1.5.10.2-1: Attributes of the ChangeExtVnfConnectivityOpConfig information element

Attribute	Qualifier	Cardinality	Content	Description
parameter	M	0..N	Not specified	Array of KVP requirements for VNF-specific parameters to be passed when invoking the ChangeExtVnfConnectivity operation. See note.
NOTE: It is assumed that the KVP requirements will be implicitly used to define the value type.				

7.1.5.11 CreateSnapshotVnfOpConfig information element

7.1.5.11.1 Description

This information element defines attributes that affect the invocation of the Create VNF Snapshot operation.

7.1.5.11.2 Attributes

The SnapshotVnfOpConfig information element shall follow the indications provided in table 7.1.5.11.2-1.

Table 7.1.5.11.2-1: Attributes of the CreateSnapshotVnfOpConfig information element

Attribute	Qualifier	Cardinality	Content	Description
parameter	M	1..N	Not specified	Array of KVP requirements for VNF-specific parameters to be passed when invoking the CreateSnapshotVnfOpConfig operation. See note.
NOTE: It is assumed that the KVP requirements will be implicitly used to define the value type.				

7.1.5.12 RevertToSnapshotVnfOpConfig information element

7.1.5.12.1 Description

This information element defines attributes that affect the invocation of the Revert-To VNF Snapshot operation.

7.1.5.12.2 Attributes

The SnapshotVnfOpConfig information element shall follow the indications provided in table 7.1.5.12.2-1.

Table 7.1.5.12.2-1: Attributes of the RevertToSnapshotVnfOpConfig information element

Attribute	Qualifier	Cardinality	Content	Description
parameter	M	1..N	Not specified	Array of KVP requirements for VNF-specific parameters to be passed when invoking the RevertToSnapshotVnfOpConfig operation. See note.
NOTE: It is assumed that the KVP requirements will be implicitly used to define the value type.				

7.1.5.13 ChangeCurrentVnfPackageOpConfig information element

7.1.5.13.1 Description

This information element defines attributes that affect the invocation of the change current VNF Package operation.

7.1.5.13.2 Attributes

The ChangeCurrentVnfPackageOpConfig information element shall follow the indications provided in table 7.1.5.13.2-1.

Table 7.1.5.13.2-1: Attributes of the ChangeCurrentVnfPackageOpConfig information element

Attribute	Qualifier	Cardinality	Content	Description
opConfigId	M	1	Identifier	Identifier of this parameter set for later referencing.
parameter	M	0..N	Not specified	Array of KVP requirements for VNF-specific parameters to be passed when invoking the change current VNF Package operation. See note.
NOTE: It is assumed that the KVP requirements will be implicitly used to define the value type.				

7.1.6 Information elements related to the Vdu

7.1.6.1 Introduction

The clauses below define the information elements related to the Vdu.

7.1.6.2 Vdu information element

7.1.6.2.1 Description

The Virtualisation Deployment Unit (VDU) is a construct supporting the description of the deployment and operational behaviour of a VNFC.

A VNFC instance created based on the VDU maps to a single instance of atomic deployable unit, represented by a single VM for hypervisor-based virtualisation, or represented by one or a set of OS containers for OS virtualisation.

A UML representation of the Vdu high-level structure is shown in figure 7.1.6.2.1-1.

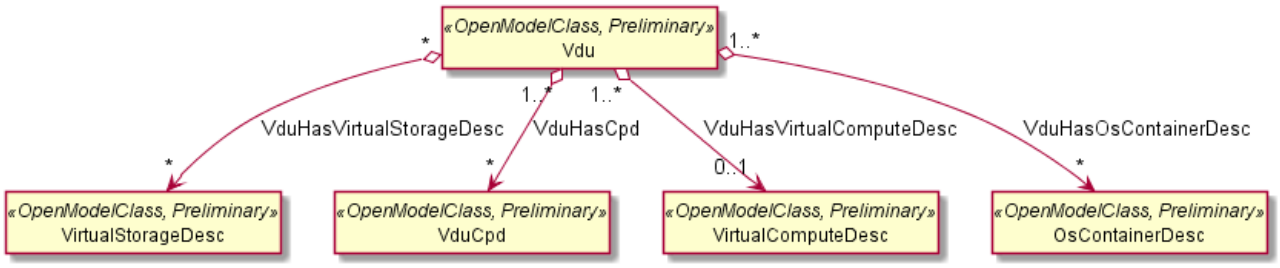

Figure 7.1.6.2.1-1: Vdu deployment view

7.1.6.2.2 Attributes

The attributes of the Vdu information element shall follow the indications provided in table 7.1.6.2.2-1.

Table 7.1.6.2.2-1: Attributes of the Vdu information element

Attribute	Qualifier	Cardinality	Content	Description
vduId	M	1	Identifier	Unique identifier of this Vdu in VNFD.
name	M	1	String	Human readable name of the Vdu.
description	M	1	String	Human readable description of the Vdu.
intCpd	M	0..N	VduCpd	Describes network connectivity between a VNFC instance (based on this Vdu) and an internal Virtual Link (VL). See clause 7.1.6.4. See note 7.
virtualComputeDesc	M	0..1	Identifier (Reference to VirtualComputeDesc)	Describes CPU, Memory and acceleration requirements of the single VM realizing this Vdu. See clause 7.1.9.2.2. See note 6.
osContainerDesc	M	0..N	Identifier (Reference to OsContainerDesc)	Describes CPU, memory requirements and limits, and software images of the OS Containers realizing this Vdu corresponding to OS Containers sharing the same host and same network namespace. Each unique identifier is referenced only once within one VDU. See clause 7.1.6.13. See note 6.
virtualStorageDesc	M	0..N	Identifier (Reference to VirtualStorageDesc)	Describes storage requirements for a VirtualStorage instance attached to the virtualisation container(s) created from virtualComputeDesc or osContainerDesc defined for this Vdu. See clause 7.1.9.4.
bootOrder	M	0..N	Not specified	Describes the boot index (lowest index defines highest boot priority) of the referenced descriptors from which a valid boot device is created e.g. VirtualStorageDesc from which a VirtualStorage instance is created. See note 1 and note 6.
swImageDesc	M	0..1	Identifier (Reference to SwImageDesc)	Describes the software image which is directly loaded on the virtualisation container realizing this Vdu. See clause 7.1.6.5. See note 2 and note 6.
nfviConstraint	M	0..N	String	Describes constraints on the NFVI for the VNFC instance(s) created from this Vdu. For example, aspects of a secure hosting environment for the VNFC instance that involve additional entities or processes. See note 3.

Attribute	Qualifier	Cardinality	Content	Description
monitoringParameter	M	0..N	MonitoringParameter	Specifies the virtualised resource related performance metrics on the VDU level to be tracked by the VNFM. MonitoringParameter is defined in clause 7.1.11.3. See note 6.
configurableProperties	M	0..1	VnfcConfigurableProperties	Describes the configurable properties of all VNFC instances based on this VDU. See clause 7.1.6.7. Cardinality 0 is used when the VNFCs do not have configurable properties.
bootData	M	0..1	Not specified	Contains a string or a URL to a file contained in the VNF package used to customize a virtualised compute resource at boot time. The bootData may contain variable parts that are replaced by deployment specific values before being sent to the VIM. See notes 4 and 6.
trunkPort	M	0..N	TrunkPortTopology	Specifies the logical topology between an intCpd in trunk mode, used to describe a trunk port, and other intCpds used to describe subports of the same trunk. Cardinality 0 is used when there is no intCpd with trunkmode = True, or when no individual intCpds to describe the subports are included in the Vdu. See note 5.
logicalNode	M	0..N	LogicalNodeRequirements	The logical node requirements. See note 6.
requestAdditionalCapabilities	M	0..N	RequestedAdditionalCapabilityData	Specifies requirements for additional capabilities. These can be for a range of purposes. One example is acceleration related capabilities. See clause 7.1.9.5. See note 6
mcioConstraintParams	M	0..N	Enum	The parameter names for constraints expected to be assigned to MCIOs realizing this Vdu. The value specifies the standardized semantical context of the MCIO constraints and the parameter names for the MCIO constraints in the MCIO declarative descriptor. The mcioConstraintParams attribute shall have one of the following values, expressing the associated semantical context. VALUES: • affinityNfviPop • affinityZone • affinityZoneGroup • affinityNfviNode • affinityCisNode • antiAffinityNfviPop • antiAffinityZone • antiAffinityZoneGroup • antiAffinityNfviNode • antiAffinityCisNode • localAffinityNfviPop • localAffinityZone • localAffinityZoneGroup • localAffinityNfviNode • localAffinityCisNode • localAntiAffinityNfviPop • localAntiAffinityZone • localAntiAffinityZoneGroup • localAntiAffinityNfviNode • localAntiAffinityCisNode

Attribute	Qualifier	Cardinality	Content	Description
				• nodeAdditionalCapabilitySsd • nodeAdditionalCapabilityDpdk • nodeAdditionalCapabilitySriov • nodeAdditionalCapabilityGpu • nodeAdditionalCapabilityFpga • nodeAdditionalCapabilityCpuPin • nodeCapabilityLogicalNuma • nodePool For the associated semantical context of the values, refer to the description under the table.
mcioIdentificationData	M	0..1	Not specified	Name and type of the MCIO that realizes this VDU. It allows the VNFM to identify the MCIO e.g. when querying the CISM. It shall be present when the VDU is realized by one or a set of OS containers and shall be absent otherwise.

NOTE 1: If no boot order is defined the default boot order defined in the VIM or NFVI shall be used.
NOTE 2: More software images can be attached to the virtualisation container using VirtualStorage resources. See clause 7.1.9.4.
NOTE 3: These are constraints other than stipulating that a VNFC instance has access to a certain resource, as a prerequisite to instantiation. The attributes virtualComputeDesc and virtualStorageDesc define the resources required for instantiation of the VNFC instance.
NOTE 4: The parameters of each variable part shall be declared (1) in the VnfLcmOperationsConfiguration information element (see clause 7.1.5.2) as "volatile" parameters available to the bootData template during the respective VNF lifecycle management operation execution and/or (2) in the extension attribute of the VnfInfoModifiableAttributes information element (see clause 7.1.14) or in the VnfConfigurableProperties information element (see clause 7.1.12) as "persistent" parameters available to the bootData template during the lifetime of the VNF instance. For VNF lifecycle management operations resulting in multiple VNFC instantiations, the VNFM supports the means to provide the appropriate parameters to appropriate VNFC instances.
NOTE 5: Subport instances created dynamically do not require a dedicated intCpd different to the trunk port cpd.
NOTE 6: Only one of virtualComputeDesc or osContainerDesc shall be part of a Vdu. If the Vdu includes osContainerDesc, then bootOrder, swImageDesc, monitoringParameters and bootData shall not be present in the Vdu. If the Vdu includes virtualComputeDesc, then logicalNode and requestedAdditionalCapabilites shall not be present in the Vdu.
NOTE 7: If the VDU is realized by a VM this attribute shall be present.
If the VDU is realized by one or a set of OS containers, the presence of this attribute depends on the OS containers connectivity:
- If the OS containers only connect to the container cluster internal network, this attribute may be absent. If present, the ability to configure virtualised resources based on this attribute might not be supported by all container technologies.
- If the OS containers also connect to container cluster external networks, this attribute may be present for CPs used to connect to the primary container cluster external network and shall be present for CPs used to connect to the secondary container cluster external network.

The values of the mcioConstraintParams attributes express the following semantical context:

- affinityNfviPop: Inter-MCIO affinity placement on the same NFVI-PoP
- affinityZone: Inter-MCIO affinity placement in the same Zone
- affinityZoneGroup: Inter-MCIO affinity placement in the same ZoneGroup
- affinityNfviNode: Inter-MCIO affinity placement on the same NFVI-node
- affinityCisNode: Inter-MCIO affinity placement on the same CIS-node
- antiAffinityNfviPop: Inter-MCIO anti-affinity placement on different NFVI-PoPs
- antiAffinityZone: Inter-MCIO anti-affinity placement in different Zones
- antiAffinityZoneGroup: Inter-MCIO anti-affinity placement in different ZoneGroups
- antiAffinityNfviNode: Inter-MCIO anti-affinity placement on different NFVI-nodes
- antiAffinityCisNode: Inter-MCIO anti-affinity placement on different CIS-nodes
- localAffinityNfviPop: MCIO instances affinity placement on the same NFVI-PoP
- localAffinityZone: MCIO instances affinity placement in the same Zone
- localAffinityZoneGroup: MCIO instances affinity placement in the same ZoneGroup
- localAffinityNfviNode: MCIO instances affinity placement on the same NFVI-node
- localAffinityCisNode: MCIO instances affinity placement on the same CIS-node
- localAntiAffinityNfviPop: MCIO instances anti-affinity placement on different NFVI-PoPs

- localAntiAffinityZone: MCIO instances anti-affinity placement in different Zones
- localAntiAffinityZoneGroup: MCIO instances anti-affinity placement in different ZoneGroups
- localAntiAffinityNfviNode: MCIO instances anti-affinity placement on different NFVI-nodes
- localAntiAffinityCisNode: MCIO instances anti-affinity placement on different CIS-nodes
- nodeAdditionalCapabilitySsd: MCIO additional node capability requirement for SSD device
- nodeAdditionalCapabilityDpdk: MCIO additional node capability requirement for DPDK driver
- nodeAdditionalCapabilitySriov: MCIO additional node capability requirement for SR-IOV
- nodeAdditionalCapabilityGpu: MCIO additional node capability requirement for GPU acceleration device
- nodeAdditionalCapabilityFpga: MCIO additional node capability requirement for FPGA
- nodeAdditionalCapabilityCpuPin: MCIO additional node capability requirement for CPU pinning
- nodeCapabilityLogicalNuma: MCIO logical node hardware capability requirement for NUMA architecture
- nodePool: Pool of container cluster nodes with the same capabilities

7.1.6.3 Cpd information element

7.1.6.3.1 Description

A Cpd information element describes network connectivity to a compute resource or a VL. This is an abstract class used as parent for the various Cpd classes.

7.1.6.3.2 Attributes

The attributes of the Cpd information element shall follow the indications provided in table 7.1.6.3.2-1.

Table 7.1.6.3.2-1: Attributes of the Cpd information element

Attribute	Qualifier	Cardinality	Content	Description
cpdId	M	1	Identifier	Identifier of this Cpd information element.
layerProtocol	M	1..N	Enum	Specifies which protocol the CP uses for connectivity purposes. VALUES: • Ethernet • MPLS • ODU2 • IPV4 • IPV6 • Pseudo-Wire • etc. See notes 1, 2 and 3.
ipStackMode	M	0..1	Enum	Specifies the capability of the CP to support IP dual stack or tunnelling. Values: • IPV4 XOR IPV6: the CP supports both IPV4 and IPV6 but it can only be configured with IPV4 or with IPV6. • IPV4 OR IPV6: the CP supports both IPV4 and IPV6 and it can be configured with either of them or with both. • TUNNEL_IPV6_OVER_IPV4: the CP supports IPv6 tunnelling over IPv4. • TUNNEL_IPV4_OVER_IPV6: the CP supports IPv4 tunnelling over IPv6. See notes 2 and 3.
cpRole	M	0..1	String	Specifies the role of the port in the context of the traffic flow patterns in the VNF or parent NS. For example a VNF with a tree flow pattern within the VNF will have legal cpRoles of ROOT and LEAF.

Attribute	Qualifier	Cardinality	Content	Description
description	M	0..1	String	Provides human-readable information on the purpose of the CP (e.g. CP for control plane traffic).
cpProtocol	M	0..N	CpProtocolData	Specifies the protocol layering information the CP uses for connectivity purposes and associated information. There shall be one cpProtocol for each layer protocol as indicated by the attribute layerProtocol. When a PnfExtCpd as defined in ETSI GS NFV-IFA 014 [i.8] is inherited from this Cpd, the cardinality is set to 0.
trunkMode	M	0..1	Boolean	Information about whether the CP instantiated from this CPD is in Trunk mode (802.1Q or other). When operating in "trunk mode", the Cp is capable of carrying traffic for several VLANs. A cardinality of 0 implies that trunkMode is not configured for the Cp i.e. It is equivalent to Boolean value "false".
securityGroupRuleId	M	0..N	Identifier (Reference to SecurityGroupRule)	Reference of the security group rules bound to this CPD.
NOTE 1: This information determines, amongst other things, which type of address to assign to the access point at instantiation time. NOTE 2: If multiple values are indicated in the layerProtocol attribute they represent layers of a protocol stack with the top layer first, with the exception of IPV4 and IPV6 values, when used as consecutive values, in which case the interpretation of their presence shall be specified through the IpStackMode attribute. NOTE 3: The ipStackMode attribute shall be present If layerProtocol indicates both IPV4 and IPV6 and these are listed as consecutive values and shall be absent otherwise.				

7.1.6.4 VduCpd information element

7.1.6.4.1 Description

A VduCpd information element is a type of Cpd and describes network connectivity between a VNFC instance (based on this VDU) and a particular VL.

A VduCpd inherits from the Cpd Class (see clause 7.1.6.3). All attributes of the Cpd are also attributes of the VduCpd.

7.1.6.4.2 Attributes

The attributes of the VduCpd information element shall follow the indications provided in table 7.1.6.4.2-1.

Table 7.1.6.4.2-1: Attributes of the VduCpd information element

Attribute	Qualifier	Cardinality	Content	Description
intVirtualLinkDesc	M	0..1	Identifier (Reference to VnfVirtualLinkDesc)	Reference of the internal VLD which this internal CPD connects to. See note 2.
bitrateRequirement	M	0..1	Number	Bitrate requirement on this CP. See note 2.
virtualNetworkInterfaceRequirements	M	0..N	VirtualNetworkInterfaceRequirements	Specifies requirements on a virtual network interface realizing the CPs instantiated from this CPD. See notes 2 and 4.
order	M	0..1	Integer	The order of the NIC to be assigned on the compute instance (e.g. 2 for eth2). See notes 1 and 2. If the property is not present, it shall be left to the VIM to assign a value when creating the instance.

Attribute	Qualifier	Cardinality	Content	Description
vnicType	M	0..1	Enum	Describes the type of the virtual network interface realizing the CPs instantiated from this CPD. This is used to determine which mechanism driver(s) to be used to bind the port. VALUES: • NORMAL • MACVTAP • DIRECT • BAREMETAL • VIRTIO-FORWARDER • DIRECT-PHYSICAL • SMART-NIC Additional values of the attribute for VDUs realized by one or set of OS containers: • BRIDGE • IPVLAN • LOOPBACK • MACVLAN • PTP • VLAN • HOST-DEVICE • etc.
(inherited attributes)				All attributes inherited from Cpd. See note 3.

NOTE 1: When binding more than one port to a single compute (a.k.a multi vNICs) and ordering is desired, it is mandatory that all ports will be set with an order value. The order values shall represent a positive, arithmetic progression that starts with 0 (i.e. 0, 1, 2, ..., n).
NOTE 2: For VDUs realized by one or a set of OS containers, the ability to configure virtualised resources based on this attribute might not be supported by all container technologies.
NOTE 3: For CPs of VDUs realized by one or set of OS containers and used by the OS containers to connect to the primary container cluster external network, the ability to configure virtualised resources based on cpRole and trunkMode attributes might not be supported by all container technologies.
NOTE 4: A VNFD conformant to the present document shall not specify "virtualNetworkInterfaceRequirements" in VduCpds corresponding to primary container cluster network interfaces.

7.1.6.5 SwImageDesc information element

7.1.6.5.1 Description

This information element describes the software image for a particular VM-based VDU, OS Container or a virtual storage resource.

7.1.6.5.2 Attributes

The attributes of the SwImageDesc information element shall follow the indications provided in table 7.1.6.5.2-1.

Table 7.1.6.5.2-1: Attributes of the SwImageDesc information element

Attribute	Qualifier	Cardinality	Content	Description
id	M	1	Identifier	The identifier of this software image.
name	M	1	String	The name of this software image.
version	M	1	Version	The version of this software image.
checksum	M	0..1	ChecksumData	The checksum of the software image file. See note 3.
containerFormat	M	1	String	The container format describes the container file format in which software image is provided.
diskFormat	M	0..1	String	The disk format of a software image is the format of the underlying disk image. See note 1.
minDisk	M	0..1	Number	The minimal disk size requirement for this software image. The value of the "size of storage" attribute of the VirtualStorageDesc referencing this SwImageDesc shall not be smaller than the value of minDisk. See note 1.
minRam	M	0..1	Number	The minimal RAM requirement for this software image. The value of the "size" attribute of VirtualMemoryData of the Vdu referencing this SwImageDesc shall not be smaller than the value of minRam. See note 2.
size	M	0..1	Number	The size of this software image file. See note 3.
swImage	M	1	Identifier (Reference to SwImage)	This is a reference to the actual software image. The reference can be relative to the root of the VNF Package or can be a URL.
operatingSystem	M	0..1	String	Specifies the operating system used in the software image. This attribute may also identify if a 32 bit or 64 bit software image is used.
supportedVirtualisationEnvironment	M	0..N	String	Specifies the virtualisation environments (e.g. hypervisor) compatible with this software image.

NOTE 1: The attribute shall be present for VM-based software images referenced from a Vdu or from a VirtualStorageDesc, and shall be absent otherwise.
NOTE 2: The attribute may be present for VM-based software images referenced from a Vdu or from a VirtualStorageDesc, and shall be absent otherwise.
NOTE 3: The attribute shall be present for VM-based software images referenced from a Vdu or from a VirtualStorageDesc, and may be present otherwise.

7.1.6.6 VirtualNetworkInterfaceRequirements information element

7.1.6.6.1 Description

This information element specifies requirements on a virtual network interface.

7.1.6.6.2 Attributes

The attributes of the VirtualNetworkInterfaceRequirements information element shall follow the indications provided in table 7.1.6.6.2-1.

Table 7.1.6.6.2-1: Attributes of the VirtualNetworkInterfaceRequirements information element

Attribute	Qualifier	Cardinality	Content	Description
name	M	0..1	String	Provides a human readable name for the requirement.
description	M	0..1	String	Provides a human readable description of the requirement.
standardizedNetworkInterfaceRequirements	M	0..1	Not specified	The requirements on standardized network interface capabilities, e.g. SR-IOV or secondary container cluster network interface deployment requirements. See note.
networkInterfaceRequirements	M	0..1	Not specified	The additional network interface requirements beyond those specified in the standardizedNetworkInterfaceRequirements attribute. An element from an array of key-value pairs that articulate the network interface deployment requirements. See note.
nicIoRequirements	M	0..1	Identifier (Reference to LogicalNodeRequirements)	This references (couples) the CPD with any logical node I/O requirements (for network devices) that may have been created. Linking these attributes is necessary so that I/O requirements that need to be articulated at the logical node level can be associated with the network interface requirements associated with the CPD. See note.
NOTE: At least one of the attributes "standardizedNetworkInterfaceRequirements", "networkInterfaceRequirements", "nicIoRequirements" shall be present.				

7.1.6.7 VnfcConfigurableProperties information element

7.1.6.7.1 Description

This information element provides a means to define additional VNF-specific attributes that represent the configurable properties of a VNFC. For a VNFC instance, the values of these properties can be queried and modified through the VNFM. Modifying these values affects directly the configuration of an existing VNFC instance.

7.1.6.7.2 Attributes

The attributes of the VnfcConfigurableProperties information element shall follow the indications provided in table 7.1.6.7.2-1.

Table 7.1.6.7.2-1: Attributes of the VnfcConfigurableProperties information element

Attribute	Qualifier	Cardinality	Content	Description
additionalVnfcConfigurableProperty	M	0..N	Not specified	It provides VNFC configurable properties that can be modified using the ModifyVnfInfo operation.

7.1.6.8 CpProtocolData information element

7.1.6.8.1 Description

A CpProtocolData information element describes and associates the protocol layer that a CP uses together with other protocol and connection point information.

7.1.6.8.2 Attributes

The attributes of the CpProtocolData information element shall follow the indications provided in table 7.1.6.8.2-1.

Table 7.1.6.8.2-1: Attributes of the CpProtocolData information element

Attribute	Qualifier	Cardinality	Content	Description
associatedLayerProtocol	M	1	Enum	One of the values of the attribute layerProtocol of the Cpd IE (refer to clause 7.1.6.3). VALUES: • Ethernet • MPLS • ODU2 • IPV4 • IPV6 • Pseudo-Wire • Etc.
addressData	M	0..N	AddressData	Provides information on the addresses to be assigned to the CP(s) instantiated from the CPD.

7.1.6.9 SecurityGroupRule information element

7.1.6.9.1 Description

The SecurityGroupRule information element describes the details of a security group rule. Security group rule specifies the matching criteria for the ingress and/or egress traffic to/from the visited connection points. The design of security group rule follows a permissive model where all security group rules applied to a CP are dealt with in an "OR" logic fashion, i.e. the traffic is allowed if it matches any security group rule applied to this CP.

NOTE: For VDUs based on one or set of OS containers, associating different security group rules to different CPs of a VDU might not be supported by all underlying container technologies. In such cases all CPs of a VDU are expected to reference the same security group rule or the same set of security group rules or no security group rule.

7.1.6.9.2 Attributes

The attributes of the SecurityGroupRule information element shall follow the indications provided in table 7.1.6.9.2-1.

Table 7.1.6.9.2-1: Attributes of the SecurityGroupRule information element

Attribute	Qualifier	Cardinality	Content	Description
securityGroupRuleId	M	1	Identifier	Identifier of this SecurityGroupRule information element. See note 3.
description	M	0..1	String	Human readable description of the security group rule.
direction	M	0..1	Enum	The direction in which the security group rule is applied. VALUES: • INGRESS • EGRESS Defaults to INGRESS. See note 1.
etherType	M	0..1	Enum	Indicates the protocol carried over the Ethernet layer. VALUES: • IPV4 • IPV6 Defaults to IPV4.
protocol	M	0..1	Enum	Indicates the protocol carried over the IP layer. Permitted values: any protocol defined in the IANA protocol registry [i.7]. VALUES: • TCP • UDP • ICMP • etc. Defaults to TCP.
portRangeMin	M	0..1	Integer	Indicates minimum port number in the range that is matched by the security group rule. See note 2. Defaults to 0.
portRangeMax	M	0..1	Integer	Indicates maximum port number in the range that is matched by the security group rule. See note 2. Defaults to 65535.

NOTE 1: The direction of INGRESS or EGRESS is specified against the associated CPD. I.e. INGRESS means the packets entering a CP created from this CPD, while EGRESS means the packets sent out of a CP created from this CPD.
NOTE 2: If a value is provided at design-time, this value may be overridden at run-time based on other deployment requirements or constraints.
NOTE 3: Different VduCpd or VnfExtCpd with the same value of securityGroupRuleId imply they belong to the same security group.

7.1.6.10 ChecksumData information element

7.1.6.10.1 Description

The ChecksumData information element supports providing information about the result of performing a checksum operation over some arbitrary data.

7.1.6.10.2 Attributes

The attributes of the ChecksumData information element shall follow the indications provided in table 7.1.6.10.2-1.

Table 7.1.6.10.2-1: Attributes of the ChecksumData information element

Attribute	Qualifier	Cardinality	Content	Description
algorithm	M	1	String	Species the algorithm used to obtain the checksum value. See note
hash	M	1	String	Contains the result of applying the algorithm indicated by the algorithm attribute to the data to which this ChecksumData refers.
NOTE: The algorithm attribute value shall be one of the Hash Function Textual Names present in [2].				

7.1.6.11 TrunkPortTopology information element

7.1.6.11.1 Description

The TrunkPortTopology information element specifies the logical topology between an intCpd in trunk mode, used to describe a trunk port, and other intCpds used to describe subports of the same trunk. This information is used to request the VIM to create a trunk resource and add each CP instance initiated from a specific intCpd into the trunk, either as parent port role or as subport. Subport instances created dynamically do not require a dedicated intCpd different to the trunk port cpd.

7.1.6.11.2 Attributes

The attributes of the trunkPortTopology information element shall follow the indications provided in table 7.1.6.11.2-1.

Table 7.1.6.11.2-1: Attributes of the trunkPortTopology information element

Attribute	Qualifier	Cardinality	Content	Description
parentPortCpd	M	1	Identifier (Reference to VduCpd)	Reference of the internal VDU CPD which is used to instantiate the parent port in a logical trunk model.
subportList	M	1..N	Subport	Provides information used for the subport.

7.1.6.12 Subport information element

7.1.6.12.1 Description

The Subport information element specifies the information used for the subport of a trunk parent port.

7.1.6.12.2 Attributes

The attributes of the Subport information element shall follow the indications provided in table 7.1.6.12.2-1.

Table 7.1.6.12.2-1: Attributes of the Subport information element

Attribute	Qualifier	Cardinality	Content	Description
subportCpd	M	1	Identifier (Reference to VduCpd)	Reference of the internal VDU CPD which is used to instantiate the subport in a logical trunk model. See note.
segmentationType	M	0..1	Enum	Specifies the encapsulation type for the traffics coming in and out of the trunk subport. VALUES: • VLAN: the subport uses VLAN as encapsulation type. • INHERIT: the subport gets its segmentation type from the network it is connected to. Cardinality 0 means default value VLAN is used.
segmentationId	M	1	Integer	Specifies the segmentation ID for the subport, which is used to differentiate the traffics on different networks coming in and out of the trunk port. If a value is provided here it may be overridden by a value provided at run time when the infrastructure does not support mapping of segmentation IDs.
NOTE: The "trunkMode" attribute of the subportCpd shall be set as false.				

7.1.6.13 OsContainerDesc information element

7.1.6.13.1 Description

The OsContainerDesc information element describes the members properties of a set of co-located container compute resources when these are realizing a VDU.

7.1.6.13.2 Attributes

The attributes of the OsContainerDesc information element shall follow the indications provided in table 7.1.6.13.2-1.

Table 7.1.6.13.2-1: Attributes of the OsContainerDesc information element

Attribute	Qualifier	Cardinality	Content	Description
osContainerDescId	M	1	Identifier	Unique identifier of this OsContainerDesc in the VNFD.
name	M	1	String	Human readable name of this OS container.
description	M	1	String	Human readable description of this OS container.
requestedCpuResources	M	0..1	Integer	Number of CPU resources requested for the container (e.g. in milli-CPU-s).
requestedMemoryResources	M	0..1	Number	Amount of memory resources requested for the container (e.g. in MB).
requestedEphemeralStorageResources	M	0..1	Number	Size of ephemeral storage resources requested for the container (e.g. in GB).
extendedResourceRequests	M	0..N	Not specified	An array of key-value pairs of extended resources required by the container. See note.
cpuResourceLimit	M	0..1	Integer	Number of CPU resources the container can maximally use (e.g. in milli-CPU).
memoryResourceLimit	M	0..1	Number	Amount of memory resources the container can maximally use (e.g. in MB).
ephemeralStorageResourceLimit	M	0..1	Number	Size of ephemeral storage resources the container can maximally use (e.g. in GB).
hugePageResources	M	0..1	Not specified	Specifies HugePages resources requested for the container, which the container can maximally use (e.g. "hugepages-2Mi: 100Mi").
cpuPinningRequirements	M	0..1	VirtualCpuPinningData	Requirements for CPU pinning configuration for this OS container.
swImageDesc	M	1	Identifier (Reference to SwImageDesc)	Describes the software image realizing this OS container.
bootData	M	0..1	Not specified	Contains a string or a URL to a file contained in the VNF package used to customize a container resource at boot time. The bootData may contain variable parts that are replaced by deployment specific values before being sent.
monitoringParameters	M	0..N	MonitoringParameter	Specifies the virtualised resource related performance metrics on the OsContainerDesc level to be tracked by the VNFM. MonitoringParameter is defined in clause 7.1.11.3.
NOTE: Extended resources are to describe any type of resource provided by the container infrastructure. One example implementation of extended resources is "Extended Resources" in case the container infrastructure service is a Kubernetes® instance.				

7.1.7 Information elements related to the VLD

7.1.7.1 Introduction

The clauses below define the information elements related to the VLD.

7.1.7.2 VnfVirtualLinkDesc information element

7.1.7.2.1 Description

The VnfVirtualLinkDesc information element supports providing information about an internal VNF VL.

7.1.7.2.2 Attributes

The attributes of the VnfVirtualLinkDesc information element shall follow the indications provided in table 7.1.7.2.2-1.

Table 7.1.7.2.2-1: Attributes of the VnfVirtualLinkDesc information element

Attribute	Qualifier	Cardinality	Content	Description
virtualLinkDescId	M	1	Identifier	Unique identifier of this internal VLD in VNFD.
virtualLinkDescFlavour	M	1..N	VirtualLinkDescFlavour	Describes a specific flavour of the VL with specific bitrate requirements. See clause 7.1.8.5.
connectivityType	M	1	ConnectivityType	See clause 7.1.7.3.
testAccess	M	0..N	String	Specifies test access facilities expected on the VL (e.g. none, passive monitoring, or active (intrusive) loopbacks at endpoints).
description	M	0..1	String	Provides human-readable information on the purpose of the VL (e.g. control plane traffic).
monitoringParameter	M	0..N	MonitoringParameter	Specifies the virtualised resource related performance metrics on VLD level to be tracked by the VNFM. MonitoringParameter is defined in clause 7.1.11.3.
nfviMaintenanceInfo	M	0..1	NfviMaintenanceInfo	When present, provides information on the rules to be observed when an instance based on this VnfVirtualLinkDesc is impacted during NFVI operation and maintenance (e.g. NFVI resource upgrades). NfviMaintenanceInfo is defined in clause 7.1.8.17.
externallyManaged	M	0..1	Enum	Specifies the intent of the VNF designer with respect to the internal VL instances created from this descriptor being externally managed. VALUES: • REQUIRED • ALLOWED Defaults to ALLOWED. If the VNFD does not reference any LCM script and if the "vnfmInfo" attribute in the "Vnfd" information element indicates that the VNF can be managed by any ETSI NFV compliant VNFM, this attribute shall not be present.

7.1.7.3 ConnectivityType information element

7.1.7.3.1 Description

The ConnectivityType information element specifies the protocol exposed by a VL and the flow pattern supported by the VL.

7.1.7.3.2 Attributes

The attributes of the ConnectivityType information element shall follow the indications provided in table 7.1.7.3.2-1.

Table 7.1.7.3.2-1: Attributes of the ConnectivityType information element

Attribute	Qualifier	Cardinality	Content	Description
layerProtocol	M	1..N	Enum	Specifies the protocols that the VL uses. VALUES: • Ethernet • MPLS • ODU2 • IPV4 • IPV6 • Pseudo-Wire • Etc. See note 1 and note 2.
flowPattern	M	0..1	String	Specifies the flow pattern of the connectivity (Line, Tree, Mesh, etc.).
NOTE 1: The top layer protocol of the VL protocol stack shall always be provided. The lower layer protocols may be included when there are specific requirements on these layers. NOTE 2: If more than 1 values are present, the first value represents the highest layer protocol data, and the last value represents the lowest layer protocol data.				

7.1.8 Information elements related to the DeploymentFlavour

7.1.8.1 Introduction

The clauses below define the information elements related to the DF.

7.1.8.2 VnfDf information element

7.1.8.2.1 Description

The VnfDf information element describes a specific deployment version of a VNF.

7.1.8.2.2 Attributes

The attributes of the VnfDf information element shall follow the indications provided in table 7.1.8.2.2-1.

Table 7.1.8.2.2-1: Attributes of the VnfDf information element

Attribute	Qualifier	Cardinality	Content	Description
flavourId	M	1	Identifier	Identifier of this DF within the VNFD.
description	M	1	String	Human readable description of the DF.
vduProfile	M	1..N	VduProfile	Describes additional instantiation data for the VDUs used in this flavour.
virtualLinkProfile	M	0..N	VirtualLinkProfile	Defines the internal VLD along with additional data which is used in this DF. See notes 1 and 2.
vipCpProfile	M	0..N	VipCpProfile	Defines the minimum and maximum number of VIP CP instances created from each of the VipCpds used in this flavour. Shall be present if the deployment flavour can contain VIP CP instances.
mciopProfile	M	0..N	MciopProfile	Describes additional instantiation data for the MCIOPs used in this deployment flavour. This attribute shall be present if the DF references (via the vduProfile) containerized workloads based on a MCIOP.

Attribute	Qualifier	Cardinality	Content	Description
instantiationLevel	M	1..N	InstantiationLevel	Describes the various levels of resources that can be used to instantiate the VNF using this flavour. Examples: Small, Medium, Large. If there is only one "instantiationLevel" entry, it shall be treated as the default instantiation level for this DF.
defaultInstantiationLevelId	M	0..1	Identifier (Reference to InstantiationLevel)	References the "instantiationLevel" entry which defines the default instantiation level for this DF. It shall be present if there are multiple "instantiationLevel" entries.
supportedOperation	M	0..N	Enum	Indicates which operations are available for this DF via the VNF LCM interface. Instantiate VNF, Query VNF and Terminate VNF are supported in all DF and therefore need not be included in this list. VALUES: • Scale VNF • Scale VNF to Level • Heal VNF • Operate VNF • Etc.
vnfLcmOperationsConfiguration	M	1	VnfLcmOperationsConfiguration	Configuration parameters for the VNF Lifecycle Management operations.
affinityOrAntiAffinityGroup	M	0..N	AffinityOrAntiAffinityGroup	Specifies affinity or anti-affinity relationship applicable between the virtualisation containers (e.g. virtual machines) to be created using different VDUs or internal VLs to be created using different VnfVirtualLinkDesc(s) in the same affinity or anti-affinity group. See clause 7.1.8.12. See note 3.
vnfIndicator	M	0..N	VnfIndicator	Declares the VNF indicators that are supported by this VNF (specific to this DF).
supportedVnfInterface	M	0..N	VnfInterfaceDetails	Indicates which interfaces the VNF produces and provides additional details on how to access the interface endpoints.
supportedCoordinationActions	M	0..N	LcmCoordinationActionMapping	References applicable LCM coordination actions that can be invoked during each of the listed VNF LCM operations.
monitoringParameter	M	0..N	MonitoringParameter	Specifies the virtualised resource related performance metrics to be tracked by the VNFM. MonitoringParameter is defined in clause 7.1.11.3.
scalingAspect	M	0..N	ScalingAspect	The scaling aspects supported by this DF of the VNF. scalingAspect shall be present if the VNF supports scaling.
initialDelta	M	0..1	ScalingDelta	Represents the minimum size of the VNF (i.e. scale level zero for all scaling aspects). Shall be present if the "aspectDeltaDetails" attribute is present in the "ScalingAspect" information element.
dependencies	M	0..N	Dependencies	Specifies the order in which instances of the VNFCs have to be created.

NOTE 1: This allows for different VNF internal topologies between DFs.
NOTE 2: virtualLinkProfile needs to be provided for all VLs that the CPs of the VDUs in the VDU profiles connect to.
NOTE 3: In the present document, including either VDU(s) or VnfVirtualLinkDesc(s) into the same affinity or anti-affinity group is supported. Extension to support including both VDU(s) and VnfVirtualLinkDesc(s) into the same affinity or anti-affinity group is left for future specification.

7.1.8.3 VduProfile information element

7.1.8.3.1 Description

The VduProfile information element describes additional instantiation data for a given VDU used in a DF.

7.1.8.3.2 Attributes

The attributes of the VduProfile information element shall follow the indications provided in table 7.1.8.3.2-1.

Table 7.1.8.3.2-1: Attributes of the VduProfile information element

Attribute	Qualifier	Cardinality	Content	Description
vduId	M	1	Identifier (Reference to Vdu)	Uniquely references a VDU.
minNumberOfInstances	M	1	Integer	Minimum number of instances of the VNFC based on this VDU that is permitted to exist for this flavour. Shall be zero or greater.
maxNumberOfInstances	M	1	Integer	Maximum number of instances of the VNFC based on this VDU that is permitted to exist for this flavour. Shall be greater than zero.
localAffinityOrAntiAffinityRule	M	0..N	LocalAffinityOrAntiAffinityRule	Specifies affinity or anti-affinity rules applicable between the virtualisation containers (e.g. virtual machines) to be created based on this VDU. See clause 7.1.8.11. When the cardinality is greater than 1, both affinity rule(s) and anti-affinity rule(s) with different scopes (e.g. "Affinity with the scope resource zone and anti-affinity with the scope NFVI node") are applicable to the virtualisation containers (e.g. virtual machines) to be created based on this VDU.
affinityOrAntiAffinityGroupId	M	0..N	Identifier (Reference to AffinityOrAntiAffinityGroup)	References the affinity or anti-affinity group(s) the VDU belongs to. See note 1.
nfviMaintenanceInfo	M	0..1	NfviMaintenanceInfo	When present, provides information on the impact tolerance and rules to be observed when instance(s) of the VDU are impacted during NFVI operation and maintenance (e.g. NFVI resource upgrades). NfviMaintenanceInfo is defined in clause 7.1.8.17. See note 2.

NOTE 1: Each identifier references an affinity or anti-affinity group which expresses affinity or anti-affinity relationships between the virtualisation container(s) (e.g. virtual machine(s)) to be created using this VDU and the virtualisation container(s) (e.g. virtual machine(s)) to be created using other VDU(s) in the same group.

NOTE 2: An NFVI level operation (e.g. restart of a virtual machine) can impact a VNF and the VNF may be able to tolerate only a limited number of such impacts simultaneously. The nfviMaintenanceInfo provides constraints related to detection and tolerance so that negative impact on VNF functionality can be avoided during NFVI maintenance operations.

7.1.8.4 VirtualLinkProfile information element

7.1.8.4.1 Description

The VirtualLinkProfile information element describes additional instantiation data for a given VL used in a DF.

7.1.8.4.2 Attributes

The attributes of the VirtualLinkProfile information element shall follow the indications provided in table 7.1.8.4.2-1.

Table 7.1.8.4.2-1: Attributes of the VirtualLinkProfile information element

Attribute	Qualifier	Cardinality	Content	Description
vnfVirtualLinkDescId	M	1	Identifier (Reference to VnfVirtualLinkDesc)	Uniquely references a VNF VLD.
flavourId	M	1	Identifier (Reference to VirtualLinkDescFlavour)	References a flavour within the VnfVirtualLinkDesc.
localAffinityOrAntiAffinityRule	M	0..N	LocalAffinityOrAntiAffinityRule	Specifies affinity or anti-affinity rules applicable between the VLs based on this VnfVirtualLinkDesc. See clause 7.1.8.11. When the cardinality is greater than 1, both affinity rule(s) and anti-affinity rule(s) with different scopes are applicable to the VLs based on this VnfVirtualLinkDesc.
affinityOrAntiAffinityGroupId	M	0..N	Identifier (Reference to AffinityOrAntiAffinityGroup)	References the affinity or anti-affinity group(s) the VnfVirtualLinkDesc belongs to. See note 1.
maxBitRateRequirements	M	1	LinkBitrateRequirements	Specifies the maximum bitrate requirements for a VL instantiated according to this profile. See clause 7.1.8.6. See note 2.
minBitRateRequirements	M	1	LinkBitrateRequirements	Specifies the minimum bitrate requirements for a VL instantiated according to this profile. See clause 7.1.8.6. See note 2.
virtualLinkProtocolData	M	0..N	VirtualLinkProtocolData	Specifies the protocol data for a VL instantiated according to this profile. Cardinality 0 is used when no protocol data needs to be specified. See note 3.

NOTE 1: Each identifier references an affinity or anti-affinity group which expresses affinity or anti-affinity relationship between the VL(s) using this VnfVirtualLinkDesc and the VL(s) using other VnfVirtualLinkDesc(s) in the same group.
NOTE 2: These attributes are used to control scaling boundaries.
NOTE 3: If the cardinality is more than 1, the order shall be the same as the order of the layerProtocol occurrences in the connectivityType attribute of the corresponding VnfVirtualLinkDesc, i.e. the first occurrence of the virtualLinkProtocolData represents the highest layer protocol data, and the last occurrence represents the lowest layer protocol data.

7.1.8.5 VirtualLinkDescFlavour information element

7.1.8.5.1 Description

The VirtualLinkDescFlavour information element describes additional instantiation data for a given internal VL used in a DF.

7.1.8.5.2 Attributes

The attributes of the VirtualLinkDescFlavour information element shall follow the indications provided in table 7.1.8.5.2-1.

Table 7.1.8.5.2-1: Attributes of the VirtualLinkDescFlavour information element

Attribute	Qualifier	Cardinality	Content	Description
flavourId	M	1	Identifier	Identifies a flavour within a VnfVirtualLinkDesc.
qos	M	0..1	QoS	QoS of the VL.

7.1.8.6 LinkBitrateRequirements information element

7.1.8.6.1 Description

The LinkBitrateRequirements information element describes the requirements in terms of bitrate for a VL.

7.1.8.6.2 Attributes

The attributes of the LinkBitrateRequirements information element shall follow the indications provided in table 7.1.8.6.2-1.

Table 7.1.8.6.2-1: Attributes of the LinkBitrateRequirements information element

Attribute	Qualifier	Cardinality	Content	Description
root	M	1	Number	Specifies the throughput requirement of the link (e.g. bitrate of E-Line, root bitrate of E-Tree, aggregate capacity of E-LAN).
leaf	M	0..1	Number	Specifies the throughput requirement of leaf connections to the link when applicable to the connectivity type (e.g. for E-Tree and E-LAN branches). See note.
NOTE: The present document does not specify the means to declare different bitrate requirements for leaf connections (e.g. E-LAN leaves).				

7.1.8.7 InstantiationLevel information element

7.1.8.7.1 Description

The InstantiationLevel information element describes a given level of resources to be instantiated within a DF in term of the number of VNFC instances to be created from each VDU, the number of VIP CP instances and/or bit rate requirements.

All the VDUs and/or VipCpds referenced in the level shall be part of the corresponding DF and their number shall be within the range (min/max) for this DF.

7.1.8.7.2 Attributes

The attributes of the InstantiationLevel information element shall follow the indications provided in table 7.1.8.7.2-1.

Table 7.1.8.7.2-1: Attributes of the InstantiationLevel information element

Attribute	Qualifier	Cardinality	Content	Description
levelId	M	1	Identifier	Uniquely identifies a level within the DF.
description	M	1	String	Human readable description of the level.
vduLevel	M	1..N	VduLevel	Indicates the number of instances of this VDU to deploy for this level.
virtualLinkBitRateLevel	M	0..N	VirtualLinkBitRateLevel	Specifies bitrate requirements applicable to virtual links created from particular virtual link descriptors for this level. See note.
vipCpLevel	M	0..N	VipCpLevel	Indicates the number of VIP CP instances based on a particular VipCpd to be part of this level. If a particular VipCpd is defined with minNumberOfInstances= maxNumberOfInstances=1 in the vipCpProfile of the DF, that vipCpd may be omitted from the "vipCpLevel" attribute, which shall be interpreted that one related VIP CP instance is part of this level.
scaleInfo	M	0..N	ScaleInfo	Represents for each aspect the scale level that corresponds to this instantiation level. scaleInfo shall be present if the VNF supports scaling.
NOTE: If not present, it is assumed that the bitrate requirements can be derived from those specified in the VduCpd instances applicable to the internal VL. If present in both the InstantiationLevel and the VduCpd instances applicable to the internal VL, the highest value takes precedence.				

7.1.8.8 ScaleInfo information element

7.1.8.8.1 Description

The ScaleInfo information element represents a scale level for a particular scaling aspect.

7.1.8.8.2 Attributes

The attributes of the ScaleInfo information element shall follow the indications provided in table 7.1.8.8.2-1.

Table 7.1.8.8.2-1: Attributes of the ScaleInfo information element

Attribute	Qualifier	Cardinality	Content	Description
aspectId	M	1	Identifier (Reference to ScalingAspect)	References the scaling aspect.
scaleLevel	M	1	Integer	The scale level, greater than or equal to 0.
NOTE: Vertical scaling (scale up, scale down) is not supported in the present document.				

7.1.8.9 VduLevel information element

7.1.8.9.1 Description

The VduLevel information element indicates for a given VDU in a given level the number of instances to deploy.

7.1.8.9.2 Attributes

The attributes of the VduLevel information element shall follow the indications provided in table 7.1.8.9.2-1.

Table 7.1.8.9.2-1: Attributes of the VduLevel information element

Attribute	Qualifier	Cardinality	Content	Description
vduId	M	1	Identifier (Reference to Vdu)	Uniquely references a VDU.
numberOfInstances	M	1	Integer	Number of instances of VNFC based on this VDU to deploy for an instantiation level or for a scaling delta. Shall be zero or greater.

7.1.8.10 QoS information element

7.1.8.10.1 Description

The QoS information element describes QoS data for a given VL used in a DF.

7.1.8.10.2 Attributes

The attributes of the QoS information element shall follow the indications provided in table 7.1.8.10.2-1.

Table 7.1.8.10.2-1: Attributes of the QoS information element

Attribute	Qualifier	Cardinality	Content	Description
latency	M	1	Number	Specifies the maximum latency in ms.
packetDelayVariation	M	1	Number	Specifies the maximum jitter in ms.
packetLossRatio	M	0..1	Number	Specifies the maximum packet loss ratio.

7.1.8.11 LocalAffinityOrAntiAffinityRule information element

7.1.8.11.1 Description

The LocalAffinityOrAntiAffinityRule information element describes the affinity or anti-affinity rule applicable between the virtualisation containers to be created based on a particular VDU, or between internal VLs to be created based on a particular VnfVirtualLinkDesc.

Per VNF, the affinity/anti-affinity rules defined using this information element, using the AffinityOrAntiAffinityGroup information element, and using the placement constraints in the GrantLifecycleOperation as defined in ETSI GS NFV-IFA 007 [i.3] should be conflict-free. In case of conflicts, the placement constraints in the GrantLifecycleOperation shall take precedence.

Annex B provides additional description and examples about the usage of the affinity/anti-affinity rules.

7.1.8.11.2 Attributes

The attributes of the LocalAffinityOrAntiAffinityRule information element shall follow the indications provided in table 7.1.8.11.2-1.

Table 7.1.8.11.2-1: Attributes of the LocalAffinityOrAntiAffinityRule information element

Attribute	Qualifier	Cardinality	Content	Description
type	M	1	Enum	Specifies whether the rule is an affinity rule or an anti-affinity rule. VALUES: • AFFINITY • ANTI_AFFINITY
scope	M	1	Enum	Specifies the scope of the rule. VALUES: • NFVI-PoP • CIS-node • Zone • ZoneGroup • NFVI-node • network-link-and-node See notes 1 and 3.
nfviMaintenanceGroupInfo	M	0..1	NfviMaintenanceInfo	When present, provides information on the impact tolerance and rules to be observed when a group of instances based on the same VDU is impacted during NFVI operation and maintenance (e.g. NFVI resource upgrades). NfviMaintenanceInfo is defined in clause 7.1.8.17. See note 2.

NOTE 1: When used in an anti-affinity relationship, the "network-link-and-node" scope is conceptually similar to link and node disjoint paths capabilities used commonly in network Traffic Engineering (TE). For example, as in Fast Reroute Resource Reservation Protocol Traffic Engineering (RSVP-TE) for Label-Switched Path (LSP) tunnels as introduced in IETF RFC 4090 [i.9].

NOTE 2: An NFVI level operation (e.g. restart of a virtual machine) can impact a VNF and the VNF can be able to tolerate only a limited number of such impacts simultaneously. The nfviMaintenanceInfo provides constraints related to the tolerated simultaneous impacts on a group of resources so that negative impact on VNF functionality can be avoided during NFVI maintenance operations.

NOTE 3: The "CIS-node" scope is only applicable to express affinity or anti-affinity rules between containerized workloads.

7.1.8.12 AffinityOrAntiAffinityGroup information element

7.1.8.12.1 Description

The AffinityOrAntiAffinityGroup information element describes the affinity or anti-affinity relationship applicable between the virtualisation containers to be created based on different VDUs, or between internal VLs to be created based on different VnfVirtualLinkDesc(s).

Per VNF, the affinity/anti-affinity rules defined using this information element, using the LocalAffinityOrAntiAffinityRule information element, and using the placement constraints in the GrantLifecycleOperation as defined in ETSI GS NFV-IFA 007 [i.3] should be conflict-free. In case of conflicts, the placement constraints in the GrantLifecycleOperation shall take precedence.

Annex B provides additional description and examples about the usage of the affinity/anti-affinity rules.

7.1.8.12.2 Attributes

The attributes of the AffinityOrAntiAffinityGroup information element shall follow the indications provided in table 7.1.8.12.2-1.

Table 7.1.8.12.2-1: Attributes of the AffinityOrAntiAffinityGroup information element

Attribute	Qualifier	Cardinality	Content	Description
groupId	M	1	Identifier	Identifier of this AffinityOrAntiAffinityGroup information element.
affinityOrAntiAffinity	M	1	Enum	Specifies the type of relationship that the members of the group have. VALUES: • AFFINITY • ANTI_AFFINITY
scope	M	1	Enum	Specifies the scope of the affinity or anti affinity relationship. VALUES: • NFVI-PoP • Zone • ZoneGroup • NFVI-node • CIS-node • network-link-and-node • container-namespace See notes 1, 2 and 3.
NOTE 1: When used in an anti-affinity relationship, the "network-link-and-node" scope is conceptually similar to link and node disjoint paths capabilities used commonly in network Traffic Engineering (TE). For example, as in Fast Reroute Resource Reservation Protocol Traffic Engineering (RSVP-TE) for Label-Switched Path (LSP) tunnels as introduced in IETF RFC 4090 [i.9]. NOTE 2: The "container-namespace" scope is used to express the affinity or anti-affinity relationship between containerized workloads which are deployed based on a MCIOP. NOTE 3: The "CIS-node" scope is only applicable to express affinity or anti-affinity relationships between containerized workloads.				

7.1.8.13 VirtualLinkProtocolData information element

7.1.8.13.1 Description

The VirtualLinkProtocolData information element describes the protocol layer and associated protocol data for a virtual link.

7.1.8.13.2 Attributes

The attributes of the VirtualLinkProtocolData information element shall follow the indications provided in table 7.1.8.13.2-1.

Table 7.1.8.13.2-1: Attributes of the VirtualLinkProtocolData information element

Attribute	Qualifier	Cardinality	Content	Description
associatedLayerProtocol	M	1	Enum	One of the values of the attribute layerProtocol of the ConnectivityType IE (refer to clause 7.1.7.3). VALUES: • Ethernet • MPLS • ODU2 • IPV4 • IPV6 • Pseudo-Wire • Etc.
l2ProtocolData	M	0..1	L2ProtocolData	Specifies the L2 protocol data for this virtual link. Shall be present when the associatedLayerProtocol attribute indicates a L2 protocol and shall be absent otherwise.
l3ProtocolData	M	0..1	L3ProtocolData	Specifies the L3 protocol data for this virtual link. Shall be present when the associatedLayerProtocol attribute indicates a L3 protocol and shall be absent otherwise.

7.1.8.14 L2ProtocolData information element

7.1.8.14.1 Description

The L2ProtocolData information element describes the L2 protocol related data for a virtual link.

7.1.8.14.2 Attributes

The attributes of the L2ProtocolData information element shall follow the indications provided in table 7.1.8.14.2-1.

Table 7.1.8.14.2-1: Attributes of the L2ProtocolData information element

Attribute	Qualifier	Cardinality	Content	Description
name	M	0..1	String	Network name associated with this L2 protocol.
networkType	M	0..1	Enum	Specifies the network type for this L2 protocol. VALUES: • FLAT • VLAN • VXLAN • GRE See note.
vlanTransparent	M	0..1	Boolean	Specifies whether to support VLAN transparency for this L2 protocol or not.
mtu	M	0..1	Integer	Specifies the Maximum Transmission Unit (MTU) value for this L2 protocol.
segmentationId	M	0..1	String	If present, specifies a specific virtualised network segment, which depends on the network type. For e.g. VLAN ID for VLAN network type and tunnel ID for GRE/VXLAN network types. See note.
NOTE: If this attribute is included in the VNFD, the attribute value shall be provided at run-time, unless a default value is provided at design time in the VNFD. If a default value is provided at design-time, this value may be overridden at run-time.				

7.1.8.15 L3ProtocolData information element

7.1.8.15.1 Description

The L3ProtocolData information element describes the L3 protocol related data for a virtual link.

7.1.8.15.2 Attributes

The attributes of the L3ProtocolData information element shall follow the indications provided in table 7.1.8.15.2-1.

Table 7.1.8.15.2-1: Attributes of the L3ProtocolData information element

Attribute	Qualifier	Cardinality	Content	Description
name	M	0..1	String	Network name associated with this L3 protocol.
ipVersion	M	1	Enum	Specifies IP version of this L3 protocol. VALUES: • IPV4 • IPV6 See note 1.
cidr	M	1	Not specified	Specifies the CIDR (Classless Inter-Domain Routing) of this L3 protocol. See note 2.
ipAllocationPools	M	0..N	Not specified	Specifies the allocation pools with start and end IP addresses for this L3 protocol. See note 2.
gatewayIp	M	0..1	IpAddress	Specifies the gateway IP address for this L3 protocol. See note 2.
dhcpEnabled	M	0..1	Boolean	Indicates whether DHCP (Dynamic Host Configuration Protocol) is enabled or disabled for this L3 protocol. See note 2.
ipv6AddressMode	M	0..1	Enum	Specifies IPv6 address mode. VALUES: • SLAAC • DHCPV6-STATEFUL • DHCPV6-STATELESS May be present when the value of the ipVersion attribute is "IPV6" and shall be absent otherwise. See note 2.

NOTE 1: The value of the ipVersion attribute shall be consistent with the value of the layerProtocol attribute of the ConnectivityType IE (refer to clause 7.1.7.3).
NOTE 2: If this attribute is included in the VNFD, the attribute value shall be provided at run-time, unless a default value is provided at design time in the VNFD. If a default value is provided at design-time, this value may be overridden at run-time.

7.1.8.16 VnfInterfaceDetails information element

7.1.8.16.1 Description

The VnfInterfaceDetails information element specifies the details of an interface produced by the VNF on the Ve-Vnfm reference point.

7.1.8.16.2 Attributes

The attributes of the VnfInterfaceDetails information element shall follow the indications provided in table 7.1.8.16.2-1.

Table 7.1.8.16.2-1: Attributes of the VnfInterfaceDetails information element

Attribute	Qualifier	Cardinality	Content	Description
interfaceName	M	1	Enum	Specifies an interface produced by the VNF. VALUES: • VNF_CONFIGURATION • VNF_INDICATOR • VNF_LCM_COORDINATION
cpdId	M	1..N	Identifier (Reference to VnfExtCpd)	References one or more CPDs from which to instantiate external CPs through which interface endpoints on the VNF side can be reached by the VNFM. See note.
interfaceDetails	M	0..1	Not Specified	Provide additional data to access the interface endpoint (e.g. API URI prefix).
NOTE: It is assumed that when the parent NS is instantiated, these CPs will be connected to a virtual link to which the VNFM is attached, enabling bi-directional communication between the VNF and the VNFM.				

7.1.8.17 NfviMaintenanceInfo information element

7.1.8.17.1 Description

The NfviMaintenanceInfo information element describes information related to the constraints and rules applicable to virtualised resources and their groups impacted due to NFVI maintenance operations.

7.1.8.17.2 Attributes

The attributes of the NfviMaintenanceInfo information element shall follow the indications provided in table 7.1.8.17.2-1.

Table 7.1.8.17.2-1: Attributes of the NfviMaintenanceInfo information element

Attribute	Qualifier	Cardinality	Content	Description
impactNotificationLeadTime	M	1	Number	The value specifies the minimum notification lead time requested for upcoming impact of the virtualised resource or their group (i.e. between the notification and the action causing the impact).
isImpactMitigationRequested	M	0..1	Boolean	When set to True, it is requested that at the time of the notification of an upcoming change that is expected to have an impact on the VNF, virtualised resource(s) of the same characteristics as the impacted ones is/are provided to compensate for the impact. Cardinality 0 corresponds to the value False.
supportedMigrationType	M	0..N	Enum	Applicable to VirtualComputeDesc and VirtualStorageDesc. When present, specifies the allowed migration types in the order of preference in case of an impact starting with the most preferred type. VALUES: • NO_MIGRATION • OFFLINE_MIGRATION • LIVE_MIGRATION For LIVE_MIGRATION, see note 1.

Attribute	Qualifier	Cardinality	Content	Description
maxUndetectableInterruptionTime	M	0..1	Number	Applicable to VirtualComputeDesc and VirtualStorageDesc. When present, it specifies the maximum interruption time that can go undetected at the VNF level and therefore which will not trigger VNF-internal recovery during live migration (see note 1).
minRecoveryTimeBetweenImpacts	M	0..1	Number	When present, it specifies the time required by the group to recover from an impact, thus, the minimum time requested between consecutive impacts of the group (see note 2).
maxNumberOfImpactedInstances	M	0..N	MaxNumberOfImpactedInstances	When present, specifies for different group sizes the maximum number of instances that can be impacted simultaneously within the group of virtualised resources without losing functionality. Zero cardinality indicates no constraint (see note 2). MaxNumberOfImpactedInstances is defined in clause 7.1.8.18. See note 3.
minNumberOfPreservedInstances	M	0..N	MinNumberOfPreservedInstances	When present, specifies for different group sizes the minimum number of instances which need to be preserved simultaneously within the group of virtualised resources. Zero cardinality indicates no constraint (see note 2). MinNumberOfPreservedInstances is defined in clause 7.1.8.22. See note 3.

NOTE 1: When the maximum undetectable interruption time is specified it constrains the live migration. If it cannot be guaranteed on an NFVI that the interruption caused by the live migration will be less than the indicated maximum undetectable interruption time, then life migration should be downgraded according to the order of preference.
NOTE 2: Impacts to instances of the group happening within the minimum recovery time are considered simultaneous impacts.
NOTE 3: Either "maxNumberOfImpactedInstances" or "minNumberOfPreservedInstances" may be provided, but not both.

7.1.8.18 MaxNumberOfImpactedInstances information element

7.1.8.18.1 Description

The MaxNumberOfImpactedInstances information element specifies the maximum number of instances of a given VDU or VnfVirtualLinkDesc that may be impacted simultaneously without impacting the functionality of the group of a given size.

7.1.8.18.2 Attributes

The attributes of the MaxNumberOfImpactedInstances information element shall follow the indications provided in table 7.1.8.18.2-1.

Table 7.1.8.18.2-1: Attributes of the MaxNumberOfImpactedInstances information element

Attribute	Qualifier	Cardinality	Content	Description
groupSize	M	0..1	Integer	When present, it determines the size of the group for which the maxNumberOfImpactedInstances is specified. Otherwise the size is not limited. See notes 1 and 2.
maxNumberOfImpactedInstances	M	1	Integer	The maximum number of instances that can be impacted simultaneously within the group of the specified size. See notes 1 and 2.
NOTE 1: Each groupSize value specified for a group of virtual resources shall be unique, and it shall be possible to form an ascending ordered list of groupSizes. NOTE 2: The number of instances in the group for which the maxNumberOfImpactedInstances is specified may be equal to groupSize or less. When the number of instances is less than the groupSize, it shall be at least 1 if this is the first groupSize in the ordered list of groupSizes, or it shall be greater by at least 1 than the previous groupSize in the ordered list of groupSizes.				

7.1.8.19 Dependencies information element

7.1.8.19.1 Description

The Dependencies information element provides indications on the order in which VNFCs associated to different VDU Profiles are to be instantiated.

7.1.8.19.2 Attributes

The attributes of the Dependencies information element shall follow the indications provided in table 7.1.8.19.2-1.

Table 7.1.8.19.2-1: Attributes of the Dependencies information element

Attribute	Qualifier	Cardinality	Content	Description
primaryId	M	1..N	Identifier (Reference to VduProfile)	References a VduProfile for describing dependencies between VNFCs in terms of primary entities. See note.
secondaryId	M	1..N	Identifier (Reference to VduProfile)	References a VduProfile for describing dependencies between VNFCs in terms of secondary entities. See note.
NOTE: NFV Management and Orchestration functions shall instantiate VNFCs from the VduProfile in the primaryId attribute before instantiating VNFCs from the VduProfile referenced in the secondaryId attribute.				

7.1.8.20 MciopProfile information element

7.1.8.20.1 Description

A Managed Container Infrastructure Object Package (MCIOP) is a hierarchical aggregate of information objects for OS container management and orchestration. Multiple MCIOPs can be included in a VNF Package. The MciopProfile information element provides properties of the MCIOP which are used during deployment of containerized workloads based on a MCIOP, associated to a VNF deployment flavour.

7.1.8.20.2 Attributes

The attributes of the MciopProfile information element shall follow the indications provided in table 7.1.8.20.2-1.

Table 7.1.8.20.2-1: Attributes of the MciopProfile information element

Attribute	Qualifier	Cardinality	Content	Description
mciopId	M	1	Identifier	Identifies the MCIOP in the VNF package.
deploymentOrder	M	0..1	Integer	Indicates the order in which this MCIOP shall be deployed in relation to other MCIOPs. A lower value specifies an earlier deployment.
affinityOrAntiAffinityGroupId	M	0..N	Identifier (Reference to AffinityOrAntiAffinityGroup)	References the affinity or anti-affinity group(s) the MCIOP belongs to. See note.
associatedVdu	M	0..N	Identifier (Reference to Vdu)	List of VDUs which are associated to this MCIOP and which are deployed using this MCIOP.
NOTE: Each identifier references an affinity or anti-affinity group which expresses affinity or anti-affinity relationships between the containerized workloads to be created using this MCIOP and the containerized workloads to be created using other MCIOP(s) in the same group.				

7.1.8.21 VipCpProfile information element

7.1.8.21.1 Description

The VipCpProfile information element describes additional instantiation data for a given VIP CP used in a DF.

7.1.8.21.2 Attributes

The attributes of the VipCpProfile information element shall follow the indications provided in table 7.1.8.21.2-1.

Table 7.1.8.21.2-1: Attributes of the VipCpProfile information element

Attribute	Qualifier	Cardinality	Content	Description
vipCpdId	M	1	Identifier (Reference to VipCpd)	Uniquely references a VIP CPD.
minNumberOfInstances	M	1	Integer	Minimum number of instances of the VIP CP based on the referenced VIP CPD that is permitted to exist for this flavour. Shall be zero or greater.
maxNumberOfInstances	M	1	Integer	Maximum number of instances of the VIP CP based on the referenced VIP CPD that is permitted to exist for this flavour. Shall be greater than zero and not less than the value of "minNumberOfInstances".

7.1.8.22 MinNumberOfPreservedInstances information element

7.1.8.22.1 Description

The MinNumberOfPreservedInstances information element specifies the minimum number of instances of a given VDU or VnfVirtualLinkDesc which need to be preserved simultaneously within the group of a given size of virtualised resources.

7.1.8.22.2 Attributes

The attributes of the MinNumberOfPreservedInstances information element shall follow the indications provided in table 7.1.8.22.2-1.

Table 7.1.8.22.2-1: Attributes of the MinNumberOfPreservedInstances information element

Attribute	Qualifier	Cardinality	Content	Description
groupSize	M	0..1	Integer	When present, it determines the size of the group for which the minNumberOfPreservedInstances is specified. Otherwise the size is not limited. See notes 1 and 2.
minNumberOPreservedInstances	M	1	Integer	The minimum number of instances which need to be preserved simultaneously within the group of the specified size. See notes 1 and 2.
NOTE 1: Each groupSize value specified for a group of virtual resources shall be unique, and it shall be possible to form an ascending ordered list of groupSizes.				
NOTE 2: The number of instances in the group for which the minNumberOfPreservedInstances is specified may be equal to groupSize or less.				

7.1.9 Information elements related to Virtual Resource descriptors

7.1.9.1 Introduction

The clauses below define the Information elements related to Virtual Resource descriptors.

7.1.9.2 Information elements related to Virtual CPU

7.1.9.2.1 Introduction

The clauses below define the information elements related to Virtual CPU.

7.1.9.2.2 VirtualComputeDesc information element

7.1.9.2.2.1 Description

The VirtualComputeDesc information element supports the specification of requirements related to virtual compute resources.

7.1.9.2.2.2 Attributes

The attributes of the VirtualComputeDesc information element shall follow the indications provided in table 7.1.9.2.2.2-1.

If the VIM supports the concept of virtual compute resource flavours, it is assumed that a flavour is selected or created based on the information in the VirtualComputeDesc information element.

Table 7.1.9.2.2.2-1: Attributes of the VirtualComputeDesc information element

Attribute	Qualifier	Cardinality	Content	Description
virtualComputeDescId	M	1	Identifier	Unique identifier of this VirtualComputeDesc in the VNFD.
logicalNode	M	0..N	LogicalNodeRequirements	The logical node requirements.
requestAdditionalCapabilities	M	0..N	RequestedAdditionalCapabilityData	Specifies requirements for additional capabilities. These may be for a range of purposes. One example is acceleration related capabilities. See clause 7.1.9.5.
computeRequirements	M	0..N	Not specified	Specifies compute requirements.
virtualMemory	M	1	VirtualMemoryData	The virtual memory of the virtualised compute. See clause 7.1.9.3.2.
virtualCpu	M	1	VirtualCpuData	The virtual CPU(s) of the virtualised compute. See clause 7.1.9.2.3.
virtualDisk	M	0..N	BlockStorageData	The local or ephemeral disk(s) of the virtualised compute. See clause 7.1.9.4.3.

7.1.9.2.3 VirtualCpuData information elements

7.1.9.2.3.1 Description

The VirtualCpuData information element supports the specification of requirements related to virtual CPU(s) of a virtual compute resource.

7.1.9.2.3.2 Attributes

The attributes of the VirtualCpuData information element shall follow the indications provided in table 7.1.9.2.3.2-1.

Table 7.1.9.2.3.2-1: Attributes of the VirtualCpuData information element

Attribute	Qualifier	Cardinality	Content	Description
cpuArchitecture	M	0..1	String	CPU architecture type. Examples are x86, ARM. The cardinality can be 0 during the allocation request, if no particular CPU architecture type is requested.
numVirtualCpu	M	1	Integer	Number of virtual CPUs.
virtualCpuClock	M	0..1	Number	Minimum virtual CPU clock rate (e.g. in MHz). The cardinality can be 0 during the allocation request, if no particular value is requested.
virtualCpuOversubscriptionPolicy	M	0..1	Not specified	The CPU core oversubscription policy e.g. the relation of virtual CPU cores to physical CPU cores/threads. The cardinality can be 0 during the allocation request, if no particular value is requested.
vduCpuRequirements	M	0..N	Not specified	Array of key-value pair requirements on the Compute (CPU) for the VDU.
virtualCpuPinning	M	0..1	VirtualCpuPinningData	The virtual CPU pinning configuration for the virtualised compute resource. See clause 7.1.9.2.4.

7.1.9.2.4 VirtualCpuPinningData information element

7.1.9.2.4.1 Description

The VirtualCpuPinningData information element supports the specification of requirements related to the virtual CPU pinning configuration of a virtual compute resource.

7.1.9.2.4.2 Attributes

The attributes of the VirtualCpuPinningData information element shall follow the indications provided in table 7.1.9.2.4.2-1.

Table 7.1.9.2.4.2-1: Attributes of the VirtualCpuPinningData information element

Attribute	Qualifier	Cardinality	Content	Description
virtualCpuPinningPolicy	M	0..1	Enum	Indicates the policy for CPU pinning. VALUES: • STATIC • DYNAMIC In case of "STATIC" the virtual CPU cores are requested to be allocated to logical CPU cores according to the rules defined in virtualCpuPinningRules. In case of "DYNAMIC" the allocation of virtual CPU cores to logical CPU cores is decided by the VIM or CISM (e.g. SMT (Simultaneous Multi-Threading) requirements).
virtualCpuPinningRule	M	0..N	Not specified	List of rules that should be considered during the allocation of the virtual CPUs to logical CPUs in case of "STATIC" virtualCpuPinningPolicy.

7.1.9.3 Information elements related to Virtual Memory

7.1.9.3.1 Introduction

The clauses below define the information elements related to Virtual Memory.

7.1.9.3.2 VirtualMemoryData information element

7.1.9.3.2.1 Description

The VirtualMemoryData information element supports the specification of requirements related to virtual memory of a virtual compute resource.

7.1.9.3.2.2 Attributes

The attributes of the VirtualMemoryData information element shall follow the indications provided in table 7.1.9.3.2.2-1.

Table 7.1.9.3.2.2-1: Attributes of the VirtualMemoryData information element

Attribute	Qualifier	Cardinality	Content	Description
virtualMemSize	M	1	Number	Amount of virtual Memory (e.g. in MB).
virtualMemOversubscription Policy	M	0..1	Not specified	The memory core oversubscription policy in terms of virtual memory to physical memory on the platform. The cardinality can be 0 during the allocation request, if no particular value is requested.
vduMemRequirements	M	0..N	Not specified	Array of key-value pair requirements on the memory for the VDU.
numaEnabled	M	0..1	Boolean	It specifies the memory allocation to be cognisant of the relevant process/core allocation. The cardinality can be 0 during the allocation request, if no particular value is requested.
hugePagesRequirements	M	0..1	Not specified	Specifies requirements on the huge pages resources for the virtual memory.

7.1.9.4 Information elements related to Virtual Storage

7.1.9.4.1 Introduction

The clauses below define the information elements related to Virtual Storage.

7.1.9.4.2 VirtualStorageDesc information element

7.1.9.4.2.1 Description

The VirtualStorageDesc information element supports the specifications of requirements related to persistent virtual storage resources. Ephemeral virtual storage of a virtual machine is specified in VirtualComputeDesc information element. Ephemeral virtual storage of an OS container is specified in OsContainerDesc information element. Annex C provides additional description and examples details about how to describe different cases of ephemeral storage.

NOTE: The present document does not specify the support for consuming specific path volumes from CIS node or NVFI-node.

7.1.9.4.2.2 Attributes

The attributes of the VirtualStorageDesc information element shall follow the indications provided in table 7.1.9.4.2.2-1.

Table 7.1.9.4.2.2-1: Attributes of the VirtualStorageDesc information element

Attribute	Qualifier	Cardinality	Content	Description
id	M	1	Identifier	Unique identifier of this VirtualStorageDesc in the VNFD.
typeOfStorage	M	1	Enum	Type of virtualised storage resource. VALUES: • BLOCK • OBJECT • FILE
blockStorageData	M	0..1	BlockStorageData	Specifies the details of block storage. It shall be present when the "typeOfStorage" attribute is set to "BLOCK". It shall be absent otherwise.
objectStorageData	M	0..1	ObjectStorageData	Specifies the details of object storage. It shall be present when the "typeOfStorage" attribute is set to "OBJECT". It shall be absent otherwise.
fileStorageData	M	0..1	FileStorageData	Specifies the details of file storage. It shall be present when the "typeOfStorage" attribute is set to "FILE". It shall be absent otherwise.
nfviMaintenanceInfo	M	0..1	NfviMaintenanceInfo	When present, provides information on the rules to be observed when an instance based on this VirtualStorageDesc is impacted during NFVI operation and maintenance (e.g. NFVI resource upgrades).
perVnfcInstance	M	0..1	Boolean	Indicates whether the virtual storage resource shall be instantiated per VNFC instance. If the value is true (default), a virtual storage resource shall be instantiated for each VNFC instance that is based on a VDU referring to this virtual storage descriptor and have the same lifetime as the VNFC instance. If the value is false, a single virtual storage resource shall be instantiated with a lifetime independent of the lifetime of individual VNFC instances based on a VDU referring to this virtual storage descriptor. The storage resource shall have the same lifetime as the VNF instance

7.1.9.4.3 BlockStorageData information element

7.1.9.4.3.1 Description

The BlockStorageData information element specifies the details of block storage resource.

7.1.9.4.3.2 Attributes

The attributes of the BlockStorageData information element shall follow the indications provided in table 7.1.9.4.3.2-1.

Table 7.1.9.4.3.2-1: Attributes of the BlockStorageData information element

Attribute	Qualifier	Cardinality	Content	Description
sizeOfStorage	M	1	Number	Size of virtualised storage resource in GB.
vduStorageRequirements	M	0..N	Not Specified	An array of key-value pairs that articulate the storage deployment requirements.
rdmaEnabled	M	0..1	Boolean	Indicate if the storage support RDMA.
swImageDesc	M	0..1	Identifier (Reference to SwImageDesc)	References the software image to be loaded on the VirtualStorage resource created based on this VirtualStorageDesc. Shall be absent when used for virtual disks. See note.
NOTE: This attribute shall not be present in a VirtualStorageDesc used in a VDU realized by one or a set of OS containers.				

7.1.9.4.4 ObjectStorageData information element

7.1.9.4.4.1 Description

The ObjectStorageData information element specifies the details of object storage resource.

7.1.9.4.4.2 Attributes

The attributes of the ObjectStorageData information element shall follow the indications provided in table 7.1.9.4.4.2-1.

Table 7.1.9.4.4.2-1: Attributes of the ObjectStorageData information element

Attribute	Qualifier	Cardinality	Content	Description
maxSizeOfStorage	M	0..1	Number	Max size of virtualised storage resource in GB.

7.1.9.4.5 FileStorageData information element

7.1.9.4.5.1 Description

The FileStorageData information element specifies the details of file storage resource.

7.1.9.4.5.2 Attributes

The attributes of the FileStorageData information element shall follow the indications provided in table 7.1.9.4.5.2-1.

Table 7.1.9.4.5.2-1: Attributes of the FileStorageData information element

Attribute	Qualifier	Cardinality	Content	Description
sizeOfStorage	M	1	Number	Size of virtualised storage resource in GB.
fileSystemProtocol	M	1	String	The shared file system protocol (e.g. NFS, CIFS).
intVirtualLinkDesc	M	1	Identifier (Reference to VnfVirtualLinkDesc)	Reference of the internal VLD which this file storage connects to. The attached VDUs shall connect to the same internal VLD.

7.1.9.5 RequestedAdditionalCapabilityData information element

7.1.9.5.1 Description

This information element describes requested additional capability for a particular VDU. Such a capability may be for acceleration or specific tasks.

7.1.9.5.2 Attributes

The attributes of the RequestedAdditionalCapabilityData information element shall follow the indications provided in table 7.1.9.5.2-1.

Table 7.1.9.5.2-1: Attributes of the RequestedAdditionalCapabilityData information element

Attribute	Qualifier	Cardinality	Content	Description
requestedAdditionalCapabilityName	M	1	String	Specifies a requested additional capability for the VDU. ETSI GS NFV-IFA 002 [i.1] describes acceleration capabilities.
supportMandatory	M	1	Boolean	Indicates whether the requested additional capability is mandatory for successful operation.
minRequestedAdditionalCapabilityVersion	M	0..1	Version	Specifies the minimum version of the requested additional capability.
preferredRequestedAdditionalCapabilityVersion	M	0..1	Version	Specifies the preferred version of the requested additional capability.
targetPerformanceParameters	M	1..N	KeyValuePair	Specifies specific attributes, dependent on the requested additional capability type.

7.1.9.6 LogicalNodeRequirements information element

7.1.9.6.1 Description

This information element describes compute, memory and I/O requirements that are to be associated with the logical node of infrastructure. The logical node requirements are a sub-component of the VDU level requirements. As an example for illustration purposes, a logical node correlates to the concept of a NUMA cell in libvirt terminology.

7.1.9.6.2 Attributes

The attributes of the LogicalNodeRequirements information element shall follow the indications provided in table 7.1.9.6.2-1.

Table 7.1.9.6.2-1: Attributes of the LogicalNodeRequirements information element

Attribute	Qualifier	Cardinality	Content	Description
id	M	1	Identifier	Identifies this set of logical node requirements.
logicalNodeRequirementDetail	M	1..N	Not specified	The logical node-level compute, memory and I/O requirements. An array of key-value pairs that articulate the deployment requirements. This could include the number of CPU cores on this logical node, a memory configuration specific to a logical node (e.g. such as available in the Linux kernel via the libnuma library) or a requirement related to the association of an I/O device with the logical node.

7.1.10 Information elements related to scaling

7.1.10.1 Introduction

The clauses below define the information elements related to scaling. An explanation of the scaling model is provided in annex A.

7.1.10.2 ScalingAspect information element

7.1.10.2.1 Description

The ScalingAspect information element describes the details of an aspect used for horizontal scaling.

7.1.10.2.2 Attributes

The attributes of the ScalingAspect information element shall follow the indications provided in table 7.1.10.2.2-1.

Table 7.1.10.2.2-1: Attributes of the ScalingAspect information element

Attribute	Qualifier	Cardinality	Content	Description
id	M	1	Identifier	Unique identifier of this aspect in the VNFD.
name	M	1	String	Human readable name of the aspect.
description	M	1	String	Human readable description of the aspect.
maxScaleLevel	M	1	PositiveInteger	The maximum scaleLevel for total number of scaling steps that can be applied with respect to this aspect. The value of this attribute corresponds to the number of scaling steps can be applied to this aspect when scaling it from the minimum scale level (i.e. 0) to the maximum scale level defined by this attribute. See note 2.
aspectDeltaDetails	M	0..1	AspectDeltaDetails	A specification of the deltas in terms of number of instances of VNFCs and virtual link bit rates that correspond to the scaling steps of this aspect. A cardinality of zero indicates that this mapping has to be specified in a lifecycle management script or be otherwise known to the VNFM. The information in this attribute, if provided, shall be consistent with the information provided in the "InstantiationLevel" information element. If this attribute is provided, it shall be provided for all scaling aspects. See notes 1 and 3.

NOTE 1: In the present release, support for modifying the internal VNF topology during the scaling of the internal VLs, is not required.
NOTE 2: A scaling step is the smallest increment by which a VNF can be scaled for a particular aspect. Scaling by a single step does not imply that only one VNFC instance is created or removed. It means that one or more VNFC instances are created from the same VDU or from different VDUs, or that a more complex setup occurs.
NOTE 3: The presence of this attribute does not preclude associating lifecycle management scripts to scaling-related events in the VNFD.
NOTE 4: Void.

7.1.10.3 AspectDeltaDetails information element

7.1.10.3.1 Description

The AspectDeltaDetails information element defines the increments in terms of number of instances of VNFCs and virtual link flavours that correspond to the scaling steps of a scaling aspect.

7.1.10.3.2 Attributes

The attributes of the AspectDeltaDetails information element shall follow the indications provided in table 7.1.10.3.2-1.

Table 7.1.10.3.2-1: Attributes of the AspectDeltaDetails information element

Attribute	Qualifier	Cardinality	Content	Description
deltas	M	1..N	ScalingDelta	Declares different scaling deltas, each of which is applied for one or more scaling steps of this aspect.
stepDeltas	M	0..N	Identifier (Reference to ScalingDelta)	References the individual scaling deltas to be applied for the subsequent scaling steps of this aspect. The first entry in the array shall correspond to the first scaling step (between scale levels 0 to 1) and the last entry in the array shall correspond to the last scaling step (between maxScaleLevel-1 and maxScaleLevel). Each referenced scaling delta shall be declared in the "deltas" attribute. See note.
NOTE: A scaling aspect for which only one scaling delta is defined (i.e. for which the "deltas" attribute has only one entry) is called a "uniform aspect". The single delta that is declared for a uniform aspect is called the "uniform delta"; it is applied in all scaling steps of that aspect. For a uniform aspect, the "stepDeltas" attribute may be omitted, as the same scaling delta is applied for all scaling steps.				

7.1.10.4 ScalingDelta information element

7.1.10.4.1 Description

The ScalingDelta information element defines the number of VNFC instances per VDU, the number of VIP CP instances and/or the bitrate delta per virtual link that corresponds to a single scaling step for a particular scaling aspect. When scaling out by one step, this delta is added to the resources of the VNF instance, whereas when scaling in, this delta is removed. The ScalingDelta information element also defines the minimum size of the VNF, as defined by the initialDelta attribute (see table 7.1.8.2.2-1).

7.1.10.4.2 Attributes

The attributes of the ScalingDelta information element shall follow the indications provided in table 7.1.10.4.2-1.

Table 7.1.10.4.2-1: Attributes of the ScalingDelta information element

Attribute	Qualifier	Cardinality	Content	Description
scalingDeltaId	M	1	Identifier	Identifier of this scaling delta.
vduDelta	M	0..N	VduLevel	The number of VNFC instances based on particular VDUs to be created or removed. See notes 1 and 3.
virtualLinkBitRateDelta	M	0..N	VirtualLinkBitRateLevel	The bitrate to be added or removed to virtual links created from particular virtual link descriptors. See note 1.
vipCpDelta	M	0..N	VipCpLevel	Number of VIP CP instances based on a particular VipCpd to be created or removed. See notes 2 and 3.

NOTE 1: At least one of the attributes "vduDelta" and "virtualLinkBitRateDelta" shall be present.

NOTE 2: A particular entry in the attribute "vipCpDelta" may be present if a related "vduDelta" entry is present, and shall be absent otherwise. A "vduDelta" entry is said to be "related" to a "vipCpDelta" entry if the "vipCpDelta" entry references a VduCpd structure that is included in a Vdu structure referenced from the "vduDelta" entry.

NOTE 3: The VNFM shall apply the following default rules for distributing VNFC instances among associated VIP CP instances:

1) If the number of VNFC instances in the scaling delta is a multiple of the number of the related VIP CP instances in the scaling delta, the VNFC instances shall be distributed uniformly among the VIP CP instances.
2) If it is not a multiple, the integer part of the division shall be distributed uniformly among the VIP CP instances. The distribution of the remaining VNFC instances is determined by means outside the scope of the present document.

This default behaviour may be overridden by the VNF provider in LCM scripts.

7.1.10.5 VirtualLinkBitRateLevel information element

7.1.10.5.1 Description

The VirtualLinkBitRateLevel information element specifies bitrate requirements applicable to a virtual link instantiated from a particular VnfVirtualLinkDesc.

7.1.10.5.2 Attributes

The attributes of the VirtualLinkBitRateLevel information element shall follow the indications provided in table 7.1.10.5.2-1.

Table 7.1.10.5.2-1: Attributes of the VirtualLinkBitRateLevel information element

Attribute	Qualifier	Cardinality	Content	Description
vnfVirtualLinkDescId	M	1	Identifier (Reference to VnfVirtualLinkDesc)	Uniquely references a VnfVirtualLinkDesc.
bitrateRequirements	M	1	LinkBitrateRequirements	Bitrate requirements for an instantiation level or bitrate delta for a scaling step.

7.1.10.6 VipCpLevel information element

7.1.10.6.1 Description

The VipCpLevel information element indicates for a given VIP CPD in a given level the number of instances to deploy.

7.1.10.6.2 Attributes

The attributes of the VipCpLevel information element shall follow the indications provided in table 7.1.10.6.2-1.

Table 7.1.10.6.2-1: Attributes of the VipCpLevel information element

Attribute	Qualifier	Cardinality	Content	Description
vipCpdId	M	1	Identifier (Reference to VipCpd)	Uniquely references a VIP CPD.
numberOfInstances	M	1	Integer	Number of VIP CP instances based on the referenced VipCpd to deploy for an instantiation level or for a scaling delta. Shall be zero or greater.

7.1.11 Information elements related to monitoring

7.1.11.1 Introduction

The clauses below define the information elements related to monitoring.

7.1.11.2 VnfIndicator information element

7.1.11.2.1 Description

The VnfIndicator information element defines the indicator the VNF supports.

7.1.11.2.2 Attributes

The attributes of the VnfIndicator information element shall follow the indications provided in table 7.1.11.2.2-1.

Table 7.1.11.2.2-1: Attributes of the VnfIndicator information element

Attribute	Qualifier	Cardinality	Content	Description
id	M	1	Identifier	Unique identifier of the VnfIndicator.
name	M	0..1	String	The human readable name of the VnfIndicator.
indicatorValue	M	1..N	String	Defines the allowed values or value ranges of this indicator.
source	M	1	Enum	Describe the source of the indicator. VALUES: • VNF • EM • Both This tells the consumer where to send the subscription request.

7.1.11.3 MonitoringParameter information element

7.1.11.3.1 Description

This information element specifies the virtualised resource related performance metrics to be tracked by the VNFM, e.g. for auto-scaling purposes. The VNFM collects the values of performance metrics identified by this information element from the VIM(s) using one or more locally initiated PM Jobs. These values can be used as inputs to auto-scaling rules.

7.1.11.3.2 Attributes

The attributes of the MonitoringParameter information element shall follow the indications provided in table 7.1.11.3.2-1.

Table 7.1.11.3.2-1: Attributes of the MonitoringParameter information element

Attribute	Qualifier	Cardinality	Content	Description
monitoringParameterId	M	1	Identifier	Unique identifier of the monitoring parameter.
name	M	0..1	String	Human readable name of the monitoring parameter.
performanceMetric	M	1	String	Specifies the virtualised resource performance metric.
collectionPeriod	M	0..1	Not specified	An attribute that describes the periodicity at which to collect the performance information.

7.1.12 VnfConfigurableProperties information element

7.1.12.1 Description

This information element provides a means to define in the VNFD attributes that represent the configurable properties of a VNF. Configurable properties can be standardized as listed below (e.g. related to auto scaling, auto healing and interface configuration), or can be VNF-specific as defined by the VNF provider. For a VNF instance, the value of these properties can be queried and modified through the VNFM, using the Query VNF and Modify VNF Information operations. Modifying these values affects directly the configuration of an existing VNF instance. If a configurable property is defined in the VNFD, an initial value may be defined as well.

7.1.12.2 Attributes

The attributes of the VnfConfigurableProperties information element shall follow the indications provided in table 7.1.12.2-1.

Table 7.1.12.2-1: Attributes of the VnfConfigurableProperties information element

Attribute	Qualifier	Cardinality	Content	Description
isAutoscaleEnabled	M	0..1	Boolean	Permits to enable (TRUE)/disable (FALSE) the auto-scaling functionality. See note 1.
isAutohealEnabled	M	0..1	Boolean	Permits to enable (TRUE)/disable (FALSE) the auto-healing functionality. See note 1.
vnfmInterfaceInfo	M	0..1	Not specified	Contains information enabling access to the NFV-MANO interfaces produced by the VNFM (e.g. URIs and credentials) See notes 1 and 2.
vnfmOauthServerInfo	M	0..1	Not specified	Contains information to enable discovery of the authorization server protecting access to VNFM interfaces See notes 1 and 2.
vnfOauthServerInfo	M	0..1	Not specified	Contains information to enable discovery of the authorization server to validate the access tokens provided by the VNFM when the VNFM accesses the VNF interfaces, if that functionality (token introspection) is supported by the authorization server. See notes 1 and 2.
additionalConfigurableProperty	M	0..N	Not specified	It provides VNF specific configurable properties that can be modified using the Modify VNF Information operation. See notes 3 and 4.

NOTE 1: A cardinality of "0" indicates that configuring this VNF configurable property is not supported by a particular VNF.
NOTE 2: If this attribute is declared for a VNF, its value shall be set prior to or at instantiation time (as initial value in the VNFD or via the VNF LCM interface). Its value shall be further modifiable after instantiation via the Modify VNF information operation.
NOTE 3: If children of this attribute are declared for a VNF, their values shall be set prior to or at instantiation time (as initial value in the VNFD or via the VNF LCM interface). Their values may be modifiable after instantiation via the Modify VNF information operation if such modification of individual attributes is supported by the VNF and declared per attribute in the VNFD.
NOTE 4: The VNFD shall include information for each of these configurable properties whether its value is writeable (a) prior to/at instantiation time or (b) anytime (i.e. prior to/at instantiation time as well as after instantiation). By default they are writable anytime. The definition of the mechanism to define this is left to the protocol design stage.

7.1.13 LifeCycleManagementScript information element

7.1.13.1 Description

Clause 7.1.13.2 defines the information elements related to the lifecycle management script for the VNF.

7.1.13.2 Attributes

The content of the LifeCycleManagementScript type shall comply with the indications provided in table 7.1.13.2-1.

Table 7.1.13.2-1: Attributes of the LifeCycleManagementScript information element

Attribute	Qualifier	Cardinality	Content	Description
lcmScriptId	M	0..1	Identifier	Identifier of this script for later referencing. Shall be present if there is the need to reference this script from another information element. May be absent otherwise.

Attribute	Qualifier	Cardinality	Content	Description
event	M	0..N	Enum	Describes VNF lifecycle event(s) or an external stimulus detected on a VNFM reference point. The set of lifecycle events triggered internally by the VNFM includes below values. VALUES: • EVENT_START_INSTANTIATION • EVENT_END_INSTANTIATION • EVENT_START_SCALING • EVENT_END_SCALING • EVENT_START_SCALING_TO_LEVEL • EVENT_END_SCALING_TO_LEVEL • EVENT_START_HEALING • EVENT_END_HEALING • EVENT_START_TERMINATION • EVENT_END_TERMINATION • EVENT_START_VNF_FLAVOR_CHANGE • EVENT_END_VNF_FLAVOR_CHANGE • EVENT_START_VNF_OPERATION_CHANGE • EVENT_END_VNF_OPERATION_CHANGE • EVENT_START_VNF_EXT_CONN_CHANGE • EVENT_END_VNF_EXT_CONN_CHANGE • EVENT_START_VNFINFO_MODIFICATION • EVENT_END_VNFINFO_MODIFICATION EVENT_START_VNF_SNAPSHOT_CREATION • EVENT_END_VNF_SNAPSHOT_CREATION • EVENT_START_VNF_SNAPSHOT_REVERTINGTO • EVENT_END_VNF_SNAPSHOT_REVERTINGTO • EVENT_START_CHANGE_CURRENT_VNF_PACKAGE • EVENT_END_CHANGE_CURRENT_VNF_PACKAGE The set of external stimuli includes below values. VALUES: • receipt of request message of instantiation, scaling, healing, termination • change of VNF flavour • change of the operation state of the VNF • change of external VNF connectivity • creation of and reverting to VNF snapshot • change of current VNF Package • modification of VNF information • receipt of a notification regarding the change of a VNF indicator value See note 1.
lcmTransitionEvent	M	0..N	String	Describes the transition VNF lifecycle event(s) that cannot be mapped to any of the enumerated values defined for the event attribute. See note 1.
script	M	1	Not specified	Includes a VNF LCM script (e.g. written in a DSL as specified in requirement VNF_PACK.LCM.001) triggered to react to one of the events listed in the event attribute.
scriptDsl	M	1	String	Defines the domain specific language (i.e. the type) of script that is provided. Types of scripts could include bash, python, etc.
scriptInput	M	0..N	Not specified	Array of KVP requirements with the key as the parameter name and the value as the parameter that need to be passed as an input to the script. See note 3.

NOTE 1: At least one of these two attributes shall be included.
NOTE 2: Void.
NOTE 3: The scriptInput values are passed to the scripts in addition to the parameters received in the operation invocation request or indicator value change.

7.1.14 VnfInfoModifiableAttributes information element

7.1.14.1 Description

This information element defines the VNF-specific extension and metadata attributes of the VnfInfo that are writeable via the ModifyVnfInfo operation.

7.1.14.2 Attributes

The attributes of the VnfInfoModifiableAttributes information element shall follow the indications provided in table 7.1.14.2-1.

Table 7.1.14.2-1: Attributes of the VnfInfoModifiableAttributes information element

Attribute	Qualifier	Cardinality	Content	Description
extension	M	0..N	Not specified	All additional VNF-specific attributes of VnfInfo that affect the lifecycle management of a VNF instance. For each VNF instance, these attributes are stored persistently by the VNFM and can be queried and modified through the VNFM. These attributes are intended to be consumed by the VNFM or by the lifecycle management scripts during the execution of VNF lifecycle management operations. Modifying these values has no direct effect on the VNF instance; however, modified values can be considered during subsequent VNF lifecycle management operations, which means that the modified values can indirectly affect the configuration of the VNF instance. See note 1.
metadata	M	0..N	Not specified	Additional VNF-specific attributes of VnfInfo that provide metadata describing the VNF instance and that are defined by the VNF provider. See note 2. For each VNF instance, these attributes are stored persistently by the VNFM and can be queried and modified through the VNFM. These attributes are intended to provide information to functional blocks external to the VNFM and will not be used by the VNFM or the VNF lifecycle management scripts when executing lifecycle management operations. Modifying these attributes has no effect on the VNF instance. It only affects the attribute values stored by the VNFM. See note 1.
NOTE 1: The exact data structure describing the attribute is left for data model solution specification, but it should include: name, and any constraints on the values, such as ranges, predefined values, etc. NOTE 2: Metadata attributes, including those that are not declared in the VNFD, are allowed to be provided a runtime.				

7.1.15 Information elements related to change current VNF Package

7.1.15.1 Introduction

The clauses below define the information elements related to the change of the current VNF Package.

This operation encompasses only the following cases:

- Changes of the VNF virtualised resources, such as requirements, composition and structure between the VNF versions, without changing the VNF software version.
- Changes of both the VNF software version and the VNF virtualised resources. This case includes replacing the VNF software version by means of virtualised resource management, such as terminating the virtualised resource instances running the current software version and instantiating new virtualised resource instances with the target VNF software version. The new virtualised resource instances may have the same characteristics as the current virtualised resource instances.
- Changes related to the VNFD, such as correction of bugs in the VNFD, changes in the naming scheme of VNFD components (e.g. name of the VDU, vduId), and adding/removing VnfPackageChangeInfo.

7.1.15.2 VnfPackageChangeInfo information element

7.1.15.2.1 Description

A VnfPackageChangeInfo information element describes the processes and rules to be used for performing the resource related tasks while assisting the "change current VNF Package" to change a VNF instance to a different VNF Package (destination package).

When creating a VNF package, the VNF vendor can include VnfPackageChangeInfo information elements in the package which allow the package to act as a source package or as a destination package for a modification in relation to another package, which has been created earlier or at the same time. To populate a VnfPackageChangeInfo information element and the underlying related information elements, knowledge of both the source package and the destination package is required. The following examples illustrate two main use cases.

EXAMPLE 1: Assuming a VNF package V17 created at time t1 and an evolved VNF package V18 created later at time t2, all modification information related to changing a VNF instance from package V17 to package V18, and also all modification information related to changing from package V18 to package V17 are included in package V18, since at the time of creating package V17, the specifics of package V18 were not known. In other words, in this scenario, all the VnfPackageChangeInfo information elements are defined in package V18 which plays the role of the destination package in the transition from V17 to V18, and which plays the role of the source package for the transition from V18 to V17. Typical use cases that can be covered by this example are update/upgrade (V17 -> V18) and downgrade (V18 -> V17).

EXAMPLE 2: In addition to the packages mentioned in example 1, assume another package V17.1 created also at time t2, which is an evolution of V17 and which includes a subset of the changes implemented in V18. Since V17.1 is created at the same time as V18 and all necessary information related to the transitions between V17.1 and V18 is available, it is up to the VNF vendor to choose in which of the two packages to define the VnfPackageChangeInfo information elements related to these transitions. Depending on this decision, when executing a modification, these information elements can either be found in the source package or in the destination package of that transition. For example, for the transition V17.1 -> V18, the VnfPackageChangeInfo information elements can be declared in V17.1 (source package) or in V18 (destination package). A typical use case illustrated by this example is the separation of a bugfix package (V17.1) from a feature enhancements package (V18).

In case both source and destination package contain a VnfPackageChangeInfo information element with identical VersionSelector values, these two information elements shall define the same modification, and the entity processing the packages may choose either of them to process.

7.1.15.2.2 Attributes

The attributes of the VnfPackageChangeInfo information element shall follow the indications provided in table 7.1.15.2.2-1.

Table 7.1.15.2.2-1: Attributes of the VnfPackageChangeInfo information element

Attribute	Qualifier	Cardinality	Content	Description
selector	M	1..N	VersionSelector	Information to identify the combination(s) of source and destination VNFD for, and the related deployment flavour for which the package change defined in this information element applies. See note.
additionalParamsId	M	0..1	Identifier (Reference to ChangeCurrentVnfPackageOpConfig)	References the ChangeCurrentVnfPackageOpConfig information element that defines the valid additional parameters for the change.
modificationQualifier	M	1	Enum	Specifies the type of modification resulting from transitioning from srcVnfdId to dstVnfdId. VALUES: • UP: indicating that the destination VNF version is newer than the source version • DOWN: indicating that the destination VNF version is older than the source version
additionalModificationDescription	M	0..N	String	A VNF provider may define additional information to qualify further the change between the two versions, such as "VNF upgrade", "VNF update", "VNF downgrade", etc.
componentMapping	M	0..N	ComponentMapping	Mapping information related to identifiers of components in source VNFD and destination VNFD that concern to the change process.
lcmScriptId	M	0..1	Identifier (Reference to LifeCycleManagementScript)	References a lifecycle management script that is executed as part of this "change current VNF Package" process.
coordinationActionName	M	0..N	Identifier (Reference to VnfLcmOperationCoordination)	References applicable VNF LCM operation coordination actions that can be invoked during a VNF package change as defined by the "selector" attribute.
dstFlavourId	M	1	Identifier	Identifies the deployment flavour in the destination VNF package for which this change applies. The flavour ID is defined in the destination VNF package.
NOTE: If multiple selectors are indicated, all attributes apart from "selector" in the VnfPackageChangeInfo information element define the package change that is applicable to any change path defined by any of the selectors. If change paths require e.g. different componentMappings, they shall be described by different VnfPackageChangeInfo information elements. Each triplet (srcVnfdId, dstVnfdId, srcFlavourId) represented by one selector shall not appear more than once in a VNFD. If a triplet occurs in both, source and destination package, the content of the identified VnfPackageChangeInfo information elements (apart from the selectors) shall be the same.				

7.1.15.3 VersionSelector information element

7.1.15.3.1 Description

The VersionSelector information element allows to identify the source and destination VNFDs (and implicitly, VNF packages) for a "change current VNF Package", as well as the applicable source deployment flavour. The triplet (srcVnfdId, srcFlavourId, dstVnfdId) uniquely determines a change.

7.1.15.3.2 Attributes

The attributes of the VersionSelector information element shall follow the indications provided in table 7.1.15.3.2-1.

Table 7.1.15.3.2-1: Attributes of the VersionSelector information element

Attribute	Qualifier	Cardinality	Content	Description
srcVnfdId	M	1	Identifier	Identifier of the source VNFD and the source VNF package. See note 1.
dstVnfdId	M	1	Identifier	Identifier of the destination VNFD and the destination VNF package. See note 1.
srcFlavourId	M	1	Identifier	Identifier of the deployment flavour in the source VNF package for which this modification applies. See note 2.
NOTE 1: Either the srcVnfdId or the dstVnfdId shall be equal to the vnfdId of the VNFD containing this version selector. NOTE 2: It is up to protocol design stage to decide whether there is further optimization potential to apply one modification for multiple srcFlavourIds.				

7.1.15.4 ComponentMapping information element

7.1.15.4.1 Description

With respect to a "change current VNF Package" process, a ComponentMapping information element defines a mapping between the identifier of a components or property in the source VNFD and the identifier of the corresponding component or property in the destination VNFD. Examples for components are VDUs, VLDs, etc. and an example for a property is a scaling aspect of the VNF.

7.1.15.4.2 Attributes

The attributes of the ComponentMapping information element shall follow the indications provided in table 7.1.15.4.2-1.

Table 7.1.15.4.2-1: Attributes of the ComponentMapping information element

Attribute	Qualifier	Cardinality	Content	Description
componentType	M	1	Not specified	The type of component or property. Possible values differentiate whether changes concern to some VNF component (e.g. VDU, internal VLD, etc.) or property (e.g. a Scaling Aspect, etc.).
sourceDescId	M	1	Identifier	Identifier of the component or property in the source VNFD. See note.
dstDescId	M	1	Identifier	Identifier of the component or property in the destination VNFD. See note.
description	M	0..1	String	Human readable description of the component changes.
NOTE: The attribute's content, an identifier value, references to the relevant descriptor parts in the VNFD.				

7.1.16 Information elements related to the coordination in VNF lifecycle management operations

7.1.16.1 Introduction

This clause defines information elements which represent information used for the coordination in lifecycle management operations as specified in ETSI GS NFV-IFA 008 [i.4].

Coordination actions are invoked by the VNFM towards the VNF instance or towards operation supporting management systems (e.g. EM). They can be standardized or VNF-specific. It is defined during the data model design stage how to distinguish between both categories by defining namespaces for the values of the "coordinationActionName" attribute.

Coordination actions shall be declared with their parameters in the VnfLcmOperationCoordination information element (see clause 7.1.16.2), unless they are defined in an ETSI NFV specification, in which case they may be declared. For coordination actions that are defined in a standard and declared in the VNFD, the declaration in the VNFD and the declaration in the standard shall not conflict.

References to the VNF-specific and/or standardized coordination actions that are supported by a VNF and/or expected to be supported by operation supporting management systems (e.g. EM) shall be defined in the LcmCoordinationActionMappings information element (see clause 7.1.16.3) or the VnfPackageChangeInfo information element (see clause 7.1.15.2).

7.1.16.2 VnfLcmOperationCoordination information element

7.1.16.2.1 Description

This information element defines the sets of information needed for the VNF-specific coordination actions.

7.1.16.2.2 Attributes

The VnfLcmOperationCoordination information element shall follow the indications provided in table 7.1.16.2.2-1.

Table 7.1.16.2.2-1: Attributes of the VnfLcmOperationCoordination information element

Attribute	Qualifier	Cardinality	Content	Description
coordinationActionName	M	1	Identifier	Identifies the specific VNF LCM operation coordination action. Shall be unique within the scope of the VNFD.
description	M	0..1	String	Human readable description of the coordination action.
endpointType	M	0..1	Enum	Specifies the type of the endpoint exposing the LCM operation coordination such as operations supporting management systems (e.g. EM) or the VNF instance. VALUES: • MGMT: coordination with operation supporting management systems • VNF: coordination with the VNF instance If this attribute is omitted, the endpoint that provides the interface will be determined at deployment time. If the VNF produces the LCM coordination interface, this attribute may be omitted or may have the value "VNF", and a VnfInterfaceDetails entry with the "interfaceName" attribute set to "VNF_LCM_COORDINATION" shall be specified in the related deployment flavour to signal where this interface is exposed by the VNF. If the VNF does not produce the LCM coordination interface but coordination via this interface is needed, it is expected that a management entity such as the EM exposes the coordination interface, and consequently, this attribute shall have the value "MGMT".

Attribute	Qualifier	Cardinality	Content	Description
coordinationStage	M	0..1	Enum	Indicates whether the coordination action is invoked before or after all other changes performed by the VNF LCM operation. See note 1. VALUES: • START: the coordination action is invoked after receiving the grant and before the LCM operation performs any other changes. • END: the coordination action is invoked after the LCM operation has performed all other changes. This attribute shall be omitted if the coordination action is intended to be invoked at an intermediate stage of the LCM operation, i.e. neither at the start nor at the end. In this case, the actual instant during the LCM operation when invoking the coordination is determined by means outside the scope of the present document such as VNFM-internal logic or LCM script.
inputParameter	M	0..1	Not specified	Input parameters needed by the external coordinating entity. See note 2.
outputParameter	M	0..1	Not specified	Output parameters provided by the external coordinating entity. See note 2.
NOTE 1: The changes mentioned include changes to the VNF instance, its resources or its snapshots. NOTE 2: These attributes relate to the corresponding parameters used in the VNF LCM coordination operations (refer to clauses 6.4.2.2 of ETSI GS NFV-IFA 008 [i.4]).				

7.1.16.3 LcmCoordinationActionMapping information element

7.1.16.3.1 Description

This information element defines the LCM coordination actions supported by a VNF and/or expected to be supported by operation supporting management systems (e.g. EM) for a particular VNF LCM operation.

7.1.16.3.2 Attributes

The LcmCoordinationActionMapping information element shall follow the indications provided in table 7.1.16.3.2-1.

Table 7.1.16.3.2-1: Attributes of the LcmCoordinationActionMappings information element

Attribute	Qualifier	Cardinality	Content	Description
vnfLcmOperation	M	1	Enum	Identifies the specific VNF LCM operation. VALUES: • INSTANTIATE • SCALE • SCALE_TO_LEVEL • CHANGE_FLAVOUR • TERMINATE • HEAL • OPERATE • CHANGE_EXT_CONN • MODIFY_INFO • CREATE_SNAPSHOT • REVERT_TO_SNAPSHOT See note.
coordinationActionName	M	1..N	Identifier (Reference to VnfLcmOperation Coordination)	References to the names of coordination actions that can be invoked during the LCM operation indicated by the "vnfLcmOperation" attribute. The related coordination actions shall either be declared in the VnfLcmOperationCoordination information element in the same VNFD, or shall be well-known standardized coordination action name identifiers.
NOTE: The value "CHANGE_VNFPKG" is part of this value set as the coordination actions for the "ChangeCurrrentVnfPkg" are modelled separately in the "VnfPackageChangeInfo" information element.				

7.1.17 Information elements related to VipCpd

7.1.17.1 Introduction

The clauses below define the information elements related to the VipCpd.

7.1.17.2 VipCpd information element

7.1.17.2.1 Description

A VipCpd is a type of Cpd and describes a requirement to allocate one or a set of virtual IP addresses.

A VipCpd inherits from the Cpd Class (see clause 7.1.6.3). All attributes of the Cpd are also attributes of the VipCpd.

Instances of VduCps created from a VduCpd that is indicated via the "intCpd" attribute in the VipCpd are able to communicate via the addresses associated to the VipCp instance created from the VipCpd.

7.1.17.2.2 Attributes

The attributes of the VipCpd information element shall follow the indications provided in table 7.1.17.2.2-1.

Table 7.1.17.2.2-1: Attributes of the VipCpd information element

Attribute	Qualifier	Cardinality	Content	Description
intCpd	M	1..N	Identifier (Reference to VduCpd)	References the internal VDU CPD which is used to instantiate internal CPs. These internal CPs share the virtual IP addresses allocated when a VipCp instance is created from the VipCpd. See note 3.
intVirtualLinkDesc	M	0..1	Identifier (Reference to VnfVirtualLinkDesc)	Reference of the internal VLD which this VipCpd connects to. See note 4.
dedicatedIpAddress	M	0..1	Boolean	If set to true, it indicates that the VIP address shall be different from the addresses allocated to all of the VduCp instances associated to it. If set to false, the VIP address shall be the same as one of the VduCp instances associated to it.
vipFunction	M	1	Enum	It indicates the function the virtual IP address is used for. VALUES: • high availability • load balancing See note 1.
(inherited attributes)				All attributes inherited from Cpd. See note 2.

NOTE 1: When used for high availability, only one of the internal VDU CP instances or VNF external CP instances that share the virtual IP is bound to the VIP address at a time, i.e. only one is configured in the external (to the VNF) router to receive the packets e.g. as a result of a G-ARP message previously sent by this instance. When used for load balancing purposes all CP instances that share the virtual IP are bound to it. A load balancing function sends the packet to one or the other, but not to both.

NOTE 2: For CPs exposed by VNFs realized only by one or set of OS containers and used by the OS containers to connect to the primary container cluster external network, the ability to configure virtualised resources based on cpRole and trunkMode attributes might not be supported by all container technologies.

NOTE 3: If more than one VduCpd is indicated, the intVirtualLinkDesc attribute in all VduCpds referred by the intCpd attribute shall either be present and have the same value in all VduCpds or absent in all.

NOTE 4: This attribute shall be present if it is present in all VduCpds referred by the intCpd attribute and have the same value as in all VduCpds and shall be absent if it is absent in all VduCpds referred by the intCpd attribute.

7.1.18 Information elements related to VirtualCpd

7.1.18.1 Introduction

The clauses below define the information elements related to the VirtualCpd.

7.1.18.2 VirtualCpd information element

7.1.18.2.1 Description

A VirtualCpd is a type of Cpd and describes a requirement to create a virtual connection point allowing the access to a number of VNFC instances (based on their respective VDUs).

Figure 7.1.18.2.1-1 illustrates an example of the virtual CP concept applied to a VNF:

- There are two instances of VirtualCp, one for each service. Each VirtualCp is also exposed as VnfExtCp, enabling exposing the service outside the VNF and allowing external clients to access such service.
- Each VirtualCp instance maps to a set of VNFC instances which are the ones realizing the respective service.
- Forwarding of packets is performed between the external VL and the internal VNF VL, in this example depicted as "container cluster internal network" for the case of VNF being realized as a set of OS containers.

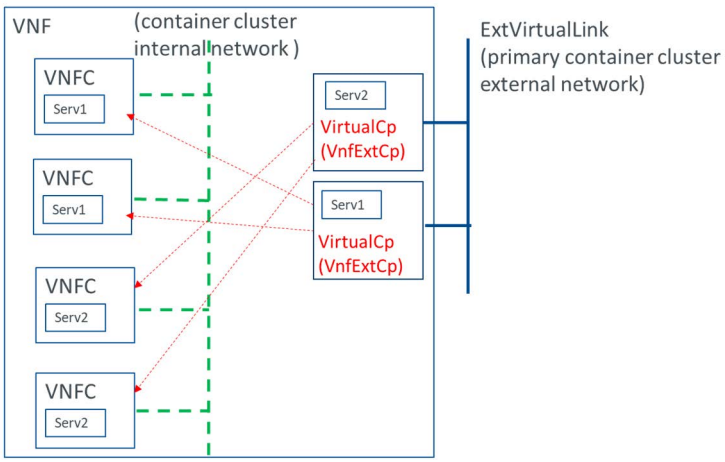

Figure 7.1.18.2.1-1: Example and use case of VirtualCp in an OS container framework

A VirtualCpd inherits from the Cpd Class (see clause 7.1.6.3). All attributes of the Cpd are also attributes of the VirtualCpd.

7.1.18.2.2 Attributes

The attributes of the VirtualCpd information element shall follow the indications provided in table 7.1.18.2.2-1.

Table 7.1.18.2.2-1: Attributes of the VirtualCpd information element

Attribute	Qualifier	Cardinality	Content	Description
vdu	M	1..N	Identifier (Reference to Vdu)	References the VDU(s) which implement this service.
additionalServiceData	M	0..N	AdditionalServiceData	Additional service identification data of the VirtualCp exposed to NFV-MANO.
(inherited attributes)				All attributes inherited from Cpd. The address type "MAC address" in AddressData is not applicable. See notes 1 and 2.
NOTE 1: If this VirtualCp represents a load balancing virtual IP address of a VNFC realized by one or a set of OS containers and the IP address is configurable in the declarative descriptor of the corresponding MCIO, the attribute ipAddressAssignment shall be set to value=true in the L3AddressData. NOTE 2: For CPs exposed by VNFs realized only by one or a set of OS containers and used by the OS containers to connect to the primary container cluster external network, the ability to configure virtualised resources based on cpRole and trunkMode attributes might not be supported by all container technologies.				

7.1.18.3 AdditionalServiceData information element

7.1.18.3.1 Description

This information element describes the additional service data of the VirtualCp used to expose properties of the VirtualCp to NFV-MANO.

If the VirtualCp is exposed by a VNF component realized by one or a set of OS containers, the properties are mirrored from the declarative descriptor of the corresponding MCIO where available.

EXAMPLE: The attribute "serviceData" can be used to expose a list of ingress rules to the services exposed by the VirtualCp. These ingress rules could for example be specified as regular expressions or json-formatted matching rules.

7.1.18.3.2 Attributes

The attributes of the AdditionalServiceData information element shall follow the indications provided in table 7.1.18.3.2-1.

Table 7.1.18.3.2-1: Attributes of the AdditionalServiceData information element

Attribute	Qualifier	Cardinality	Content	Description
portData	M	1..N	ServicePortData	Service port numbers exposed by the VirtualCp.
serviceData	M	0..1	Not specified	Service matching information exposed by the VirtualCp. See note.
NOTE: This attribute shall only be present if additional information is needed to identify the service termination within the VNF, such as for example a url path information in an HTTP request required to allow a single VirtualCp IP address to be used for several HTTP based services that use the same portnumber.				

7.1.18.4 ServicePortData information element

7.1.18.4.1 Description

This information element describes the service identifying port properties exposed by the VirtualCp.

7.1.18.4.2 Attributes

The attributes of the ServicePortData information element shall follow the indications provided in table 7.1.18.4.2-1.

Table 7.1.18.4.2-1: Attributes of the ServicePortData information element

Attribute	Qualifier	Cardinality	Content	Description
name	M	1	String	The name of the port exposed by the VirtualCp.
protocol	M	1	Enum	The L4 protocol for this port exposed by the VirtualCp. VALUES: • TCP • UDP • SCTP
port	M	1	Integer	The L4 port number exposed by the VirtualCp.
portConfigurable	M	1	Boolean	Specifies whether the port attribute value is allowed to be configurable.

8 Functional requirements for VNF Snapshot Packaging

8.1 Generic Functional Requirements

Table 8.1-1 specifies generic functional requirements applicable to VNF Snapshot Packaging.

Table 8.1-1: Generic functional requirements for VNF Snapshot Packaging

Req Number	Requirement Description	Comments
SNAP_PACK.GEN.001	The VNF Snapshot Package contents, including the Snapshot descriptor, snapshot images and artifacts, as well as a human-readable name, checksum, etc. as appropriate constitutes a single delivery unit from a distribution perspective.	

8.2 Functional requirements for VNF Snapshot Packaging specification

8.2.1 Requirements for the structure of a VNF Snapshot Package

Table 8.2.1-1 specifies requirements applicable to the structure of a VNF Snapshot Package.

Table 8.2.1-1: Requirements for the structure of a VNF Snapshot Package

Req Number	Requirement Description	Comments
SNAP_PACK.STRUCT.001	The VNF Snapshot Package shall be assembled in one file.	
SNAP_PACK.STRUCT.002	The Snapshot Package shall be digitally signed by the creator of the package.	
SNAP_PACK.STRUCT.003	The VNF Snapshot Package shall contain files for the VNF Snapshot, its corresponding metadata, and one to many VNFC Snapshot image(s) or reference(s) and its/their corresponding metadata.	
SNAP_PACK.STRUCT.004	The VNF Snapshot Package shall enable including additional Snapshot Artifacts related to the VNF/VNFC Snapshot that are not VNFC Snapshot images or referencing these files if they are external to the package.	
SNAP_PACK.STRUCT.005	The VNF Snapshot Package shall provide means to address individually the files which it contains and/or which it references.	
SNAP_PACK.STRUCT.006	If an external reference (e.g. URL) is used, file integrity information (such as checksum/signature) shall be specified to guarantee the integrity of the referenced file, so it cannot be substituted with a different file by the same name.	

8.2.2 Requirements for the description of VNF Snapshot Package content

Table 8.2.2-1 specifies requirements applicable to the content of a VNF Snapshot Package.

Table 8.2.2-1: Requirements for the description of VNF Snapshot Package content

Req Number	Requirement Description	Comments
SNAP_PACK.DESC.001	The VNF Snapshot Package shall contain one or moreVnfcSnapshotImageInfo information elements.	
SNAP_PACK.DESC.002	The VNF Snapshot Package shall provide a mechanism to describe the package and its contents including, not limited to, human-readable name of the package, state of the package, creation date, indication whether it is a partial or full VNF Snapshot Package, and identification of the included metadata/artifacts.	
SNAP_PACK.DESC.003	The VNF Snapshot Package may contain the VNFD of the snapshotted VNF instance. See note.	
SNAP_PACK.DESC.004	VNFD metadata shall be placed in a well-known location within the VNF Snapshot Package in order for the compliant parsers to find and extract.	
SNAP_PACK.DESC.005	The VNF Snapshot Package shall contain the VnfInfo information element of the snapshotted VNF instance.	
SNAP_PACK.DESC.006	The VNF Snapshot Package shall contain the identifiers of the VNF Snapshot Info information element.	
SNAP_PACK.DESC.007	The VNF Snapshot Package shall contain the identifiers of the VNFC Snapshot Info information element(s).	
SNAP_PACK.DESC.008	The VNF Snapshot Package shall provide a means to include additional userDefinedData.	
SNAP_PACK.DESC.009	The VNF Snapshot Package shall provide means to store sets of related artifacts in the package.	
NOTE: If a VNFD is present in the VNF Snapshot Package, it shall be an exact copy of the VNFD in the VNF Package from which the snapshotted VNF was instantiated. That copy can be used for troubleshooting by entities external to NFV-MANO. The VNFD in the VNF Snapshot Package is not intended to be used by NFV-MANO entities, e.g. for VNF snapshot reversal.		

8.2.3 Requirements for security and integrity of a VNF Snapshot Package

Table 8.2.3-1 specifies the requirements applicable to the security and integrity of a VNF Snapshot Package.

Table 8.2.3-1: Requirements for security and integrity of a VNF Snapshot Package

Numbering	Requirement Description	Comments
SNAP_PACK.SEC.001	The digest and the public key of the entity signing VNF Snapshot Package shall be included in the package along with the corresponding certificate.	
SNAP_PACK.SEC.002	For each signed artifact, corresponding public key, algorithm and certificate used shall be stored in a well-known location within the VNF Snapshot Package.	
SNAP_PACK.SEC.003	Each artifact in the VNF Snapshot Package shall be signed by the VNF Snapshot Package provider.	

Annex A (informative): Explanation of the scaling model

A.1 Overview

A VNF instance can be scaled in the following directions:

- scale out: adding additional VNFC instances to the VNF to increase capacity;
- scale in: removing VNFC instances from the VNF to release unused capacity.

Scaling can be performed in two different ways:

- "ScaleVnf" operation: scaling is performed in steps separately per "scaling aspect", allowing different aspects of a VNF to be scaled independently by adding/removing deltas to/from that aspect (see clause A.2).
- "ScaleToLevel" operation: scaling to a particular target size is performed in one step. The target size can be expressed by one of the instantiation levels pre-defined in the VNFD, or by a tuple of scale levels, one per aspect (see clause A.3).

It depends on the VNF design and is defined in the VNFD which scaling operations are supported. The operations are defined in ETSI GS NFV-IFA 007 [i.3] and ETSI GS NFV-IFA 008 [i.4].

A.2 Scaling the individual scaling aspects of a VNF

Different *aspects* of a VNF can be scaled independently by the "ScaleVnf" operation.

For example, a VNF could be designed to independently scale database capacity provided by database VNFCs and call processing capacity provided by call processing VNFCs, making "database" and "call processing" two different scaling aspects.

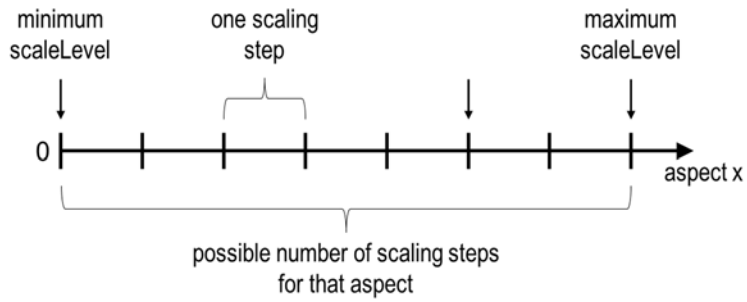

Figure A.2-1: Illustrating the concepts of scale level and scaling steps for a particular scaling aspect

Each scaling aspect can be scaled in discrete steps, the so-called "*scaling steps*", as illustrated in figure A.2-1. Each scaling step corresponds to adding or removing a *scaling delta* (set of VNFCs based on one or more VDUs, and the related virtualised storage/virtualised network resources) to or from the VNF instance, and (re)configuring the virtualised resources.

A scaling step is the smallest unit by which a particular aspect of a VNF can be scaled. For each scaling aspect, the minimum scale level is assumed as zero, and the maximum scale level is defined in the VNFD. The maximum scale level corresponds to the maximum number of scaling steps that can be performed for this aspect, starting from the minimum scale level (i.e. zero). The maximum scale level represents the maximum configuration of that aspect of the VNF in a given deployment flavour.

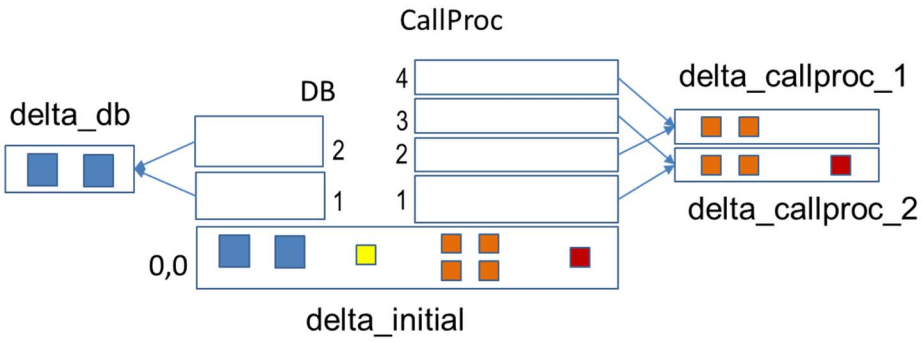

Figure A.2-2: Definition of the scaling deltas for an example VNF

Figure A.2-2 shows a VNF with two scaling aspects (DB and CallProc). The square filled boxes represent individual VNFC instances, the blue frame rectangles group these VNFC instances into scaling deltas, and the colour of the squares denotes the applicable VDU when for the VNFC instance.

The "DB" scaling aspect has two uniform scaling steps; the same delta "delta_db" is applied in each step. The "CallProc" scaling aspect has four non-uniform scaling steps, using two differently-composed scaling deltas "delta_callproc_1" and "delta_callproc_2" that are applied in an alternating way.

The initial delta "delta_initial" marks the smallest size of the VNF that can be instantiated. It is used as the baseline for any scaling operation and needs to be instantiated before any scaling delta can be added. In the example, an additional VNFC (denoted by the yellow square) is instantiated as part of the initial delta that is not subject to scaling (i.e. that does not appear in any scaling delta).

Figure A.2-3 shows the VNF instance based on the scaling model in figure A.2-2 fully scaled out, i.e. after first instantiating the initial delta, scaling out "DB" by two scaling steps (adding "delta_db" in each step), and scaling out "CallProc" by four scaling steps (adding delta_callproc_1, delta_callproc_2, delta_callproc_1, delta_callproc_2 in sequence).

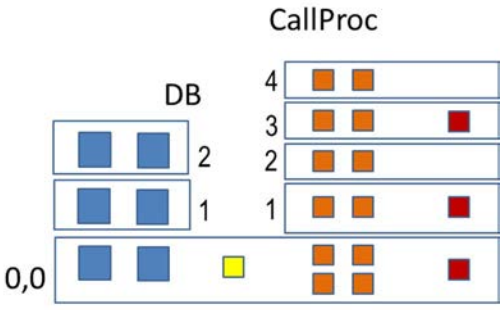

Figure A.2-3: Example VNF from figure A.2-2 fully scaled out

A.3 Scaling a VNF to a pre-defined target size

A VNF instance can also be scaled to a target size in one "ScaleToLevel" operation. Target sizes of the VNF that can be instantiated, and that can be reached by applying the "ScaleToLevel" operation, are defined as "instantiation levels" in the VNFD. An instantiation level describes a given amount of resources to be instantiated in terms of the number of VNFC instances to be created from each VDU and bit rate requirements.

Instantiation levels can also be represented in terms of scaling aspects. For that purpose, a scaling model is defined that combines the scaling aspects into a multi-dimensional space, representing each aspect as a dimension. Each possible size of the VNF is defined as a point in that scaling space, represented by a tuple in which each entry expressed the scale level (number of scaling steps applied) for a particular aspect.

Figure A.3-1 illustrates the resulting scaling space for the example VNF introduced in figure A.2-2.

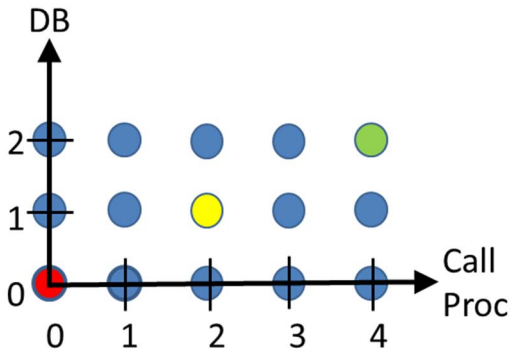

Figure A.3-1: Definition of VNF target sizes (instantiation levels) in terms of scaling aspects

The example in figure A.3-1 shows three instantiation levels, one at the minimum VNF size (represented by the red dot), one at an intermediate VNF size (represented by the yellow dot) and one at the maximum VNF size (represented by the green dot). Using the scaling model, the point marked by the yellow dot, for example, represents the tuple (DB=1, CallProc=2), i.e. one scaling step has been applied to the "DB" aspect and two scaling steps have been applied to the "CallProc" aspect.

The "ScaleToLevel" operation can be used to scale the VNF instance to a particular size, either by specifying one of the predefined instantiation levels ("red", "yellow", "green" in the example), or by specifying the target using a tuple such as (DB=1, CallProc=2) which is equivalent to the "yellow" instantiation level.

Annex B (informative): Use of affinity/anti-affinity scopes

B.1 Introduction

The "LocalAffinityOrAntiAffinityRule" and "AffinityOrAntiAffinityGroup" information elements specified in the present document enable the VNF designer to describe affinity or anti-affinity relationships among the constituents of the VNF. In both information elements:

- The "affinityOrAntiAffinity" attribute specifies the type of relationship. The "AFFINITY" value indicates that the related objects are expected to be placed on the same instance indicated by the scope. The "ANTI_AFFINITY" value indicates that the related objects are expected to be placed on different instances indicated by the scope.

- The "scope" attribute specifies the scope of the affinity/anti-affinity relationship.

The list of values in the "scope" determines the range of the affinity/anti-affinity with the expectation that the underlying infrastructure (where the virtualisation containers, VL and storage will be instantiated) has certain physical and/or logical distribution. Hence, the values that can be used in the "scope" reflect different layers of infrastructure distribution:

- NFVI-PoP: infrastructure level corresponding to a physical location. It is also sometimes used interchangeable with the term "site" and commonly mapped to a pool of NFVI resources administered as a data centre. See also the definition in ETSI GR NFV 003 [i.11].

- Zone: used as a short name for "resource zone". Resource zone refers to a set of NFVI resources logically grouped according to physical isolation and redundancy capabilities or to certain administrative policies for the NFVI. Particularly, a specific resource cannot be part of two different resource zones. See also the definition of "resource zone" in ETSI GR NFV 003 [i.11].

- ZoneGroup: used as a short name for "resource zone group". A resource zone group is a group of one or more related resource zones. This is typically used when resource zones are not elastic, and a set of resource zones with equivalent properties can be used to allow for overflowing resource allocations from one resource zone into another. Refer to the relevant description in clause 8.3.5 of ETSI GS NFV-IFA 007 [i.3].

- NFVI-node: infrastructure level corresponding to some physical device(s) deployed and managed as a single entity. Typically, this is commonly mapped to a physical server.

- CIS-node: infrastructure level corresponding to a CIS cluster node, which is a compute resource that runs a CIS instance or a CISM instance, or both. The CIS cluster node can either be physical (e.g. a server, and thus an NFVI-node), or virtual (e.g. a virtual machine).

- Network-link-and-node: network infrastructure level corresponding to network nodes and network links.

- Container-namespace: infrastructure level corresponding to a logical partition of a CIS cluster.

Figure B.1-1 illustrates the applicability of infrastructure distribution scopes including NFVI-PoP and network-link-and-node.

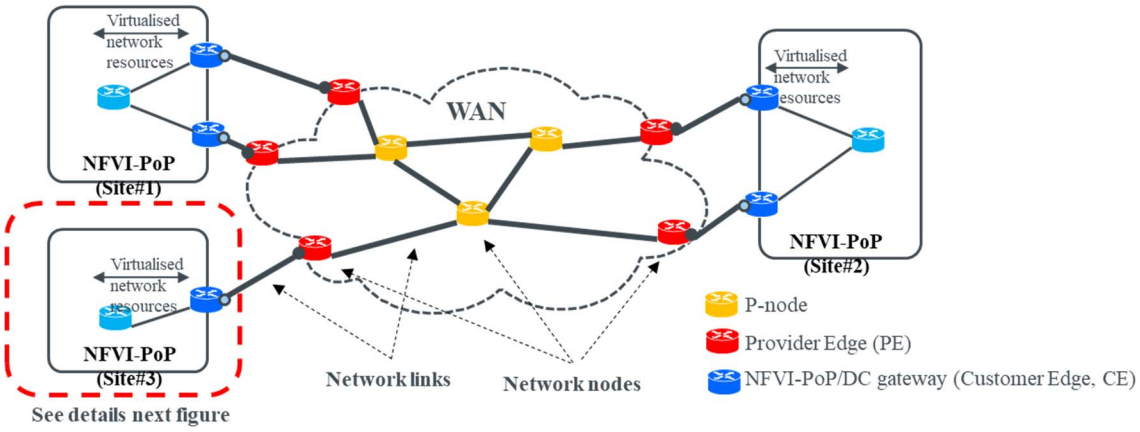

Figure B.1-1: Illustration of infrastructure distribution scopes including NFVI-PoP and network-link-and-node

Figure B.1-2 illustrates the applicability of infrastructure distribution scopes within an NFVI-PoP.

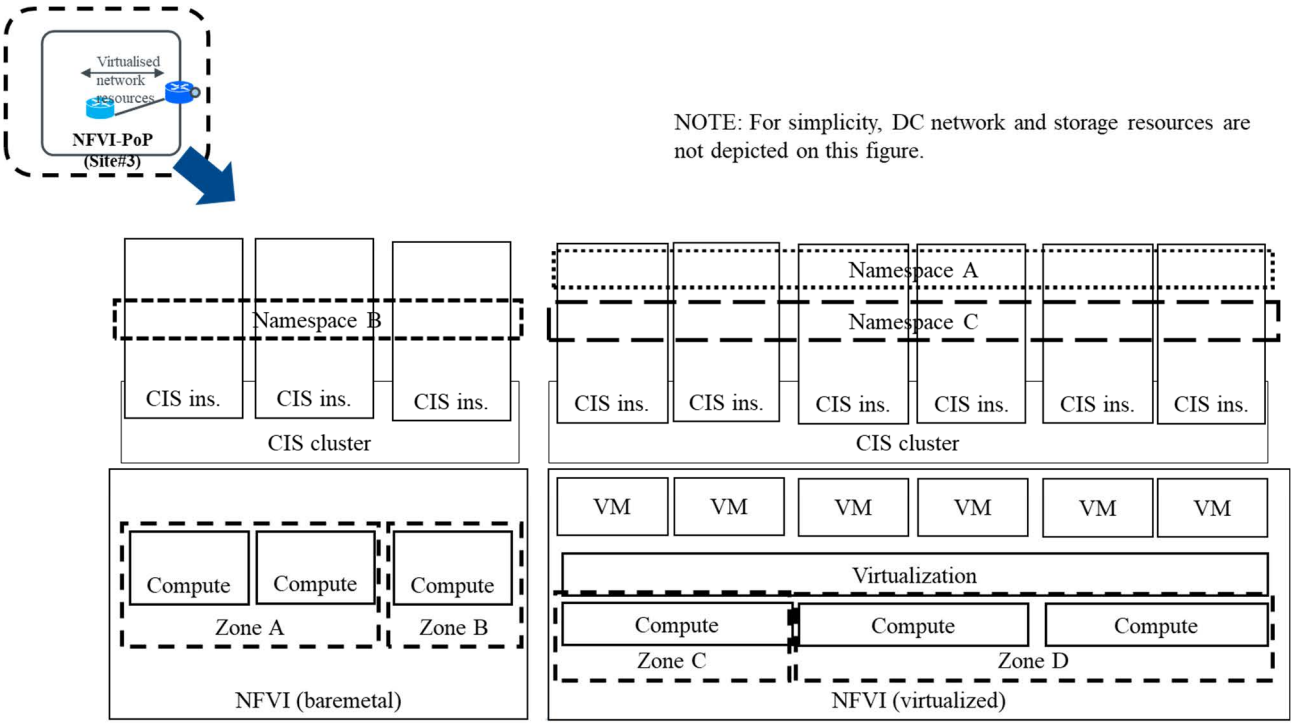

Figure B.1-2: Illustration of infrastructure distribution scopes including within NFVI-PoP

B.2 Use of affinity/anti-affinity scopes for container-based VNF

B.2.1 Overview

Since CIS clusters can also be virtual (e.g. based on virtual machines) there are additional layers of affinity/anti-affinity relationship. Hence, for container-based VNF, the scopes of "container-namespace" and "CIS-node" play a special role, in particular when reliability, availability and security requirements are expected to be fulfilled by the VNF.

For instance, considering a case where a CIS cluster is based on virtual machines and that two VNFC are expected to be instantiated on two different CIS cluster nodes, the following two cases might lead to different reliability and availability outcomes:

- the two CIS cluster nodes are instantiated onto the same NFVI node (server); or
- the two CIS cluster nodes are instantiated onto different NFVI nodes.

NOTE: In order to differentiate these cases, it is assumed that CISM can determine if two CIS cluster nodes are instantiated onto same NFVI node.

In the first case, even though the two VNFC instances would be instantiated onto two different CIS cluster nodes, a failure of the underlying same NFVI-node (server) could impact both VNFC instances. Furthermore, instantiating VNFC instances on different NFVI-nodes can help the VNF deliver better performance. Therefore, it is important that the VNF design considers CIS cluster nodes as well as underlying NFVI nodes when determining the expected affinity/anti-affinity of the VNF constituents.

B.2.2 Examples

B.2.2.1 Example of container-based VNF on physical (baremetal) CIS cluster

Figure B.2.2.1-1 illustrates an example of container-based VNF instantiated on a physical (baremetal) CIS cluster. Although not depicted on the figure B.2.2.1-1, all VNFC instances are assumed to be constituents of the VNF instance. The VNF is comprised of 9 VNFC instances from 5 different VDU/VDU profiles. In this example, the following affinity/anti-affinity relationships are assumed:

- For the VNFC based on VDU profile #1 (VNFC #1.x):
  - local (i.e. among VNFC instances of the same VDU profile) anti-affinity at the CIS node level and affinity at the Zone level;
  - group affinity with VNFC #2.x at the CIS node level;
  - group anti-affinity with VNFC #3.x and VNFC #4.x at the CIS node level; and
  - group affinity with VNFC #2.x, VNFC #3.x and VNFC #4.x at the Zone level.
- For the VNFCs based on VDU profile #2 (VNFC #2.x):
  - local (i.e. among VNFC instances of the same VDU profile) anti-affinity at the CIS node level and affinity at the Zone level (same as VNFC #1.x);
  - group affinity with VNFC #1.x at the CIS node level;
  - group affinity with VNFC #1.x, VNFC #3.x and VNFC #4.x at the Zone level.
- For the VNFCs based on VDU profile #3 (VNFC #3.x):
  - group anti-affinity with VNFC #1.x and VNFC #4.x at the CIS node level; and
  - group affinity with VNFC #1.x, VNFC #2.x and VNFC #4.x at the Zone level.
- For the VNFCs based on VDU profile #4 (VNFC #4.x):
  - local (i.e. among VNFC instances of the same VDU profile) affinity at the CIS node level;
  - group anti-affinity with VNFC #1.x and VNFC #3.x at the CIS node level; and
  - group affinity with VNFC #1.x, VNFC #2.x and VNFC #3.x at the Zone level.
- For the VNFCs based on VDU profile #5 (VNFC #5.x):
  - local (i.e. among VNFC instances of the same VDU profile) anti-affinity at the Zone level.

- For the MCIOPs:
  - VNFC #1.x, VNFC #2.x, VNFC #3.x and VNFC #4.x are deployed with MCIO descriptors that are part of MCIOP #1;
  - VNFC #5.x are deployed with MCIO descriptors that are part of MCIOP #2; and
  - there is anti-affinity relationship in between the MCIOP #1 and MCIOP #2 at the container-namespace level.

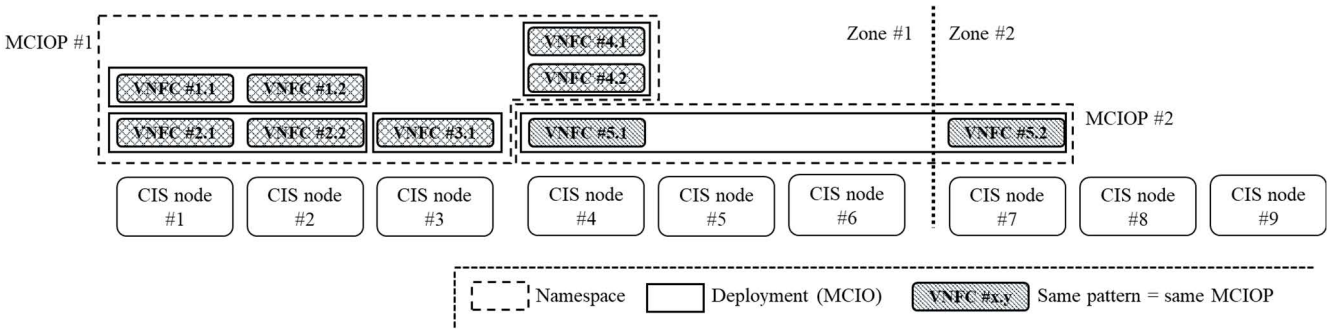

Figure B.2.2.1-1: Example of container-based VNF on physical (baremetal) CIS cluster

Below is an example of key attributes and values in the VNFD to realize the deployment as depicted on figure B.2.2.1-1:

- For the VNFCs based on VDU profile #1 (VNFC #1.x):

```
VduProfile #1:
  vduId: Vdu#1
  minNumberofInstances: 2
  maxNumberOfInstances: 2
  localAffinityOrAntiAffinityRule:
  - type: ANTI_AFFINITY
    scope: CIS-node
  - type: AFFINITY
    scope: Zone
  affinityOrAntiAffinityGroupId:
  - GroupNode#1
  - GroupNode#2
  - GroupZone#1
```

NOTE: GroupNode #1 and GroupNode#2 in VduProfile #1 do not lead to any conflict. GroupNode #1 and GroupNode #2 apply a relationship among VNFC instances created from different VDU profiles: for GroupNode#1 affinity, the relationship is between VNFC instances created from VDU profile #1 (i.e. VNFC #1.x) and VDU profile #2 (i.e. VNFC #2.x), and for GroupNode#2 anti-affinity, the relationship is between VNFC instances created from VDU profiles #1, #3 and #4 (i.e. VNFC #1.x, VNFC #3.x and VNFC #4.x). The only overlapping subset between these two groups is the VNFC #1.x, hence there is no conflict. Furthermore, GroupNode #1 does not apply for the relationship between VNFC instances created from the same VDU profile #1 (i.e. VNFC #1.1 and VNFC #1.2 in the example in figure B.2.2.1-1), because such relationship is instead determined by the "localAffinityOrAntiAffinityRule"; hence, the "local" anti-affinity and the "group" affinity do not conflict either.

- For the VNFCs based on VDU profile #2 (VNFC #2.x):

```
VduProfile #2:
  vduId: Vdu#2
  minNumberofInstances: 2
  maxNumberOfInstances: 2
  localAffinityOrAntiAffinityRule:
  - type: ANTI_AFFINITY
    scope: CIS-node
  - type: AFFINITY
    scope: Zone
  affinityOrAntiAffinityGroupId:
  - GroupNode#1
  - GroupZone#1
```

- For the VNFCs based on VDU profile #3 (VNFC #3.x):

```
VduProfile #3:
  vduId: Vdu#3
  minNumberofInstances: 1
  maxNumberOfInstances: 1
  affinityOrAntiAffinityGroupId:
  - GroupNode#2
  - GroupZone#1
```

- For the VNFCs based on VDU profile #4 (VNFC #4.x):

```
VduProfile #4:
  vduId: Vdu#4
  minNumberofInstances: 2
  maxNumberOfInstances: 2
  localAffinityOrAntiAffinityRule:
  - type: AFFINITY
    scope: CIS-node
  affinityOrAntiAffinityGroupId:
  - GroupNode#2
  - GroupZone#1
```

- For the VNFCs based on VDU profile #5 (VNFC #5.x):

```
VduProfile #5:
  vduId: Vdu#5
  minNumberofInstances: 2
  maxNumberOfInstances: 2
  localAffinityOrAntiAffinityRule:
    type: ANTI-AFFINITY
    scope: Zone
  affinityOrAntiAffinityGroupId:
```

- Affinity/anti-affinity definitions and MCIOP profiles grouping:

```
mciopProfile:
- mciopId: MCIOP#1
  affinityOrAntiAffinityGroup:
  - GroupNamespace#1
  associatedVdu:
  - Vdu#1
  - Vdu#2
  - Vdu#3
  - Vdu#4
- mciopId: MCIOP#2
  affinityOrAntiAffinityGroup:
  - GroupNamespace#1
  associatedVdu:
    - Vdu#5
```

- Affinity/anti-affinity groups definitions:

```
affinityOrAntiAffinityGroup:
- groupId: GroupNode#1
  affinityOrAntiAffinity: AFFINITY
  scope: CIS-node
- groupId: GroupNode#2
  affinityOrAntiAffinity: ANTI-AFFINITY
  scope: CIS-node
- groupId: GroupZone#1
  affinityOrAntiAffinity: AFFINITY
  scope: Zone
- groupId: GroupNamespace#1
  affinityOrAntiAffinity: ANTI-AFFINITY
  scope: container-namespace
```

B.2.2.2 Example of container-based VNF on virtual CIS cluster

Figure B.2.2.2-1 illustrates an example of container-based VNF instantiated on a virtual CIS cluster. Although not depicted on the figure B.2.2.1-1, all VNFC instances are assumed to be constituents of the VNF instance.

The VNF is comprised of 6 VNFC instances from 3 different VDU/VDU profiles. In this example, the following affinity/anti-affinity relationships are assumed:

- For the VNFCs based on VDU profile #1 (VNFC #1.x):
  - local (i.e. among VNFC instances of the same VDU profile) anti-affinity at the CIS node level and affinity at the Zone level;
  - group affinity with VNFC #2.x at the CIS node level; and
  - group affinity with VNFC #2.x and VNFC #3.x at the Zone level.
- For the VNFCs based on VDU profile #2 (VNFC #2.x):
  - local (i.e. among VNFC instances of the same VDU profile) anti-affinity at the CIS node level and affinity at the Zone level;
  - group affinity with VNFC #1.x at the CIS node level; and
  - group affinity with VNFC #1.x and VNFC #3.x at the Zone level.
- For the VNFCs based on VDU profile #3 (VNFC #3.x):
  - local (i.e. among VNFC instances of the same VDU profile) anti-affinity at the NFVI node level, and affinity at the Zone level; and
  - group affinity with VNFC #1.x and VNFC #2.x at the Zone level.
- For the MCIOPs:
  - VNFC #1.x, VNFC #2.x, and VNFC #3.x are deployed with MCIO descriptors that are part of MCIOP #1.

NOTE: In this example, VNFC #3.x are not only locally anti-affine at the CIS node level, but also at NFVI node level.

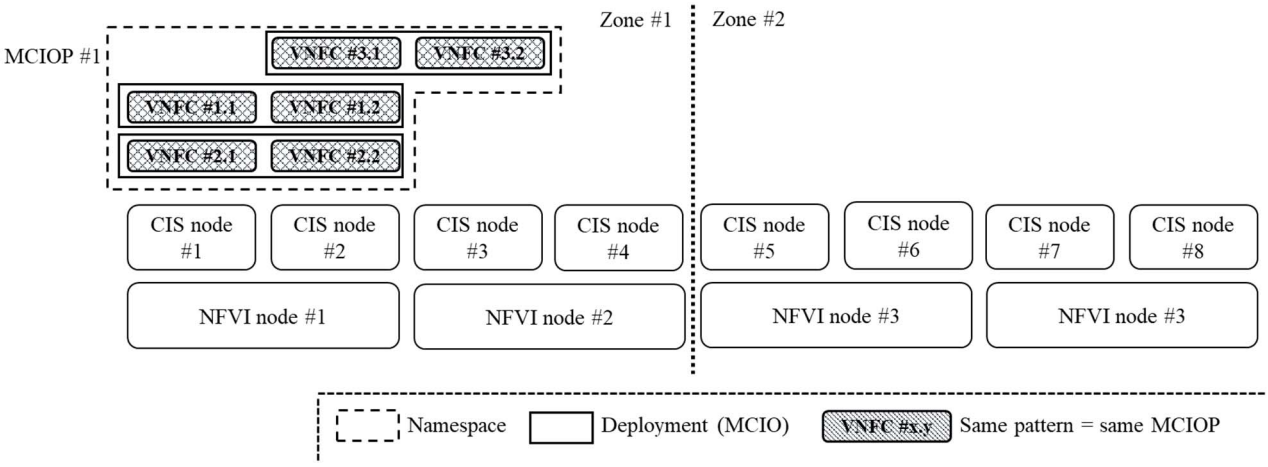

Figure B.2.2.2-1: Example of container-based VNF on virtual CIS cluster

Below is an example of key attributes and values in the VNFD to realize the deployment as depicted in figure B.2.2.2-1:

- For the VNFCs based on VDU profile #1 (VNFC #1.x):

```
VduProfile #1:
  vduId: Vdu#1
  minNumberofInstances: 2
  maxNumberOfInstances: 2
  localAffinityOrAntiAffinityRule:
  - type: ANTI_AFFINITY
    scope: CIS-node
  - type: AFFINITY
    scope: Zone
  affinityOrAntiAffinityGroupId:
  - GroupNode#1
  - GroupZone#1
```

- For the VNFCs based on VDU profile #2 (VNFC #2.x):

```
VduProfile #2:
  vduId: Vdu#2
  minNumberofInstances: 2
  maxNumberOfInstances: 2
  localAffinityOrAntiAffinityRule:
  - type: ANTI_AFFINITY
    scope: CIS-node
  - type: AFFINITY
    scope: Zone
  affinityOrAntiAffinityGroupId:
  - GroupNode#1
  - GroupZone#1
```

- For the VNFCs based on VDU profile #3 (VNFC #3.x):

```
VduProfile #3:
  vduId: Vdu#3
  minNumberofInstances: 2
  maxNumberOfInstances: 2
  localAffinityOrAntiAffinityRule:
  - type: ANTI_AFFINITY
    Scope: NFVI-node
  - type: AFFINITY
    Scope: Zone
  affinityOrAntiAffinityGroupId:
  - GroupZone#1
```

- Affinity/anti-affinity definitions and MCIOP profiles grouping:

```
mciopProfile:
- mciopId: MCIOP#1
  associatedVdu:
  - Vdu#1
  - Vdu#2
  - Vdu#3
```

- Affinity/anti-affinity groups definitions:

```
affinityOrAntiAffinityGroup:
- groupId: GroupNode#1
  affinityOrAntiAffinity: AFFINITY
  scope: CIS-node
- groupId: GroupZone#1
  affinityOrAntiAffinity: AFFINITY
  scope: Zone
```

Annex C (informative): Implementation of ephemeral storage

C.1 Introduction

In OS containers within a VNFC, shared ephemeral storage is useful as shown in the following use cases:

- Common configuration files among the OS containers.
- Unix and Unix-like domain sockets to communicate with the OS containers.
- Lock files controlled by the OS containers.

However, shared ephemeral storage is specified in a different way from isolated ephemeral storage which belongs to only one VM or one OS container. Therefore, it is important to elaborate comparison of different patterns of ephemeral storage in case of VM and OS container.

C.2 Examples

Figure C.2-1 shows three different patterns of ephemeral storage to be allocated:

- VNFC A: the VNFC A based on VM has a 20 GB ephemeral storage. The ephemeral storage is specified as a virtualDisk in VirtualComputeDesc information element.
- VNFC B: two OS containers belonging to the VNFC B respectively have 10 GB ephemeral storages. These ephemeral storages are not shared and the resource limits of them are respectively specified by ephemeralStorageResourceLimit in OsContainerDesc information element.
- VNFC C: two OS containers belonging to the VNFC C respectively have 8 GB ephemeral storages which are not shared and the resource limits of the isolated ephemeral storages are respectively specified by ephemeralStorageResourceLimit in OsContainerDesc information element as well as VNFC B. In addition to the isolated ephemeral storages, the VNFC C needs to use a 4 GB shared ephemeral storage belonging to the two OS containers at the same time. The amount of the shared ephemeral storage i.e. 4 GB is specified on top of ephemeralStorageResourceLimit in OsContainerDesc information element of any OS containers. In figure C.2-1, there are two examples shown: Option #1 and Option #2. In case of Option #1, each ephemeralStorageResourceLimit is respectively increased by 2 GB than the amount of each isolated ephemeral storage needed by each OS container. In case of Option #2, only the ephemeralStorageResourceLimit of OS container C-1 is increased by 4 GB than the amount of isolated ephemeral storage needed by the OS container.

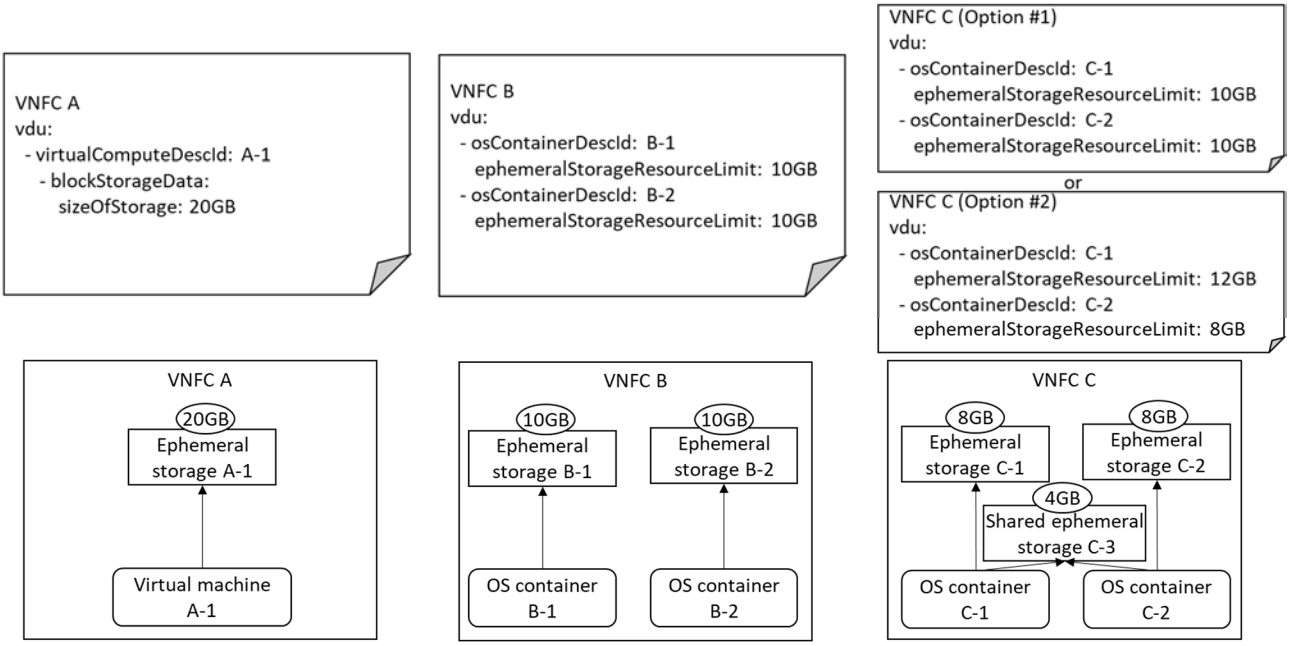

Figure C.2-1: Examples of different patterns of ephemeral storage

In conclusion:

a) ephemeral virtual storage of a virtual machine is specified in VirtualComputeDesc information element;

b) ephemeral virtual storage of an OS container is specified in OsContainerDesc information element; and

c) indeed the one shared by a set of OS containers instanced from a VDU is also specified in OsContainerDesc information element but needs to take into account specific calculation that the amount is added into any ephemeralStorageResourceLimit.

Annex D (informative): Change History

Date	Version	Information about changes
May 2017	V2.1.2	Update with CRs NFVIFA(17)234r1, NFVIFA(17)68, NFVIFA(16)1524r2.
June 2017	V2.1.3	Update with CRs NFVIFA(17)64r2, NFVIFA(17)437, NFVIFA(17)308r4, NFVIFA(17)445r1, NFVIFA(17)551r2, NFVIFA(17)503r2. Minor editorial updates.
June 2017	V2.3.1	Version update for plenary approval.
December 2017	V2.3.2	Update with CRs NFVIFA(17)000579r1, NFVIFA(17)000657r7, NFVIFA(17)000766r4, NFVIFA(17)000789r3, NFVIFA(17)000838r2, NFVIFA(17)000900r3, NFVIFA(17)000909r1, NFVIFA(17)000933, NFVIFA(17)000945r1, NFVIFA(17)000957r1, NFVIFA(17)000964r3, NFVIFA(17)001056r2, NFVIFA(17)001070, NFVIFA(17)001133r2.
February 2018	V2.4.1	Version update for publication.
March 2018	V2.4.2	Update with CRs NFVIFA(18)000141r2, NFVIFA(18)000142r1, NFVIFA(18)000163.
May 2018	V2.4.3	Update with CRs NFVIFA(18)000238 and NFVIFA(18)000331.
May 2018	V3.0.0	Release 3 baseline version created from draft v2.4.3.
June 2018	V3.0.1	Update with CRs NFVIFA(18)000554, NFVIFA(18)000555, NFVIFA(18)000589r1, NFVIFA(18)000591r1 and NFVIFA(18)000592r1.
June 2018	V3.0.2	Update with below maintenance CRs: NFVIFA(18)000593r1: IFA011ed311 MIRROR Improve modelling of scaling deltas NFVIFA(18)000602r2: IFA011ed311 MIRROR Fix to the scaling delta fix NFVIFA(18)000621r2: IFA011ed311 Rel3Mirror Adding bootdata parameter to the VNFD NFVIFA(18)000622: IFA011ed311 Rel3Mirror Support the Virtual Link Protocol Data in VNFD NFVIFA(18)000624: IFA011ed311 - nicIoRequirements NFVIFA(18)000627: IFA011ed311 Rel3Mirror Remove element groups NFVIFA(18)000634r1: IFA011ed311 Rel3Mirror scaling explanation NFVIFA(18)000676r1: IFA011ed311 Rel3Mirror Support Security Group in VNFD Update with below MegaCRs: NFVIFA(18)000540r1: IFA011 MegaCR FEAT15 VNF Snapshot
August 2018	V3.1.1	Version update for publication.
September 2018	V3.1.2	Update with CRs: NFVIFA(18)000812r1: IFA011ed321_Rel3Mirror_of_693r4_configurableProperties_correction NFVIFA(18)000813r1: IFA011ed321_Rel3Mirror_of_718r1_modifiableAttributes_correction
October 2018	V3.1.3	Update with CRs: NFVIFA(18)000746r4: NFVIFA_IFA011_8_4_2_4_cpumap NFVIFA(18)000792r1: IFA011_release_3_mirror_updating_Cpd_IE NFVIFA(18)000837: IFA011ed321 Rel3 Mirror VNFD support for using the Ve-Vnf-Vnfm reference point
October 2018	V3.1.4	Update with CR: NFVIFA(18)000807r1: IFA011 Clause 8 Functional requirements for VNF Snapshot Packaging
November 2018	V3.1.5	Update with CRs: NFVIFA(18)000956: IFA011ed321 Disambiguate checksum algorithm NFVIFA(18)000962: IFA011ed321 Rel3 Mirror of 858r2
December 2018	V3.1.6	Update with CRs: NFVIFA(18)0001093: IFA011ed321 Rel3 mirror of 1069r1 declaration of metadata and extensions NFVIFA(18)0001071r1: IFA011ed321 small changes in the description NFVIFA(18)0001072r1: IFA011ed321 fixing issue0007794
February 2019	V3.1.7	Update with below MegaCR: NFVIFA(19)000061r3: FEAT02 IFA011 MegaCR MegaCR NFVIFA(19)000061r3 only implemented partial content from CR NFVIFA(18)0001155r4 and this oversight was corrected in this version of the draft
February 2019	V3.1.8	Update with CRs: NFVIFA(19)000146: IFA011ed321 Clause 7-1-15-2 terminology alignment NFVIFA(19)000152r2: FEAT02 IFA011 Review modificationQualifier NFVIFA(19)000153r2: FEAT02 IFA011 Review fixes to VnfLcmOperationCoordination IE NFVIFA(19)000154: FEAT02 IFA011 Review declare LCM coordination interface in VnfInterfaceDetails NFVIFA(19)000163r1: CR to IFA011ed321 on individual artefact signature
April 2019	V3.2.1	Version update for publication

Date	Version	Information about changes
April 2019	V3.2.2	Update with CRs: NFVIFA(19)000293: IFA011ed321 Rel3-Mirror ONAP alignment – Class SwImageDesc NFVIFA(19)000292: IFA011ed331 rel-3 mirror ONAP alignment – Class VnfcConfigurableProperties NFVIFA(19)000271r1: IFA011ed331 rel-3 mirror clarification on securityGroupRule NFVIFA(19)000261: IFA011ed331 Change Log in the VNF Package
June 2019	V3.2.3	Misc Rapporteur corrections (case changes, line breaks in some attribute labels, etc). Update with CRs: NFVIFA(19)000429r1: IFA011ed331_rel-3_mirror_vNIC_type_value NFVIFA(19)000259r6: IFA011ed331 VipCpd for virtual IP addresses - Solution 2 NFVIFA(19)000485: IFA011ed331 Rel-3 mirror ONAP alignment – Class VirtualLinkProfile NFVIFA(19)000517: IFA011ed331 7_1_5_2 Terminology correction NFVIFA(19)000492: IFA011ed331_ SecurityGroupRule
July 2019	V3.2.4	Misc Rapporteur corrections. Update with CRs: NFVIFA(19)000562: IFA011Ed331 - Standard configurable properties NFVIFA(19)000626: IFA011ed331 Rel3Mirror 7.1.8.6 LinkBitrateRequirements IE NFVIFA(19)000632: IFA011ed331 Rel3Mirror 7.1.8.10 QoS NFVIFA(19)000646: IFA011ed331 Rel3Mirror 7.1.7.3 ConnectivityType IE NFVIFA(19)000039r7: IFA011 MegaCR FEAT03 NFVI MOD NFVIFA(19)000319r3: FEAT03: IFA011 Replacing scale levels NFVIFA(19)000745: IFA011ed331 Rel3Mirror of 427r1 Removal of supportMandatory attribute
September 2019	V3.3.1	Version update for publication
October 2019	V3.3.2	First draft for ed341
December 2019	V3.3.3	Misc Rapporteur corrections (remove Appendix on Authors & Contributors). Update with CRs: NFVIFA(19)000982r2: IFA011ed341 FixedIpAddresses NFVIFA(19)000971: IFA011ed341 FEAT02 fixing referenceability of changeCurrentVnfPkgOpConfig NFVIFA(19)000969: IFA011ed341 FEAT02 fixing TBD NFVIFA(19)000933: IFA011ed341 7.1.8 Add "network-link-and-node" value to AffinityOrAntiAffinity
February 2020	V3.3.4	Update with CRs: NFVIFA(19)000869r7: IFA011ed341 Adding Trunk port logical topology descriptions NFVIFA(20)000086: IFA011ed341 sync to IFA015 work according to 942r5 part1 NFVIFA(20)000114r3: IFA011ed341 harmonization corrections based on 942 part2 NFVIFA(20)000115: IFA011ed341 harmonization corrections based on 942 part33
May 2020	V3.3.5	Update with CRs: NFVIFA(20)000171r2: IFA011ed341 fix Enum values NFVIFA(20)000229r3: IFA011_FEAT15_vnf_snapshot_package_content
June 2020	V4.0.1	Release 4 baseline version created from published version v3.4.1
September 2020	V4.0.2	Update with CRs: NFVIFA(20)000479r6: IFA011ed411 MegaCR FEAT17 Cloud-native VNFs NFVIFA(20)000483r1: IFA011ed411 Add missing Dependencies IE definition NFVIFA(20)000504: IFA011ed411 mirror of 387r1 Fix Typos in clause 7-1-6 sync to IFA015 work NFVIFA(20)000540r2: IFA011ed411 clarification on vnfdId attribute in VNFD information element
September 2020	V4.0.3	Update with CR: NFVIFA(20)000442r5: FEAT17-IFA011ed411-New CP type to model networking MCIOs
November 2020	V4.1.1	Version update for publication
November 2020	V4.1.2	First draft for ed421
December 2020	V4.1.3	Rapporteur correction (editorial): Change "Artefact" to "artifact" Update with CRs: NFVIFA(20)000684: IFA011ed421 Rel-4 mirror updates related to the use of VIPs and floating IP addresses NFVIFA(20)000672: IFA011ed411 Rel4 mirror of 614 updates of scaling descriptors related to the use of VIPs
January 2021	V4.1.4	Update with CR: NFVIFA(20)000929: IFA011ed421 Rel 4 mirror content type of boot order

Date	Version	Information about changes
February 2021	V4.1.5	Update with CRs: NFVIFA(21)000078r1: IFA011ed421 rel 4 mirror Requirements for security and integrity of a VNF Snapshot Package NFVIFA(20)000841r1: ENH02.05 IFA011ed421 Adding support of target level VNF instantiation in the VNFD
March 2021	V4.1.6	Update with CRs: NFVIFA(21)000203r1: IFA011ed421 MegaCR FEAT17 Cloud-native VNFs NFVIFA(21)0000179: Enh02.04-IFA011ed421 MegaCR NFVIFA(21)000125r1: IFA011ed421 Rel4 mirror of 117 Cross stages alignment w.r.t. LCM coordination NFVIFA(21)000175: IFA011_release_4_mirror_clarification_for_virtualLinkProtocolData
March 2021	V4.1.7	Update with CR: NFVIFA(21)000263r1: IFA011Ed421 Software Images in a VNF Package
May 2021	V4.2.1	Version update for publication
July 2021	V4.2.2	First draft for ed431 Misc Rapporteur actions (delete supportMandatory attribute marked deprecated in v3.3.1 from VirtualNetworkInterfaceRequirements information element)
September 2021	V4.2.3	Update with CRs: NFVIFA(21)000747: IFA011ed431-Add missing Note extension from ed421 NFVIFA(21)000767r1: IFA011ed431 Add attributes to OsContainerDesc
October 2021	V4.2.4	Update with CRs: NFVIFA(21)000970: IFA011ed431-Correct SwImageDesc attribute applicability NFVIFA(21)000942r2: IFA011ed431-Enhance applicability of additionalNetworkInterfaceRequirements NFVIFA(21)000902: IFA011ed431- Add VirtualCpd constraints on inherited attributes NFVIFA(21)000841: IFA011ed431 FEAT03 Mirror of 840 Add missing minNumberofPreservedInstances attribute to NfviMaintenanceInfo IE NFVIFA(21)000771r5: IFA011ed431-Add CNI requirements to VirtualNetworkInterfaceRequirements
December 2021	V4.2.5	Update with CR: NFVIFA(21)000889: IFA011ed431 Rel.4 mirror of 888r3 Adding externallyManaged attribute to VnfVirtualLinkDesc
March 2022	V4.2.6	Update with CRs: NFVIFA(22)000172r1: IFA011Ed431 Duplicated VirtualStorageDesc NFVIFA(22)000148: IFA011ed431 Remove Reference to MAN 001 (Rel-4 mirror of NFVIFA(22)000147) NFVIFA(22)000072r1: IFA011ed431 SwImageDesc properties NFVIFA(22)000012: IFA011ed431 Mcio data NFVIFA(21)000998r3: IFA011ed431 Rel.4 mirror of 978r3 Fixing IpAddressAssignment flag NFVIFA(21)0001088r1: IFA011ed431 Rel.4 mirror of 1070r5 macAddressAssignment fixes
June 2022	V4.3.1	Version update for publication
July 2022	V4.3.2	First draft for ed441 created from published version v4.3.1
August 2022	V4.3.3	Update with CRs: NFVIFA(22)000469: IFA011ed441_Rel4_mirror_Clause_1_Scope_update NFVIFA(22)000534r3: IFA011ed441_Optional_license_info_IFA034_followup
September 2022	V4.3.4	Update with CRs: NFVIFA(22)000578: IFA011ed441 Update description of virtualComputeDesc in Vdu IE NFVIFA(22)000623: IFA011ed441_7.1.6.13_adding_CPU_pinning_requirements NFVIFA(22)000624r1: IFA011ed441_7.1.9.3_adding_huge_pages_requirements NFVIFA(22)000634: IFA011ed441_ 6.2.6_adding_DSL_reqts_for_CISM_CIR NFVIFA(22)000688: IFA011ed441_ 6.2.6_adding_missing_DSL_reqts
January 2023	V4.3.5	Update with CRs: NFVIFA(22)000734r1: IFA011ed441 Support of IP dual stack cases NFVIFA(22)000831: IFA011ed441_7.1.9.4.2.2_removing_space_from_attribute_name
January 2023	V4.3.6	Update with CR: NFVIFA(23)000022r1: Feat17 IFA011ed441 support of floating IP address for containerized VNFCs

History

Document history		
V4.1.1	November 2020	Publication
V4.2.1	May 2021	Publication
V4.3.1	June 2022	Publication
V4.4.1	March 2023	Publication