

Network Functions Virtualisation (NFV)

Understanding the concepts and technical foundations

Bruno Chatras

December 2018

Agenda

Part I: Introduction, Architecture, Challenges

Part II: Focus on Management and Orchestration

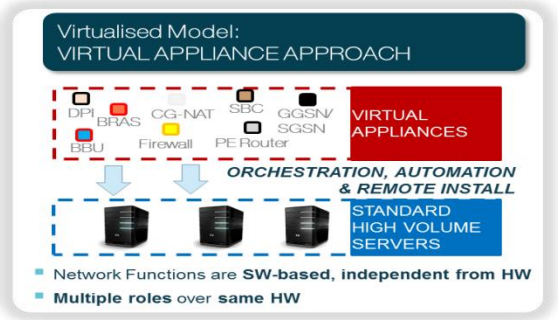
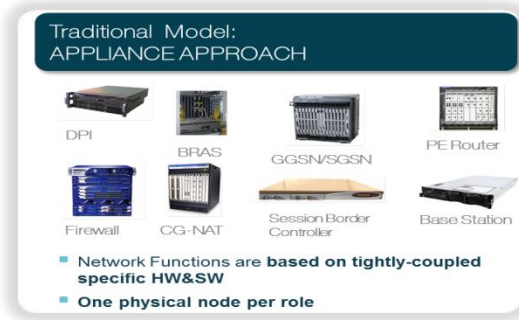
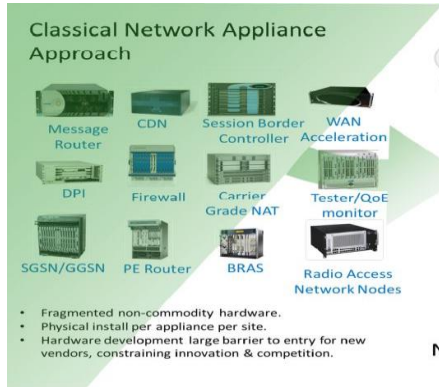
Part III: NFV, SDN and Service Chaining

Part IV: NFV, Network Slicing and 5G



Introduction
Concepts and Architecture
Technical Challenges

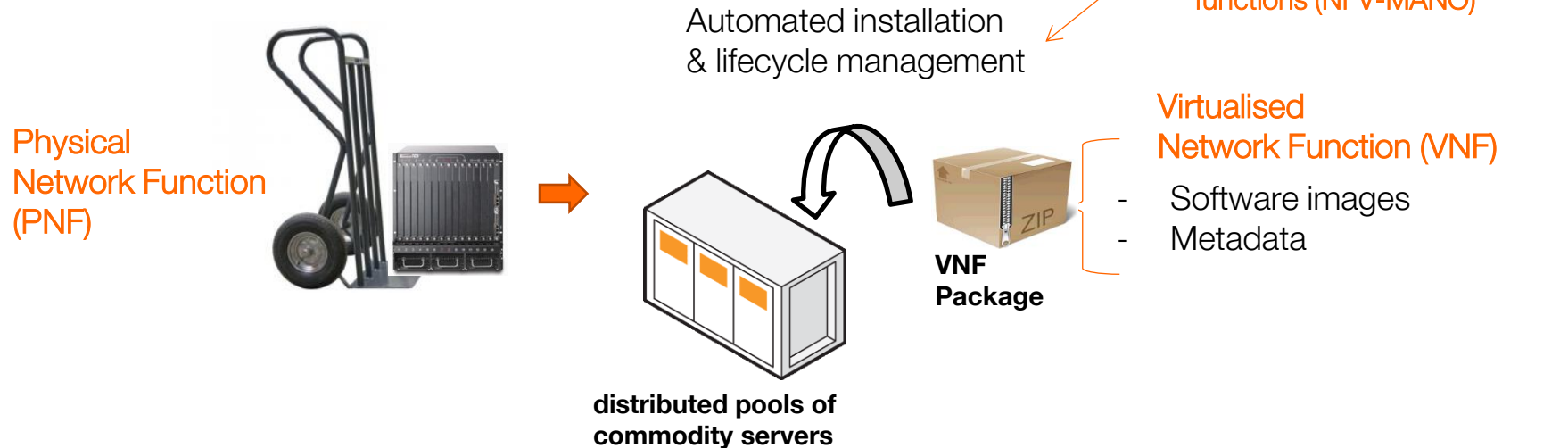
Network Functions Virtualisation (NFV) in pictures!



Network Functions Virtualisation in a Nutshell

Relocating network functions **from dedicated appliances to pools of generic industry servers**, leveraging:

- **Cloud Computing Technology**
- **Virtualisation Technologies**
- **Advances in general purpose processors performance**



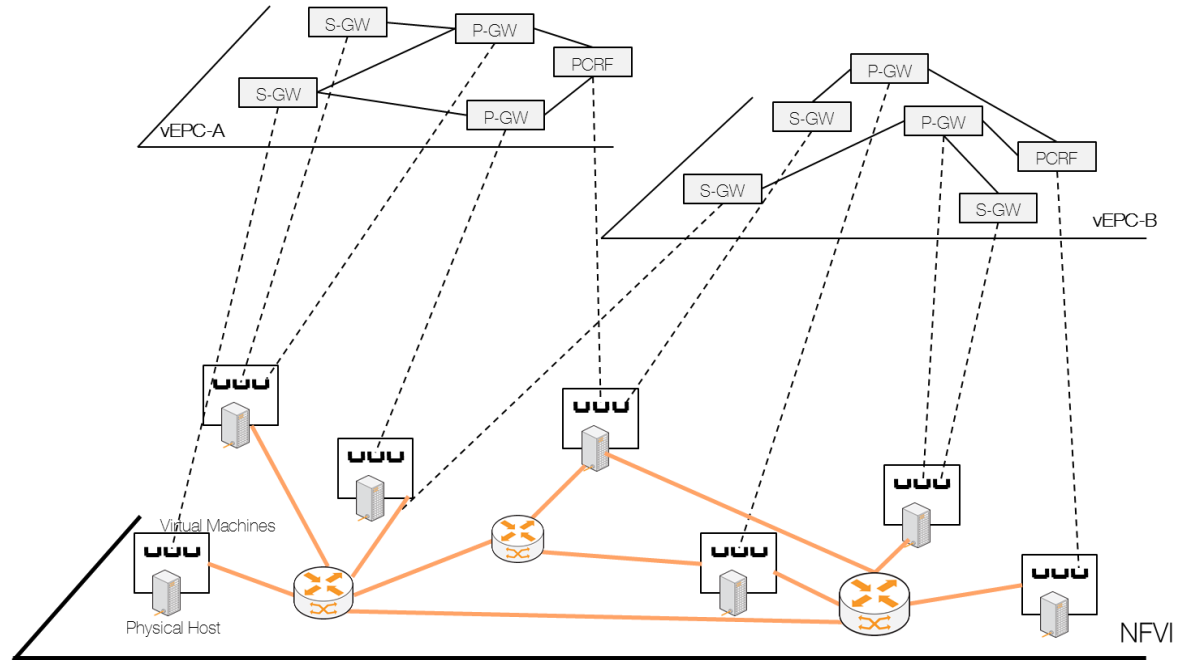
Expected Benefits: Cost Reduction and Increased Agility

Lower CAPEX (commodity servers)
and OPEX (high automation)

Greater **flexibility to scale** up and
down resources assigned to
applications based on actual usage

Reduced time-to-market to deploy
new or upgraded network services

Ability to handle **several tenants** on
the same infrastructure



Agenda

Part I: Introduction, Architecture, Challenges

Part II: Focus on Management and Orchestration

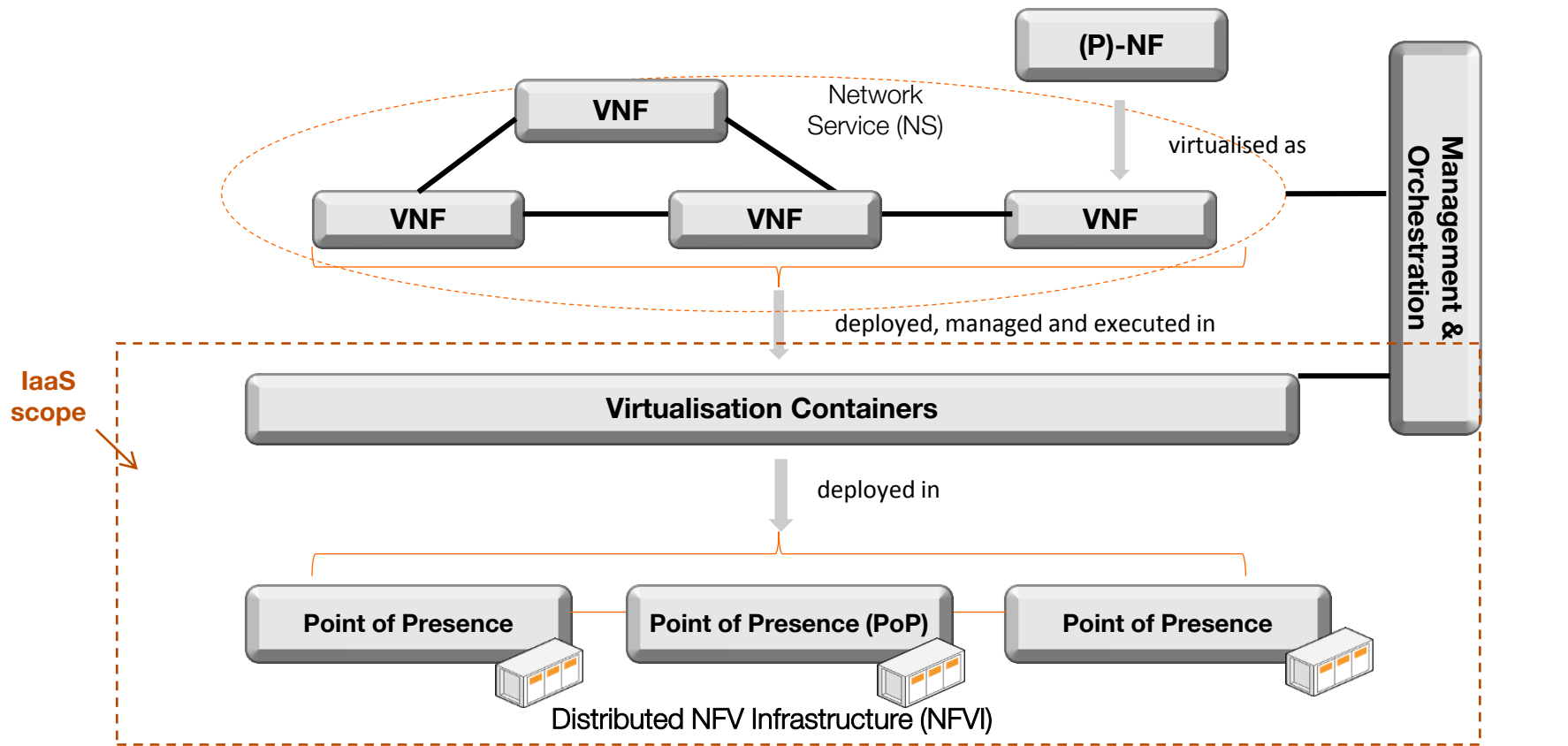
Part III: NFV, SDN and Service Chaining

Part IV: NFV, Network Slicing and 5G



Introduction
Concepts and Architecture
Technical Challenges

Architectural concepts



The NFVI is a distributed infrastructure

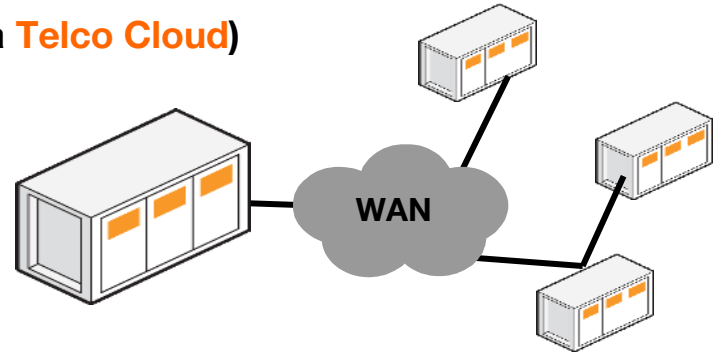
An **NFV Infrastructure** comprises one or more points of presence and is thus a **Distributed Cloud** (sometimes referred to as a **Telco Cloud**)

Examples of **NFVI point of presences** include

Highly centralized data centres (DCs)

Local / Regional network points of presence (PoPs)

... and Customer Premises



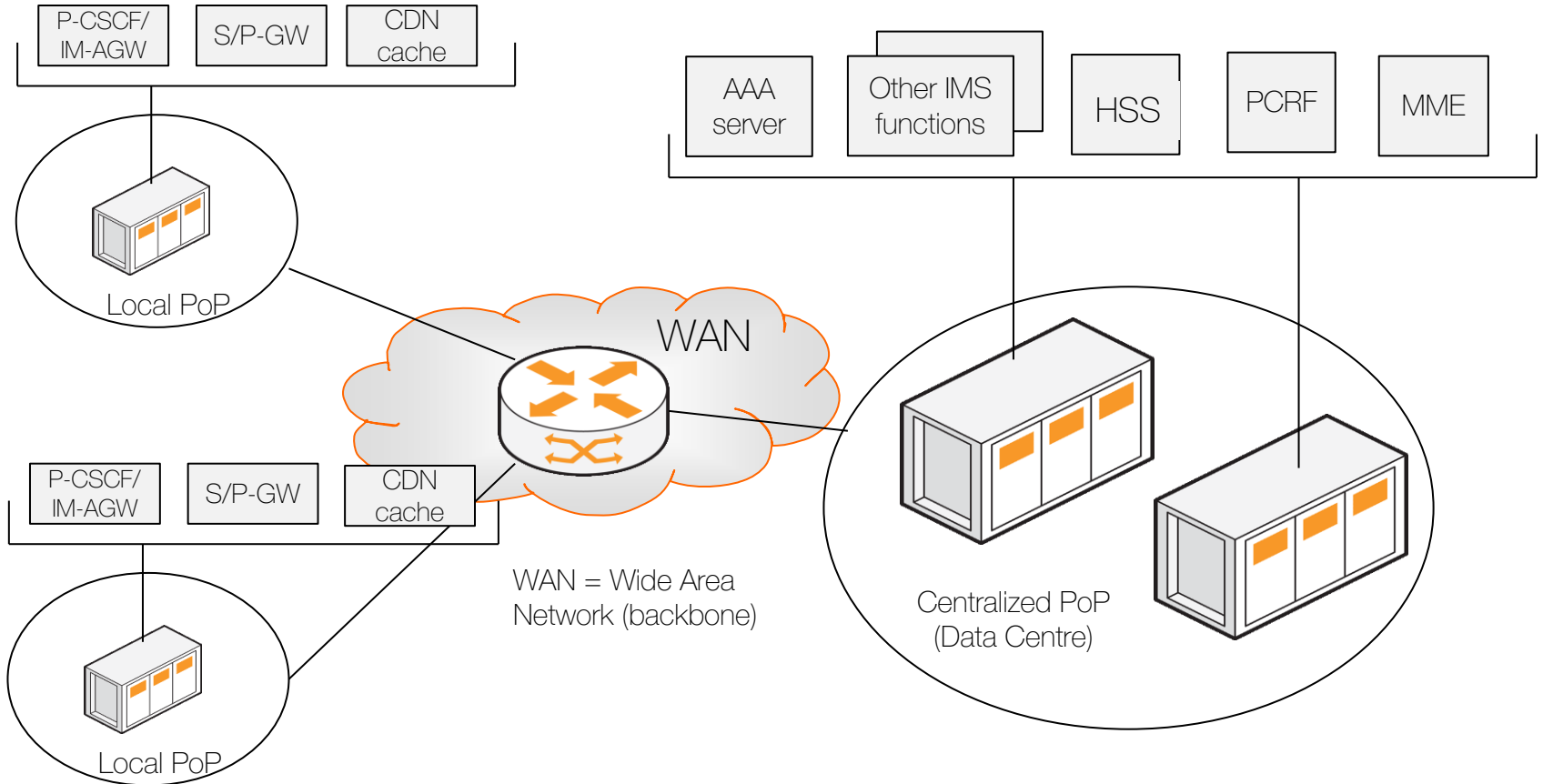
The **location** of a virtualised network function has a direct impact on the **end-to-end quality of experience** (latency):

– Rule of thumb: Data plane functions (e.g. CDN) in local/regional PoPs, control functions (e.g. IMS) in DCs

“Centralize what you can, distribute what you must”

Pradeep Sindhu, Founder, Juniper Networks

A distributed NFVI rather than a huge centralized Cloud



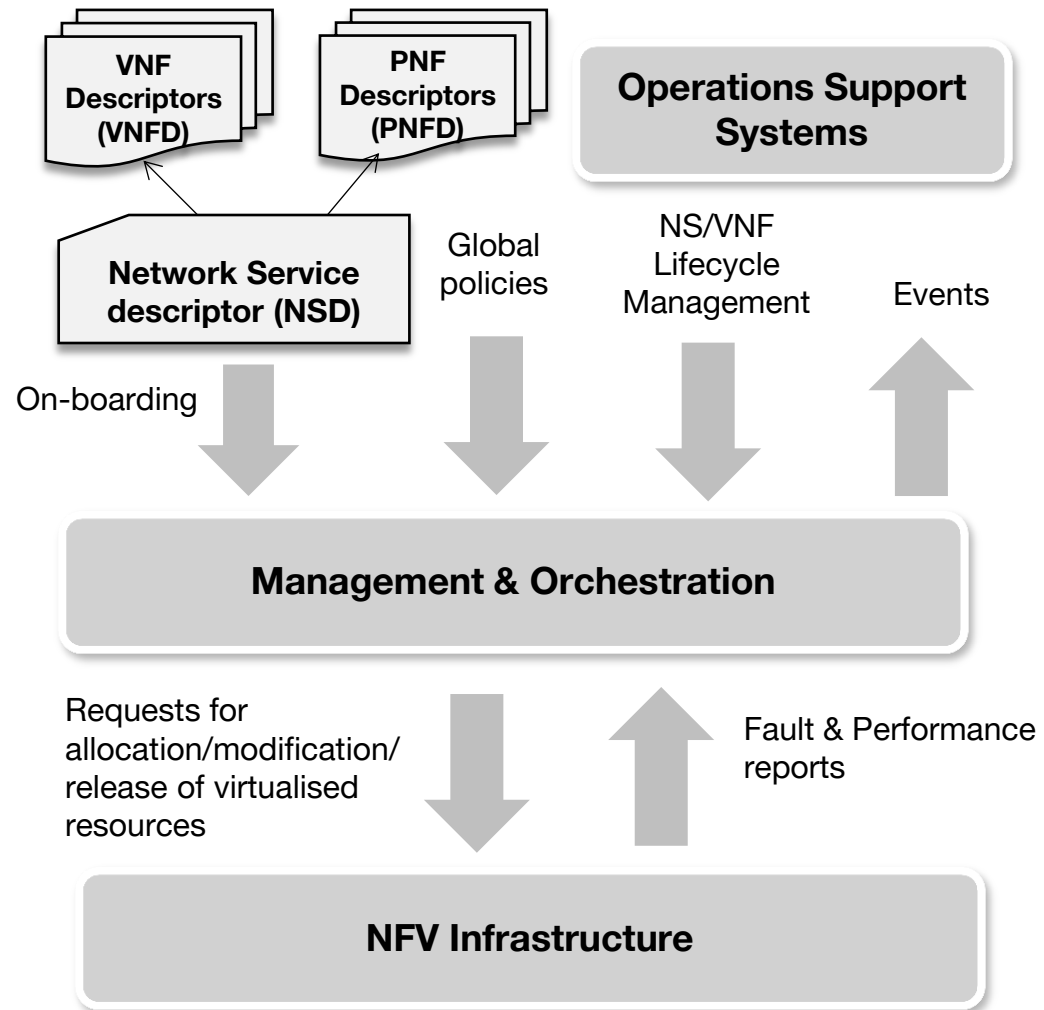
A data model driven system

NFV management and orchestration procedures are driven by a set of **machine-readable deployment templates** that include:

- Resource Requirements
- Deployment Constraints
- Lifecycle management policies and scripts

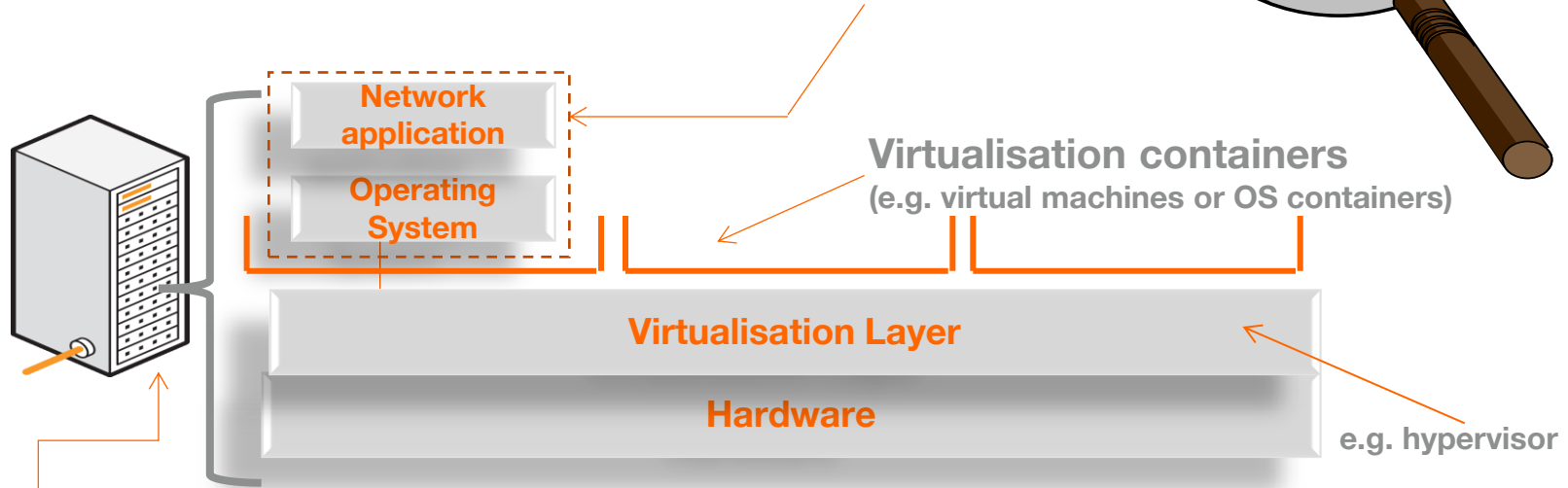
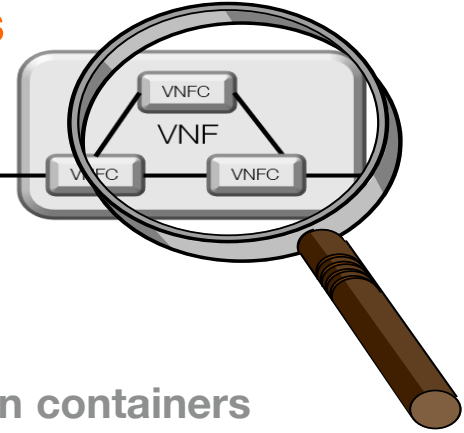
High automation of network operations and monitoring is expected to reduce:

- The time to deployment - **in minutes rather than months**
- The time to repair
- and the risk of misconfigurations



VNFs, VNF components and Virtualisation Containers

A VNF typically contains **several components** (VNFCs) each running in its own virtualisation container.



Commercial Off The Shelf (COTS) servers
with General Purpose Processors (e.g. **x86** or ARM-based)

≠ Application Specific Integrated Card (ASIC)

Options for the Virtualisation Layer

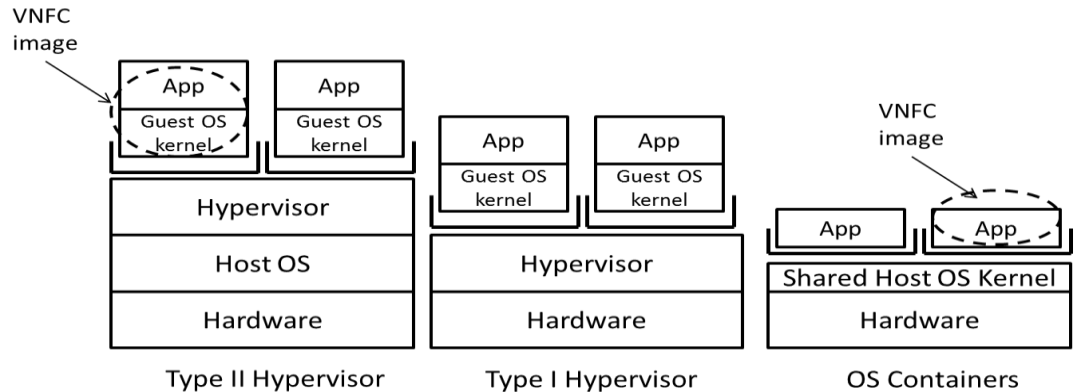
Pros & Cons analysis
(performance, security, density, etc.)
in [GS NFV EVE 004](#)

Hypervisor

Enables VNF providers to choose the VNF's OS, which can be a fully-fledged OS or a “Just Enough Operating System”, depending on desired instantiation time and memory footprint

OS Containers

OS imposed by the infrastructure provider (e.g. Linux / Docker)



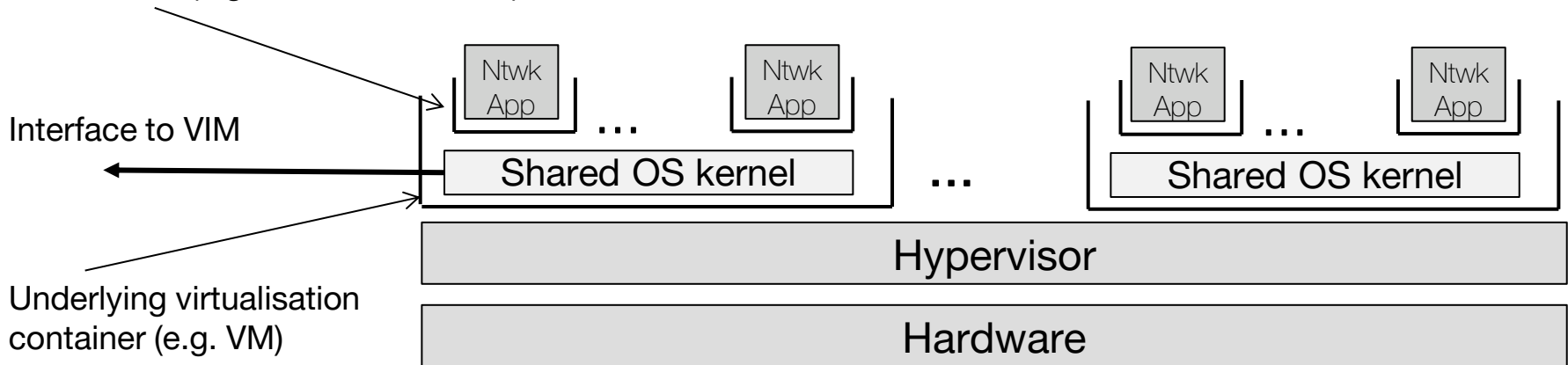
Nested Virtualisation

The virtualisation layer may be composed of multiple nested sub-layers, each using a different virtualisation technology.

Top sub-layer: Visible to the VIM, The partitions it creates provide the role of the NFV “virtualisation container”.

Other sub-layers: May or may not be visible to the VIM

NFV virtualisation container (e.g. Docker container)

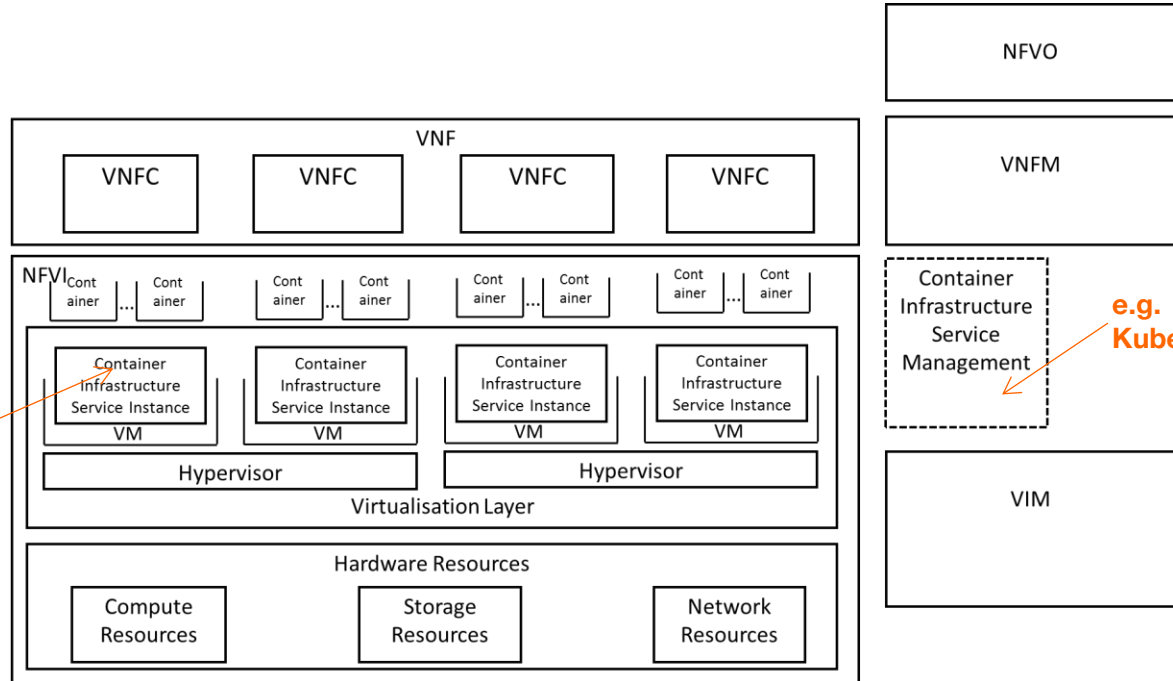


Container Infrastructure Service Management

Some de-facto industry solutions enable a **1-N mapping** between VNFC instances and Containers.

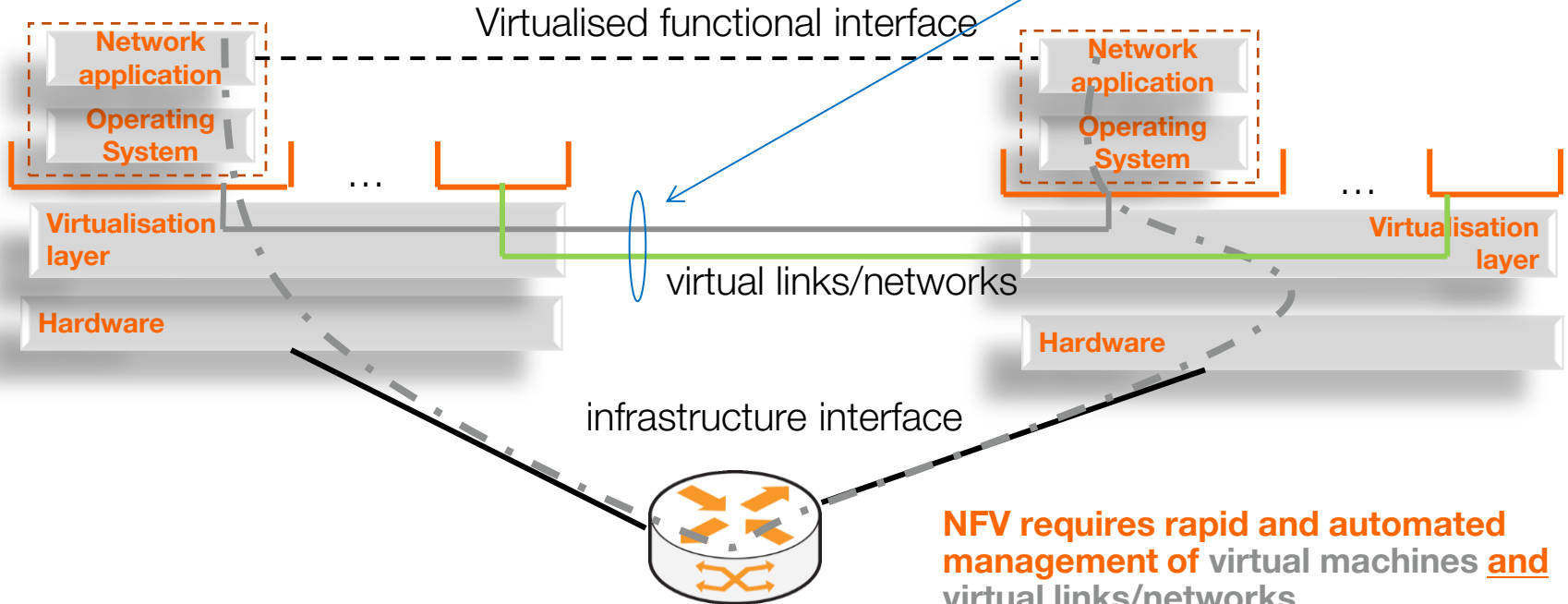
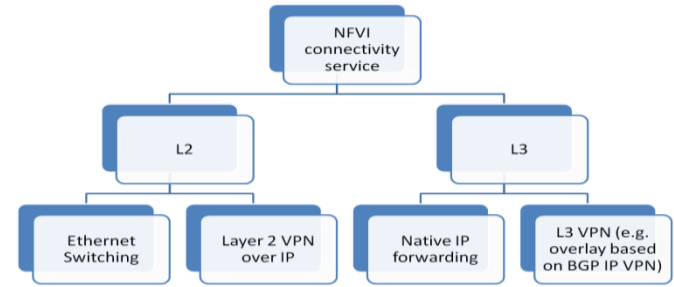
Impact on NFV-MANO under study in ETSI GR NFV-IFA 029 (NFV Release 3).

e.g.
Docker



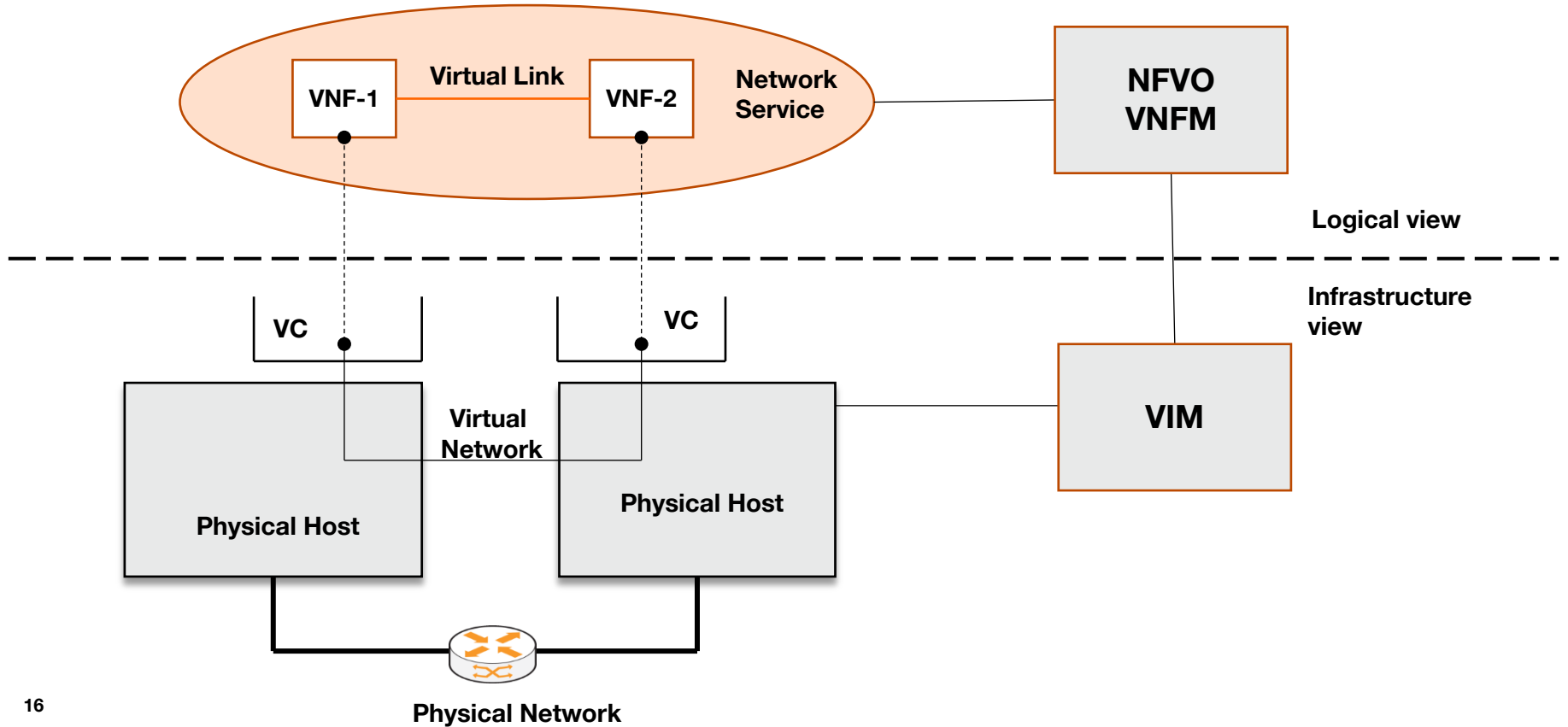
e.g.
Kubernetes

VNF to VNF interfaces

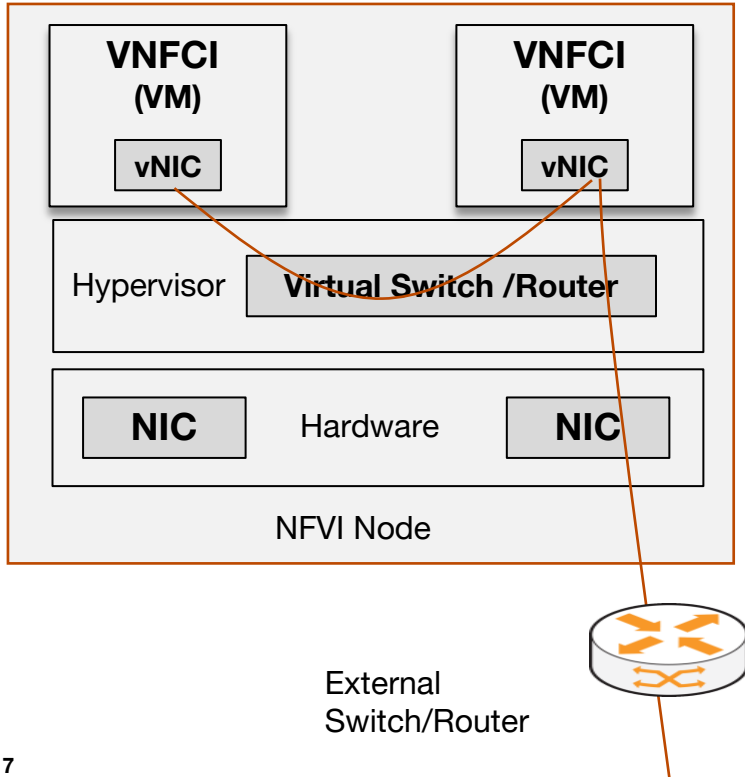


NFV requires rapid and automated management of virtual machines and virtual links/networks

Virtual Link vs. Virtual Network



Basic communication patterns (hypervisor case)



Many more options (including communication via shared memory)

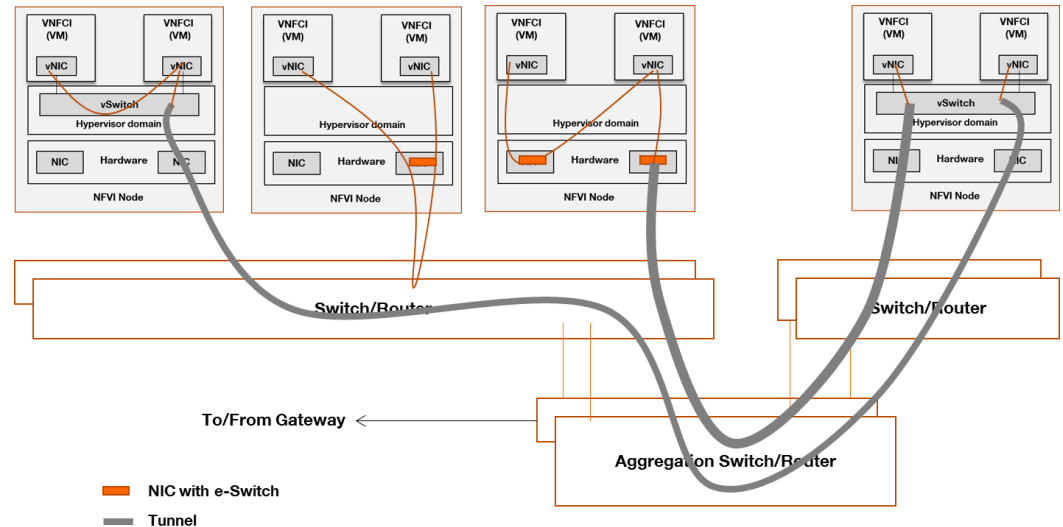
NIC = Network Interface Card

Overlay technologies in NFV

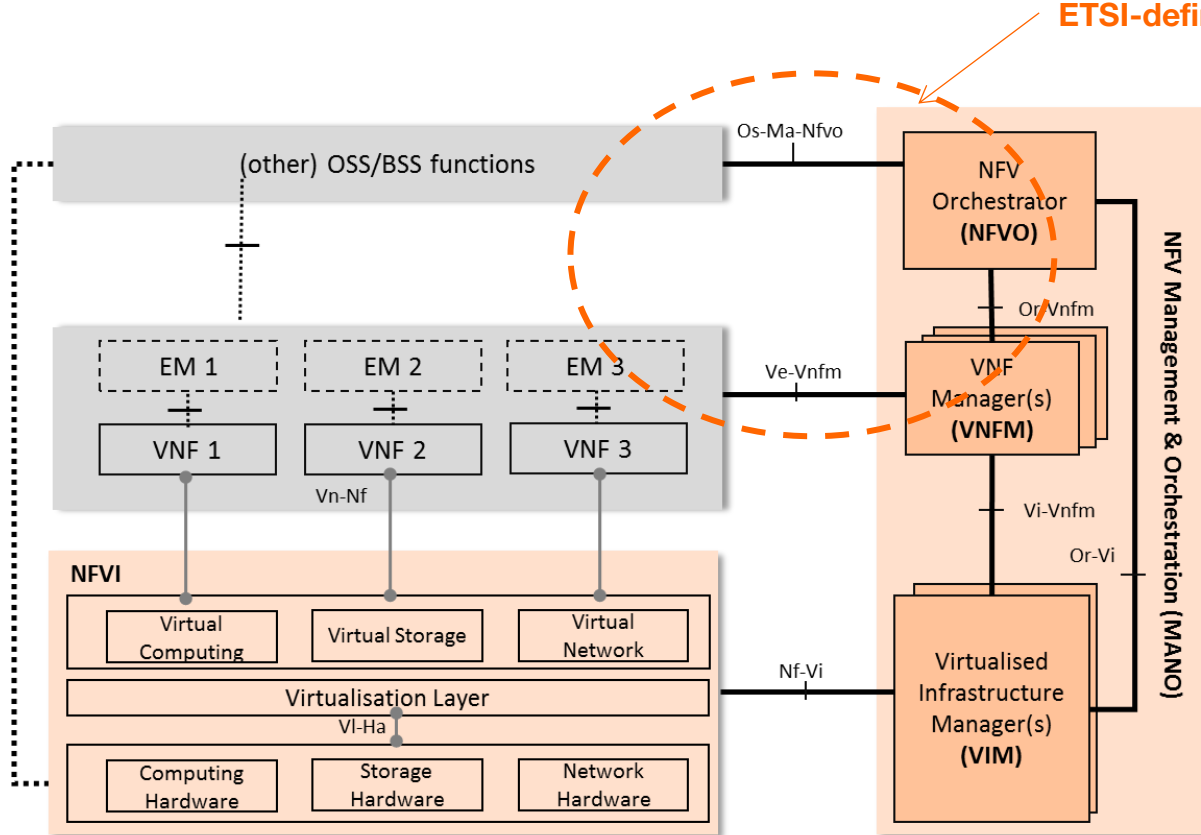
Virtual Links between VNFC instances are deployed as overlay tunnels between vSwitches/vRouters in NFVI Nodes and Gateways to the WANs.

Widely used, to minimize configuration needs on physical routers in an NFVI each time a virtualisation container is created.

– These routers form the **underlay**



The ETSI NFV architectural framework

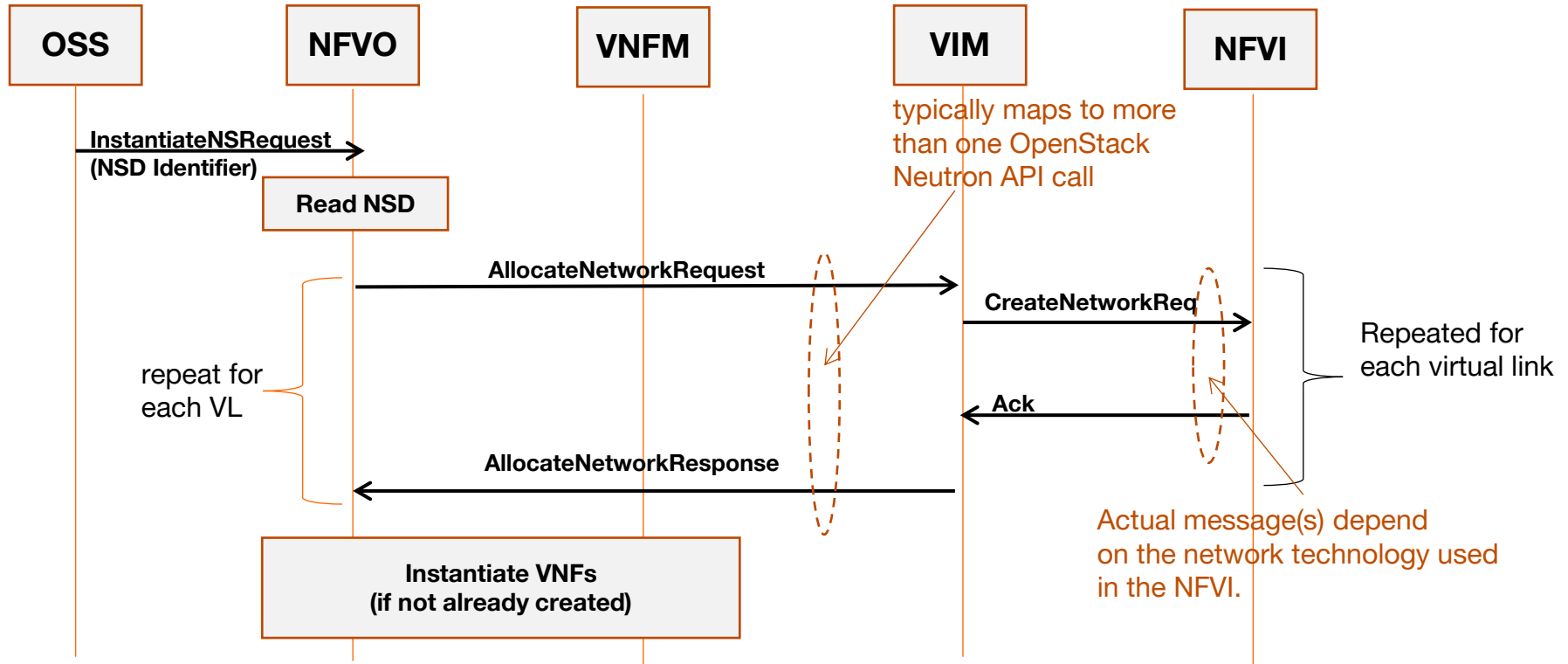


This is a **logical model**, not an implementation view.

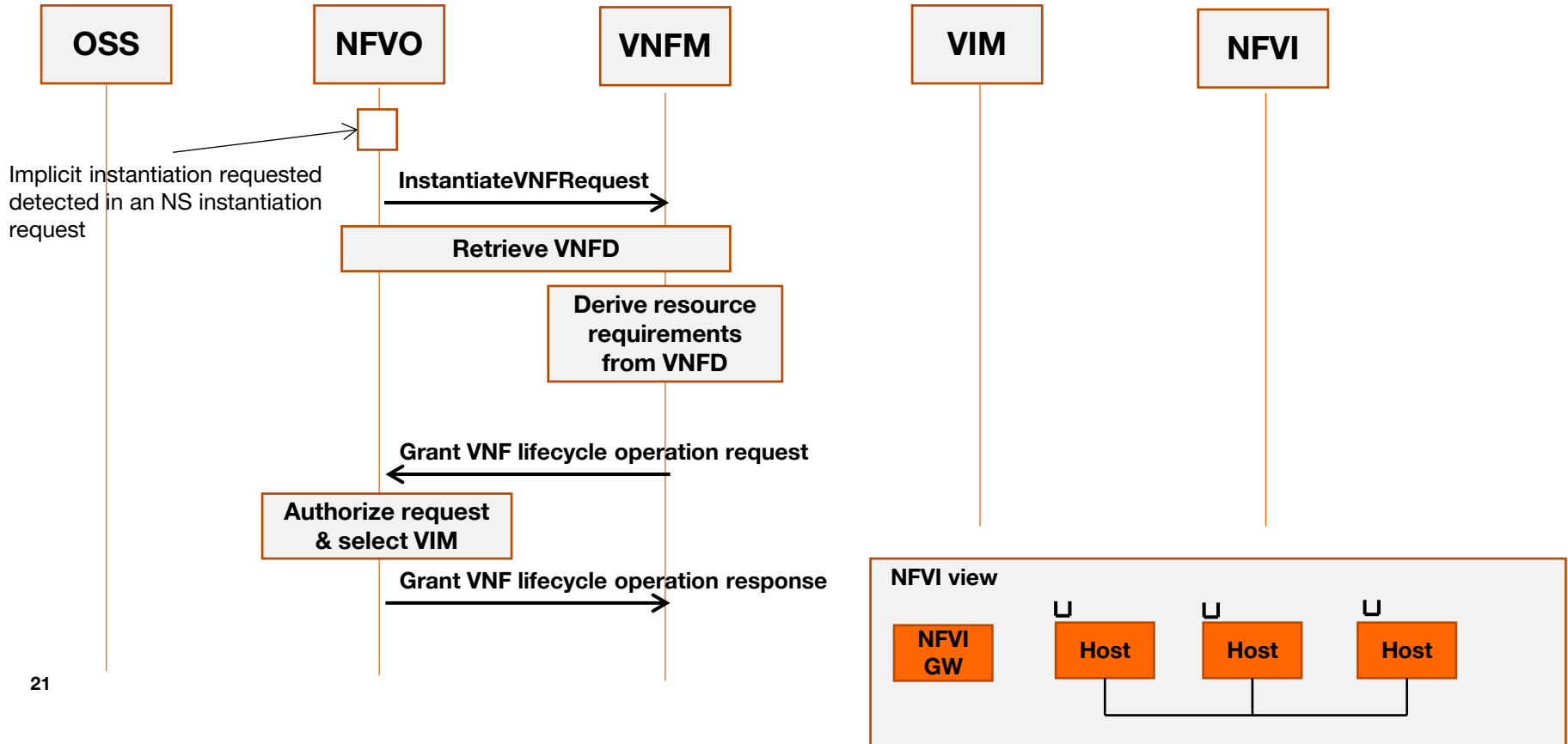
De-facto standards (e.g. **OpenStack APIs**) are used at the northbound interfaces of VIMs.

Standard **NFV-specific REST APIs** have been defined by ETSI for other reference points.

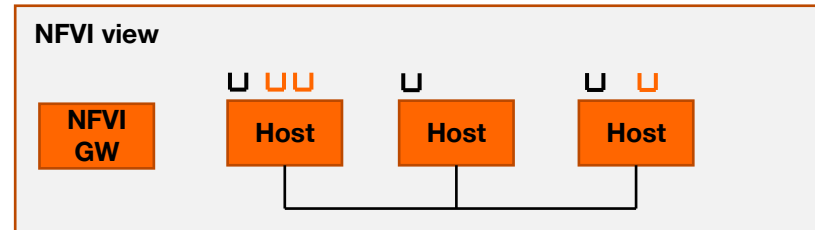
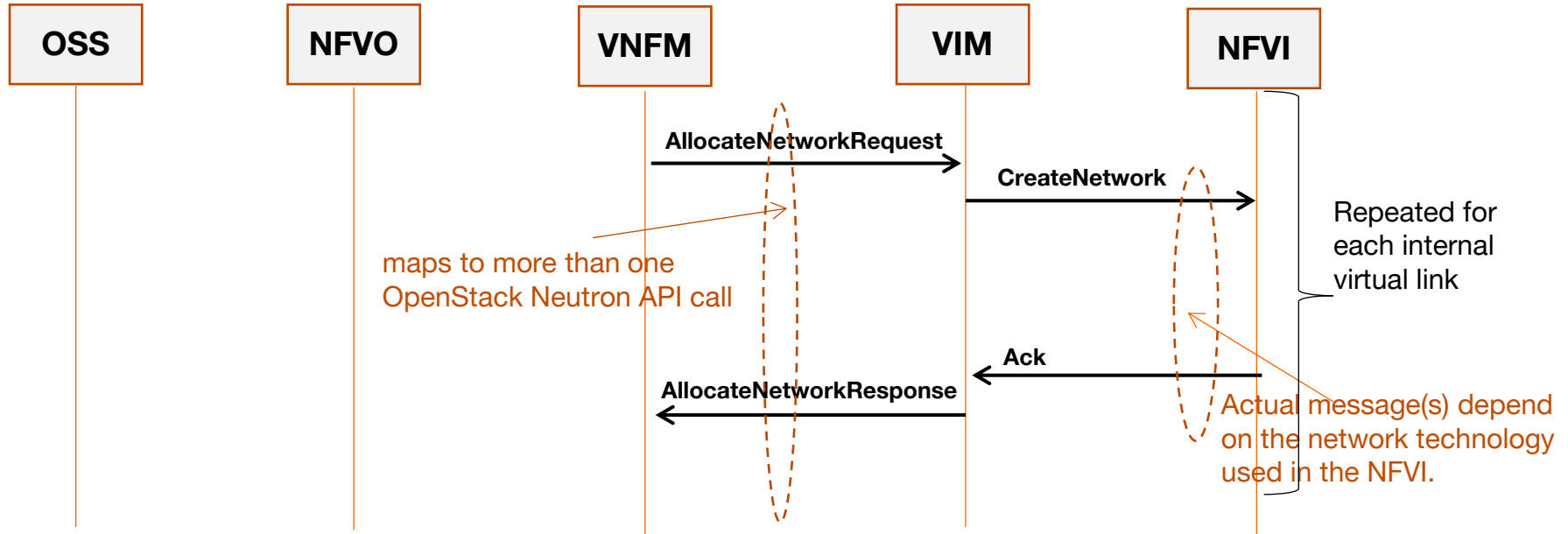
(Simplified) Flow diagram for NS instantiation



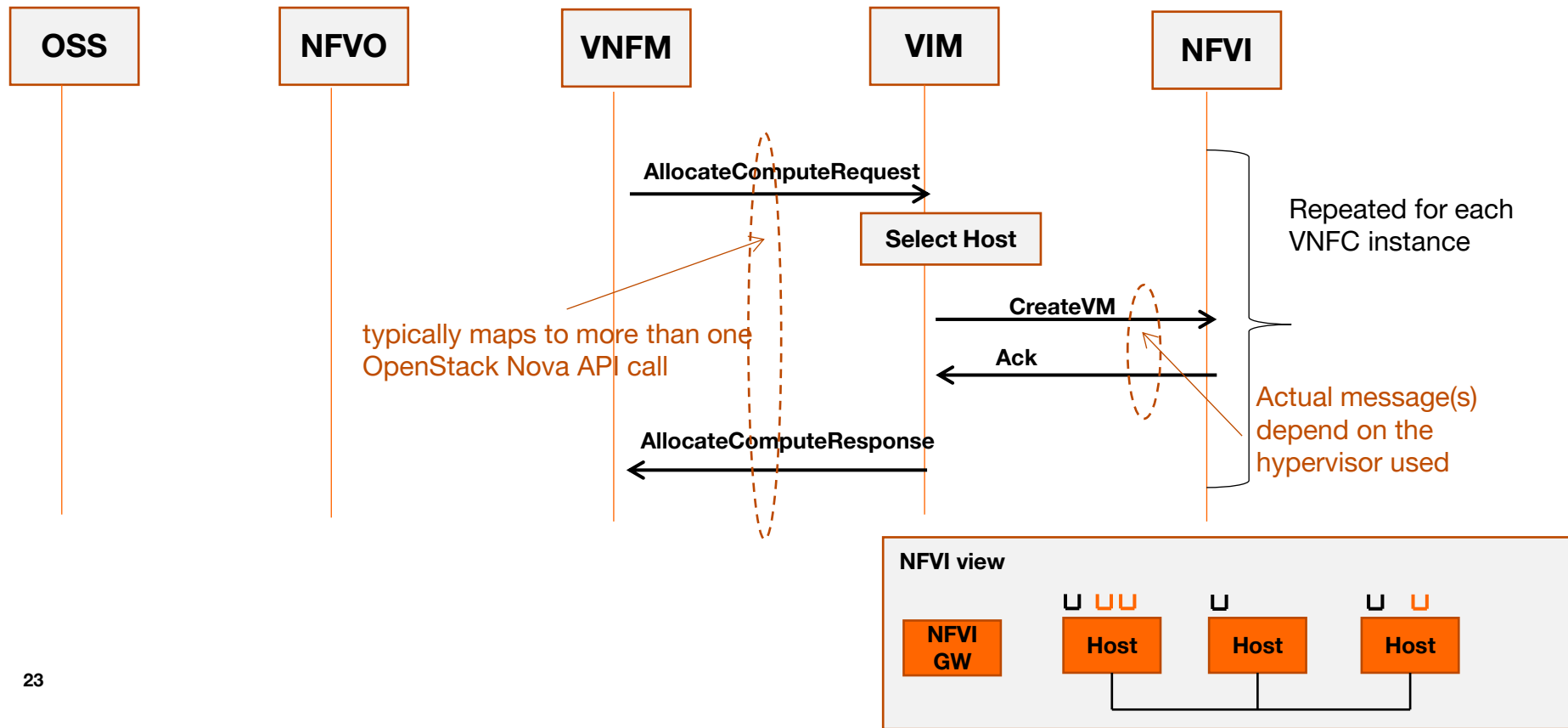
(Simplified) Flow diagram for VNF instantiation (1/3)



(Simplified) Flow diagram for VNF instantiation (2/3) (assuming an hypervisor-based solution)



(Simplified) Flow diagram for VNF instantiation (3/3) (assuming an hypervisor-based solution)



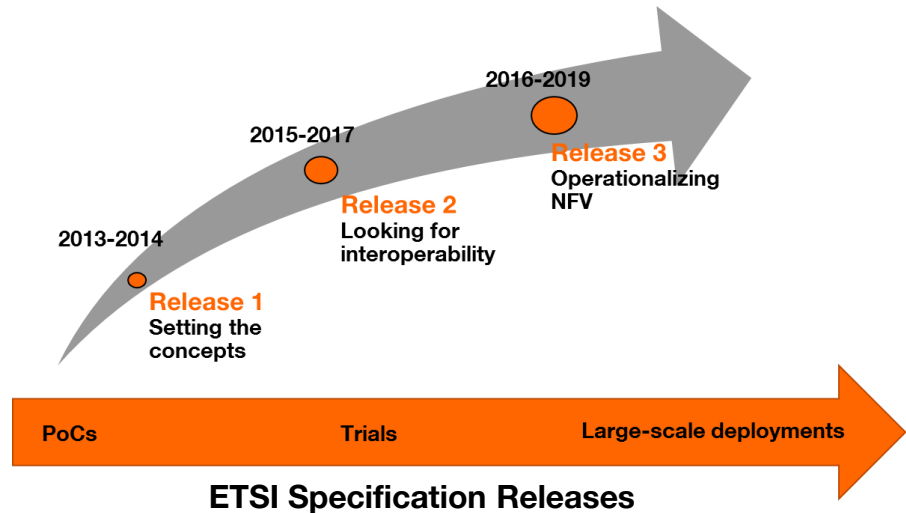
NFV Standardization and open source development

NFV standardization is driven by the NFV Industry Specification Group (ISG) in ETSI.

See: <http://www.etsi.org/nfv>

Several Open Source projects develop **software modules for NFV**, in particular

- ONAP (Linux Foundation)
- OSM (ETSI)
- Open Baton (Fraunhofer Fokus)
- OPNFV (Linux Foundation)



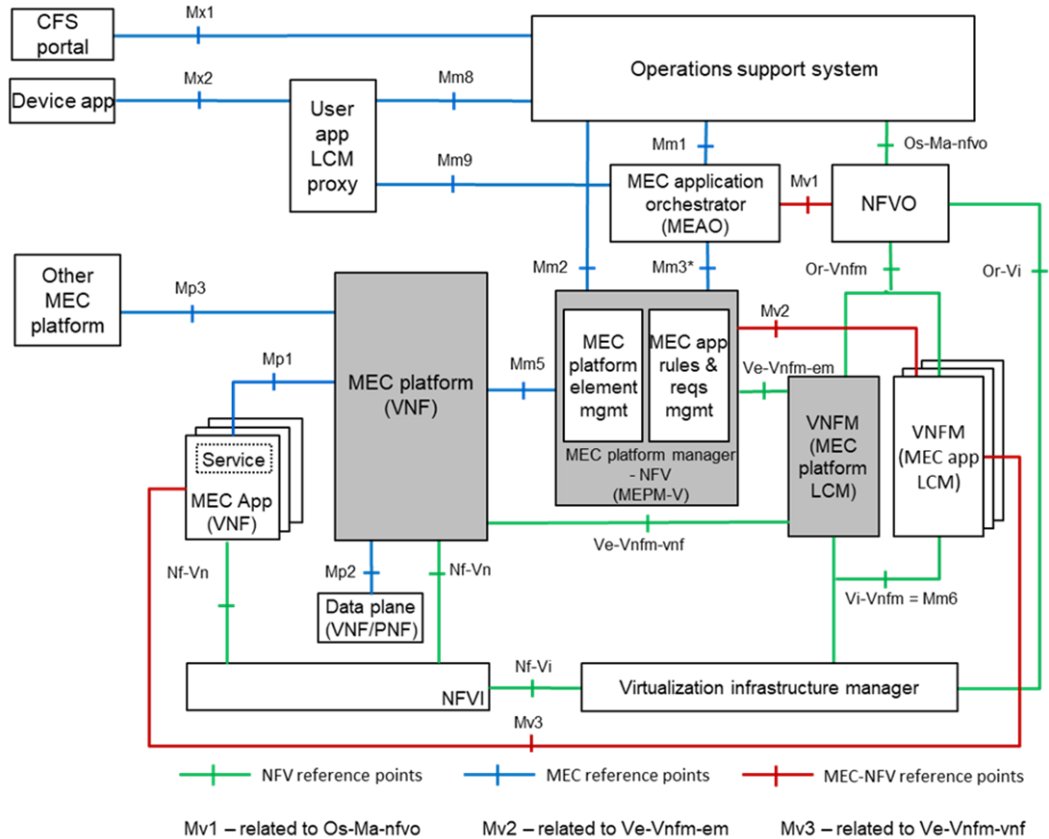
ETSI ISG MEC

MEC (Multi-services Edge Computing) is an ETSI Industry Specification Group (ISG)

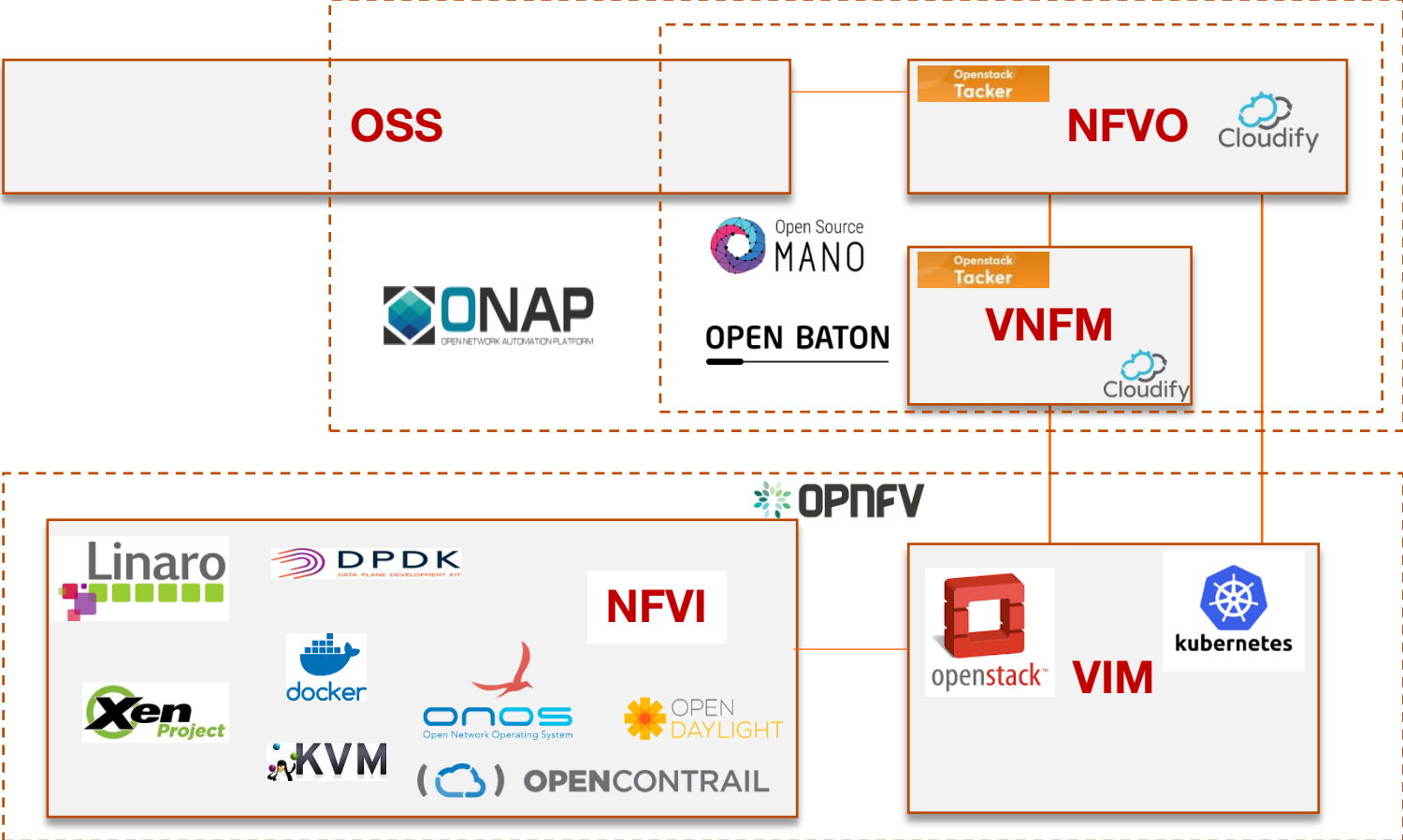
The MEC architecture leverages the NFV architecture.

VNFs are hosted in an Edge-Cloud (e.g. NFVI PoP located at Radio Base Station sites)

Two layers of orchestration, separating application (MEAO) vs resource orchestration (NFV-MANO).

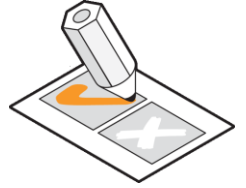


Open Source landscape for NFV (non-exhaustive list)



QUIZZ

Which of these statements are true?



1. **NFV requires that all network functions be deployed in large centralized data centres.**
2. **The main CPU in the physical servers hosting the VNFs is usually an ASIC.**
3. **The VNF software always run in virtual machines created by hypervisors.**
4. **A radio base station can be virtualised.**
5. **Physical Network Functions cannot be part of an NFV Network Service**
6. **A call server or a service platform can be implemented as a VNF.**
7. **In hypervisor-based solutions, a VNF instance is deployed in one and only one Virtual Machine.**

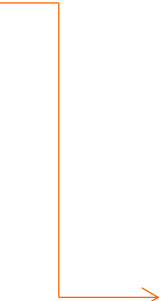
Agenda

Part I: Introduction, Architecture, Challenges

Part II: Focus on Management and Orchestration

Part III: NFV, SDN and Service Chaining

Part IV: NFV, Network Slicing and 5G



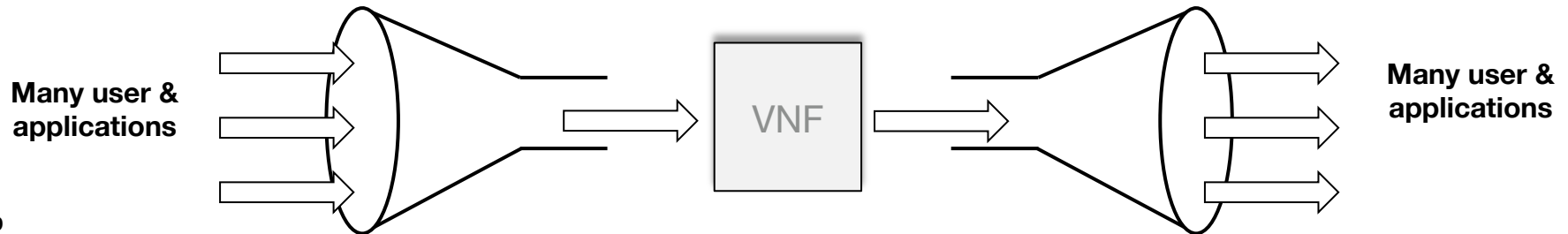
Introduction
Concepts and Architecture
Technical Challenges

NFV vs. conventional Cloud-based applications

1. **Carrier-Grade** requirements (high and predictable performance, high availability)
2. The need for **end-to-end management of network services**

Most conventional cloud applications are standalone endpoints while many network functions are intermediaries (a.k.a. middle-boxes).

Network functions involved in the same network service have to be managed in a coordinated way (e.g. adding resources to one network function can require adding resources to some others).



From challenges to solutions



There are security challenges as well...

The performance challenge

- How to achieve high and predictable performance (high throughput and low latency) “despite” the use of virtualisation and general-purpose processors?

The high availability challenge

- How to achieve high availability of network functions running in virtual machines deployed in distributed pools of commodity servers?



Software and Hardware **Acceleration** techniques and other optimizations

Network Functions and Network Services **designed to cope with infrastructure failures**

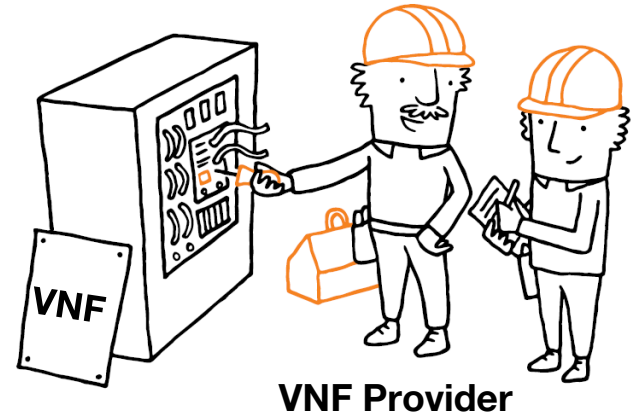
Software and Hardware Acceleration techniques and other optimizations... a tool box

VNF(C) instances **location tuning** (e.g. NFVI PoP selection, affinity rules, etc.)

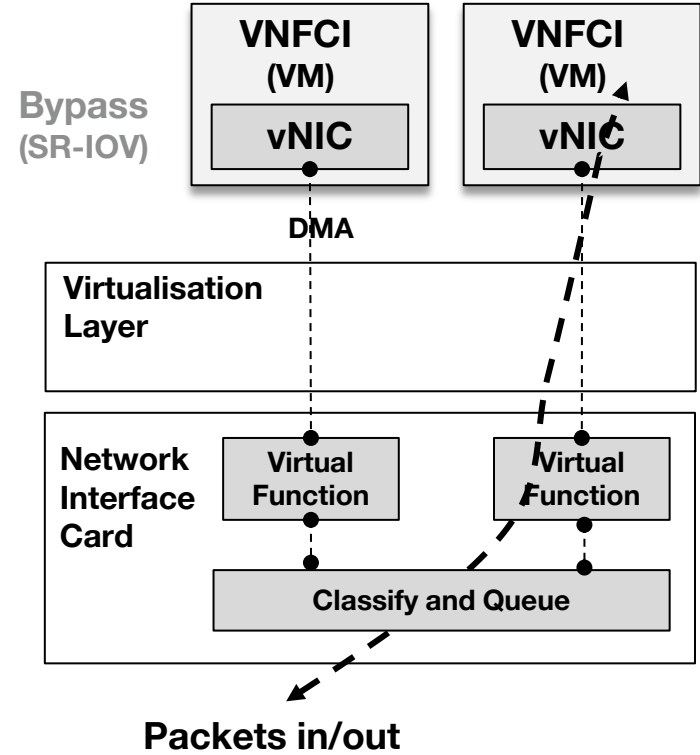
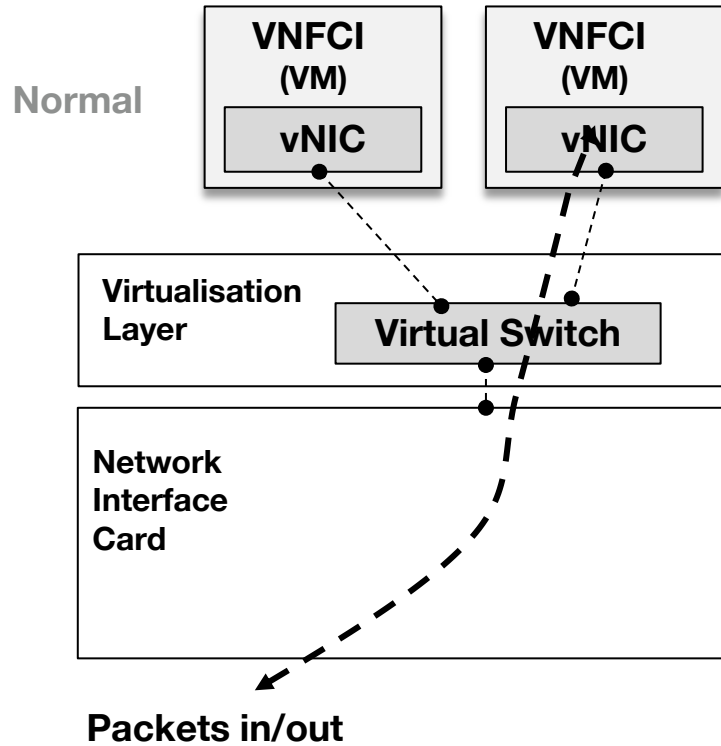
Virtualisation Layer bypass (e.g. SR-IOV)

Hardware acceleration (e.g. processing offloaded to a SmartNIC)

Software acceleration (e.g. data plane acceleration with DPDK, software tuning with CPU Pinning and NUMA, etc.)



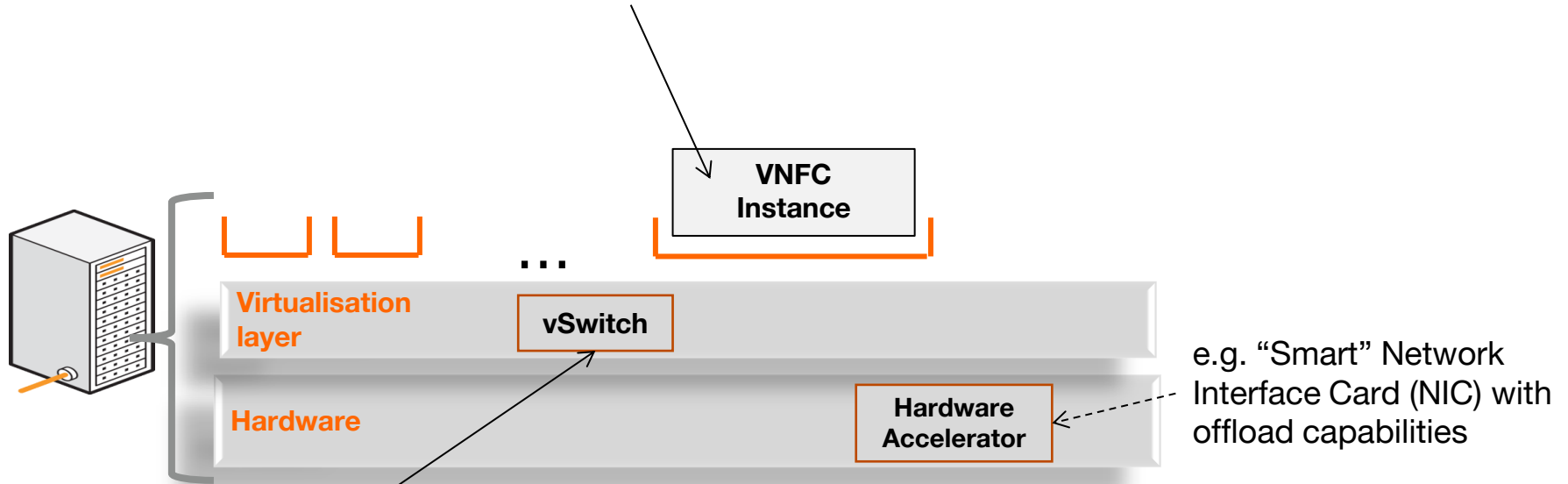
Virtualisation Layer bypass



Accelerating VNFs

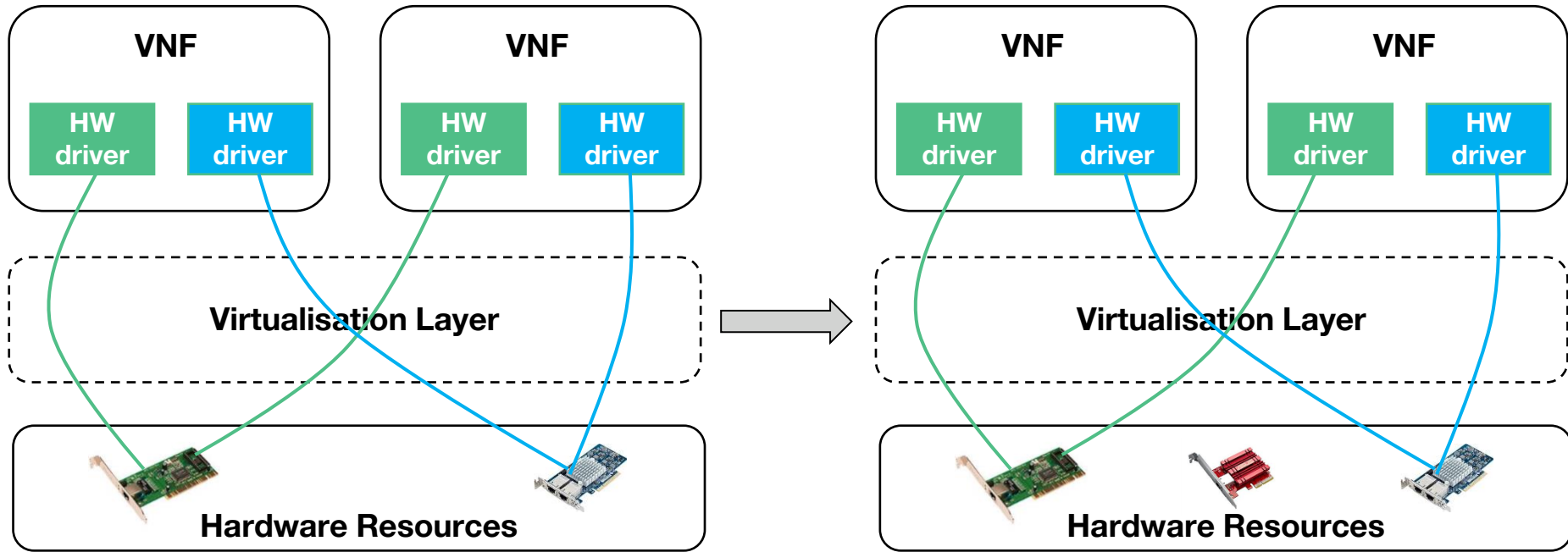
Accelerated VNF

Processing partly offloaded to the infrastructure (e.g. on a vSwitch or hardware accelerator) and/or Improved IP stack (e.g. DPDK-based IP stack)



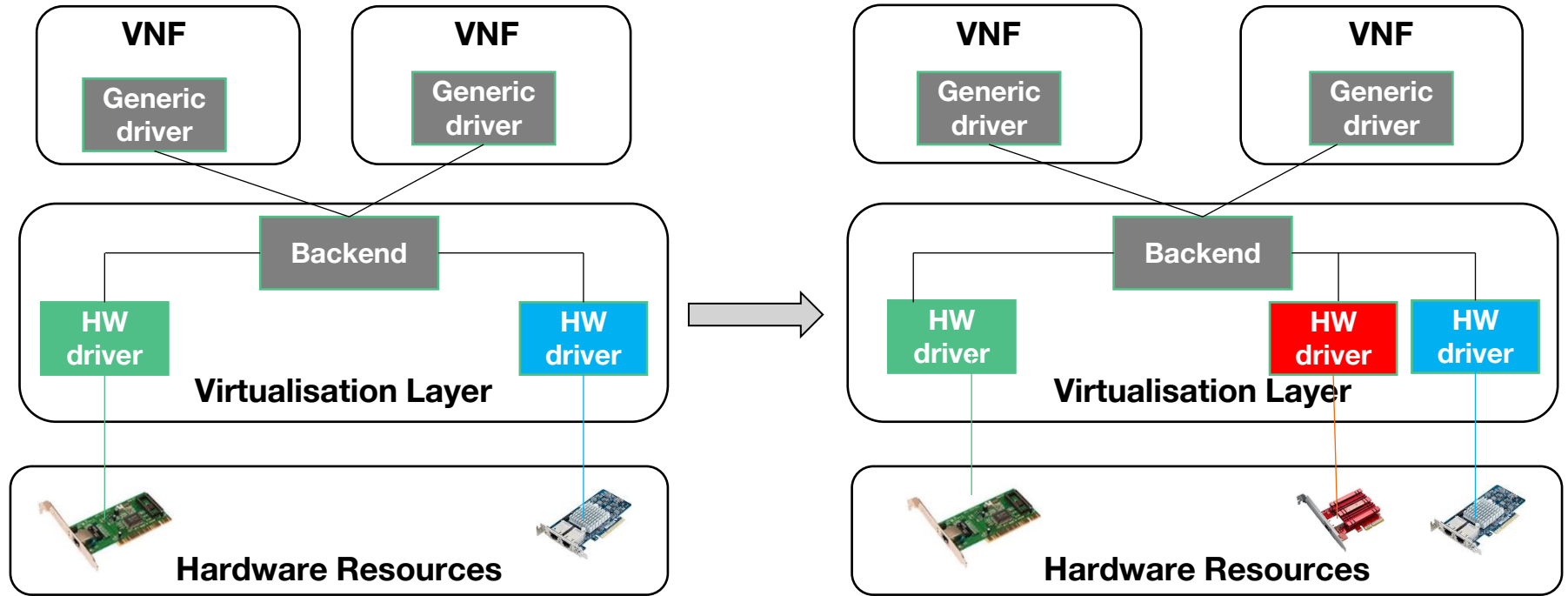
The vSwitch can be accelerated as well (e.g. DPDK-based)

Hardware Acceleration approaches: Pass-through



- Dependencies on hardware have to be specified in the VNFD.
- Restricted ability to move the VNF from one server to another.
- Assumes Single Root IO Virtualisation (SR-IOV) approach.

Hardware Acceleration approaches: Abstraction Layer



- One generic driver per type of acceleration function
- NFVI software accelerators can be incorporated as well
- Further details in ETSI GS NFV-IFA 002

Software design to cope with infrastructure failures

The goal: **Fast and easy failover**

Key design principles:

- ❑ Intra VNF redundancy
- ❑ Break down network functions in small, **atomic building blocks**, that can be quickly instantiated and migrated
- ❑ Make VNF components **dataless / stateless** wherever possible
- ❑ Make VNF components as independent as possible from each other (**loose-coupling**)

High availability in NFV

Redundancy / Fast-Failover

Intra-VNF (i.e. redundant VNFC instances)

Redundant VNF instances in a Network Service

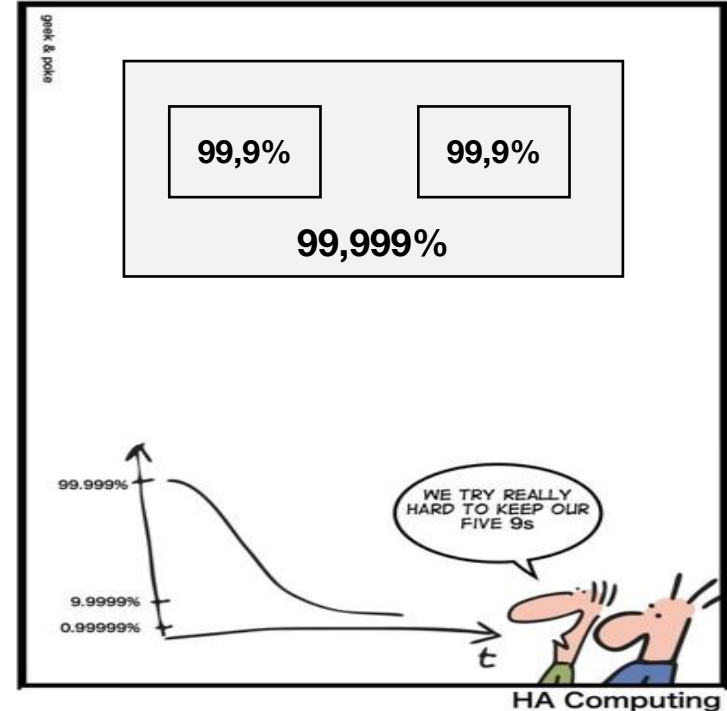
Load management

Load Balancing

Scaling-Out as a Congestion Avoidance solution

Seamless software upgrade

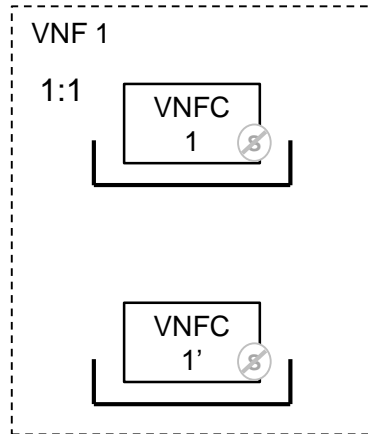
SIMPLY EXPLAINED



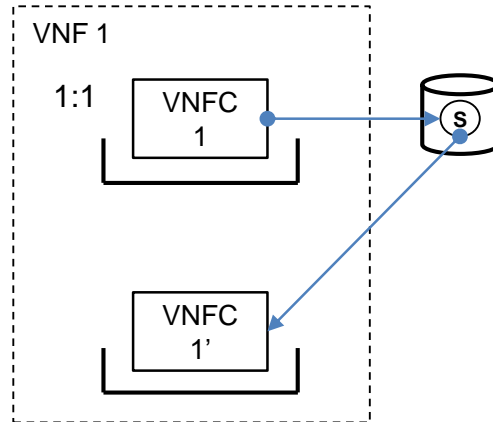
Intra VNF redundancy

Redundancy configurations: Active-Standby (1:1, n:1) vs. Active-Active (N+M) -> Not visible to the VNFM

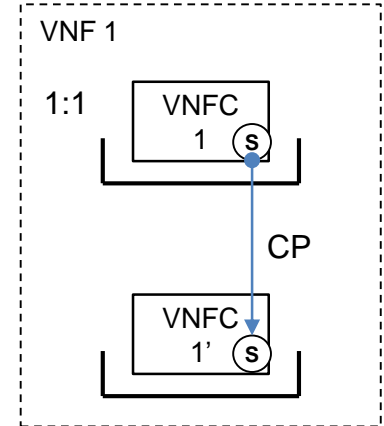
State management: Internal vs External -> The external state repository can in turn be a VNFC or a VNF or an NFVI service.



Stateless

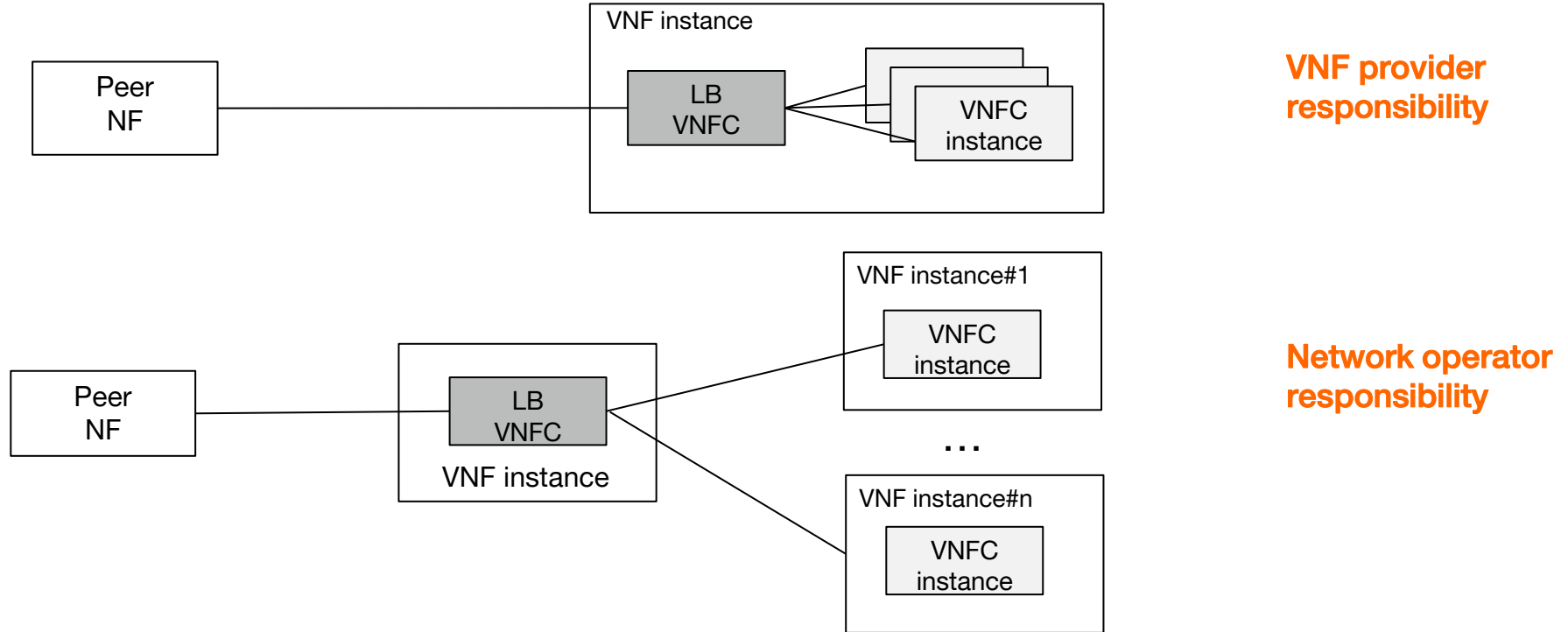


External state



State synchronization

VNF Load Balancing Models



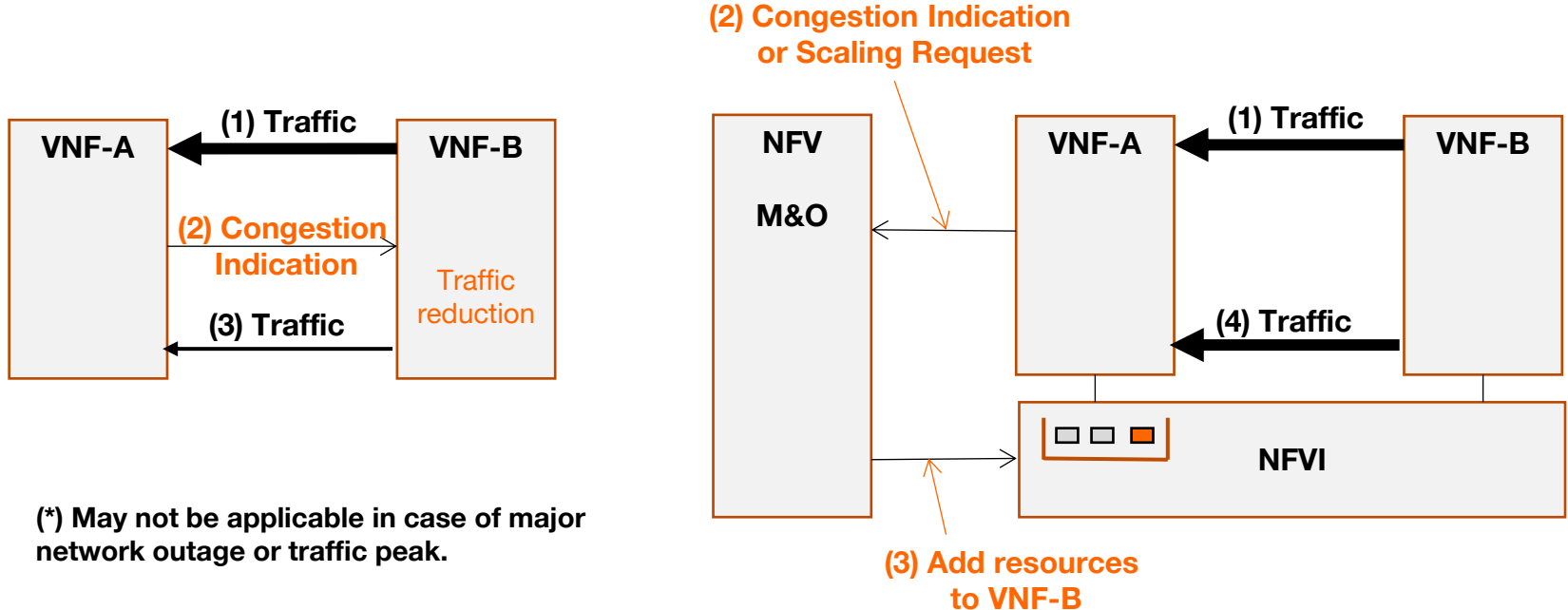
Load Balancing can also be provided by the NFV infrastructure (e.g. vSwitch), based on Layer 2 / Layer 3 criteria, e.g. under the control of a VNF or the OSS.

Scaling-Out as a Congestion Control solution



In conventional networks **congestion control** procedures are intended to **prevent overload**. They can be triggered proactively (predictable congestion) or reactively (when approaching congestion).

With NFV, **Scaling** procedures provide an alternative solution (*).



(*) May not be applicable in case of major network outage or traffic peak.

Software Upgrade

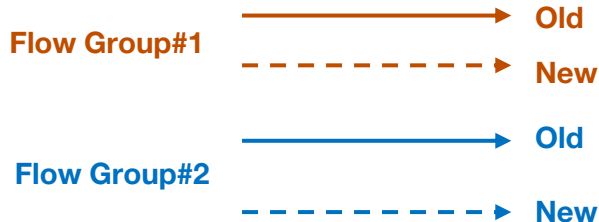
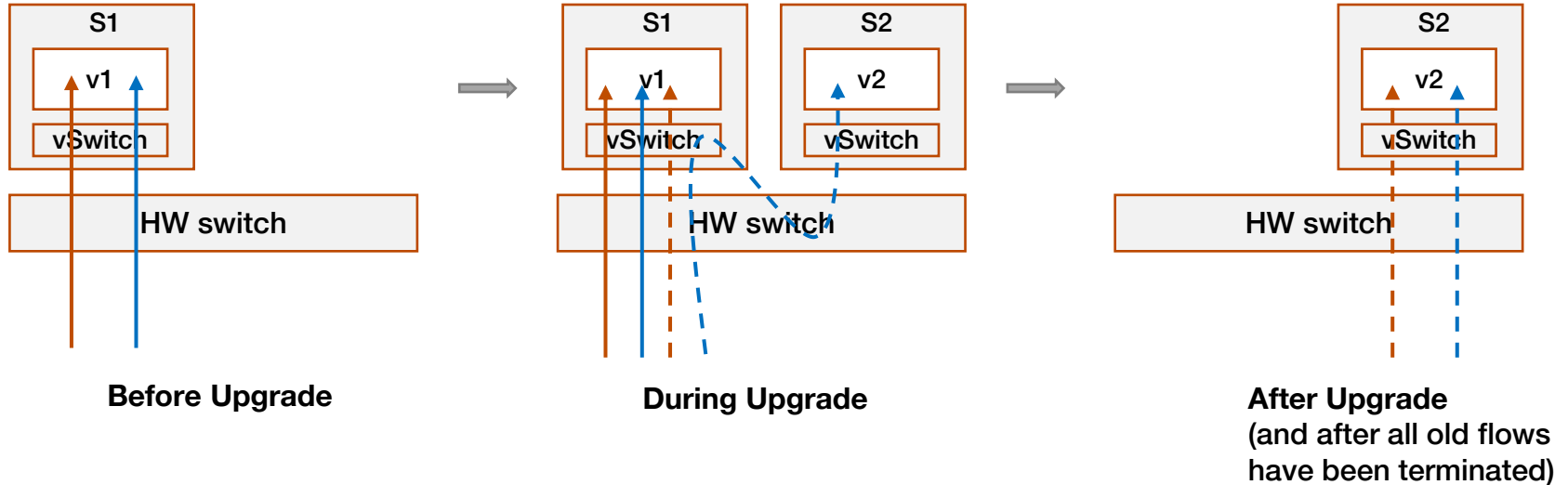
Service providers are looking for Software Update/Upgrade solutions such that **service availability and continuity is maintained**.

Software update in conventional networks typically implies reduced redundancy during upgrade process and/or upgrade can only take place during off-peak periods No failover possibility while switching over.

NFV provides an opportunity for a better approach: Software upgrade done in a **gradual and revertible way**

e.g. upgrade a fraction of the whole capacity, a certain service type or a certain user group, with the constraint of preserving the service availability and service continuity.

Scaling out with migration avoidance for software upgrade



Typically requires an SDN controller to direct an increasing % of the traffic to the new version.

Agenda

Part I: Introduction, Architecture, Challenges

Part II: Focus on Management and Orchestration

Part III: Focus on NFV Infrastructure

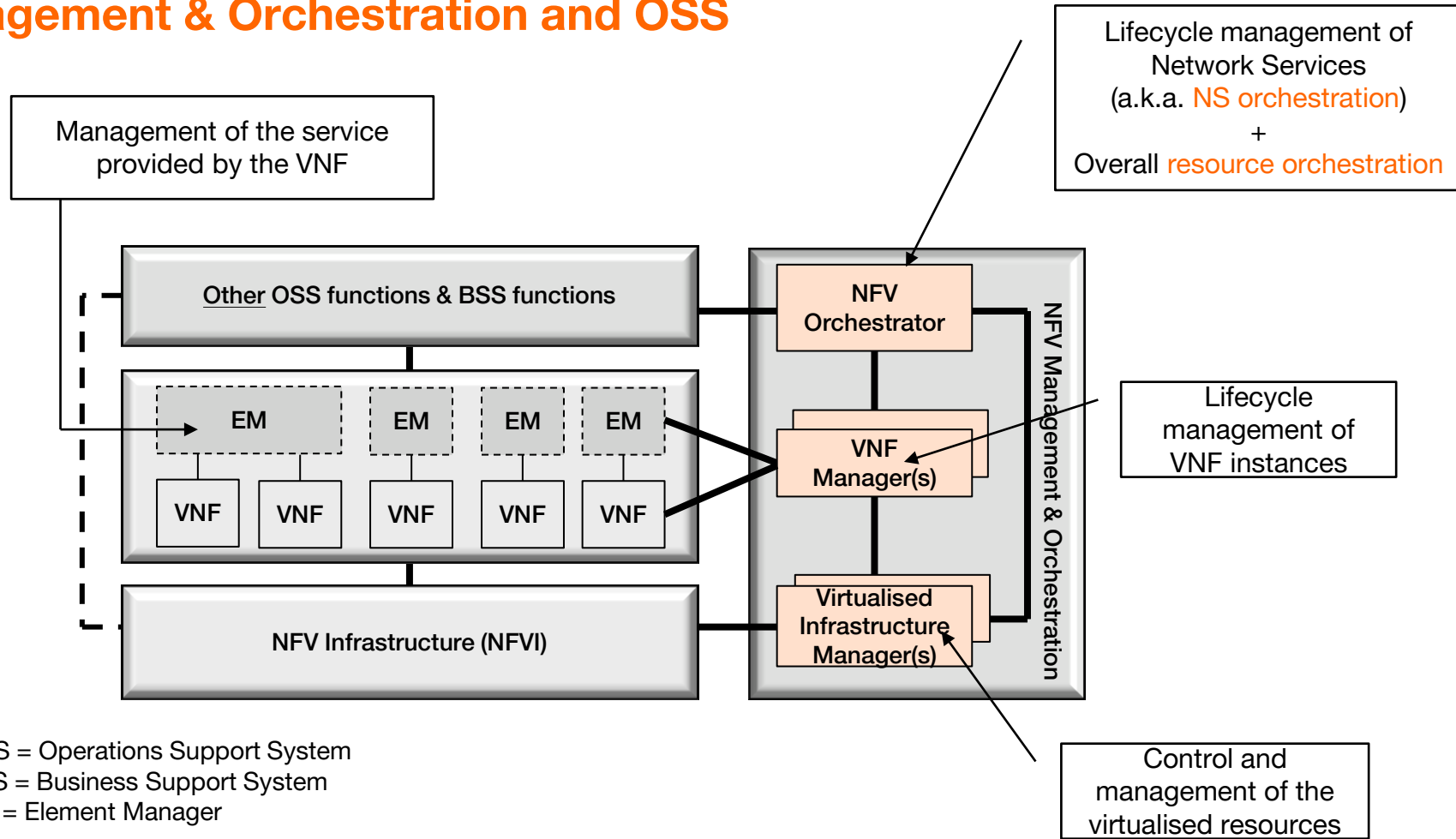
Part IV: NFV, SDN and Service Chaining

Part V: NFV, Network Slicing and 5G



General aspects
NFV Descriptors
Procedures
APIs

Management & Orchestration and OSS

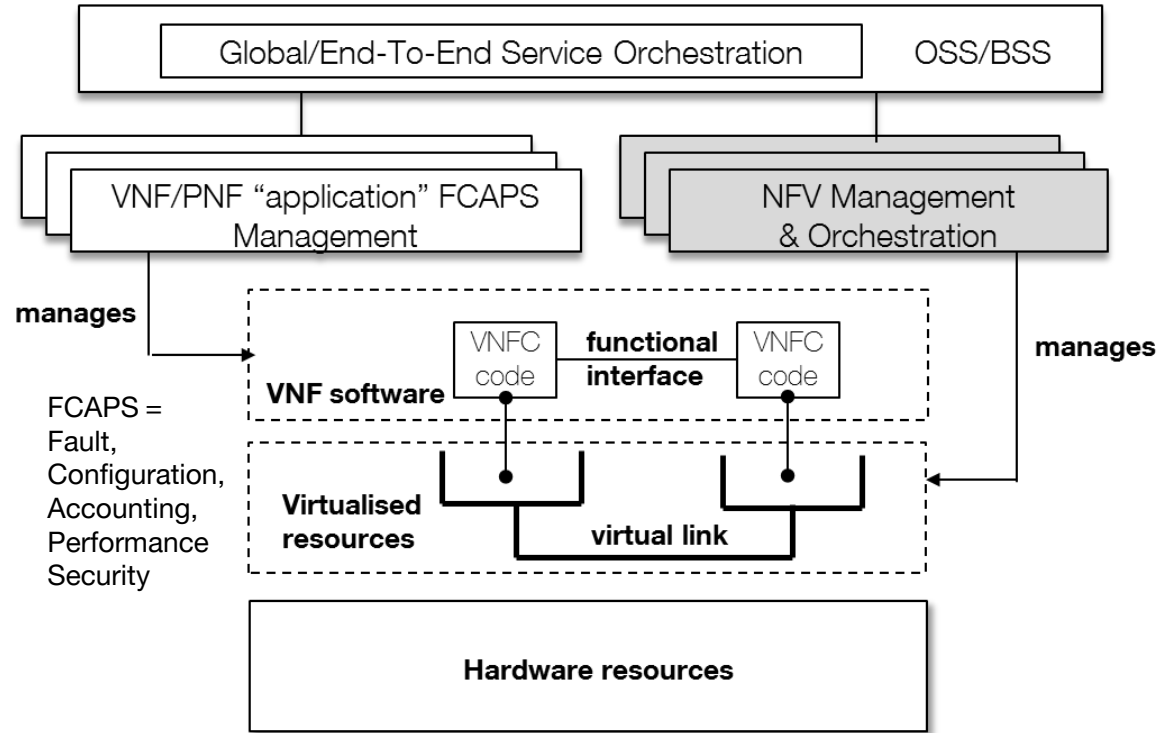


NFV Management & Orchestration is not Global Service Orchestration

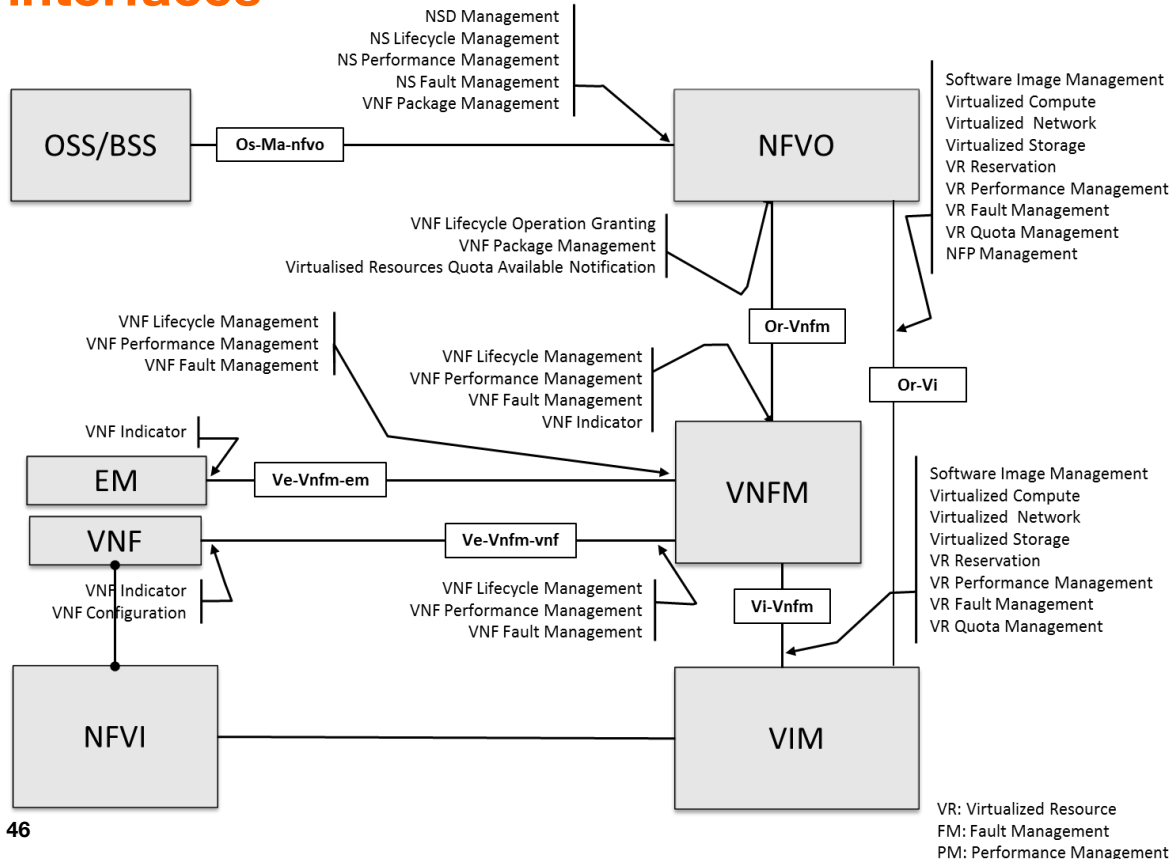
The NFV architectural framework assumes a **clear separation of concerns** between “application” and “execution platform” management.

NFV-MANO focuses on “platform” matters and is **agnostic to the type network function** being managed.

An **additional layer of orchestration** is required to enable zero-touch management of network services.



The full set of NFV Release 2 Management and Orchestration interfaces



For NFV Release 3, additional interfaces are being specified

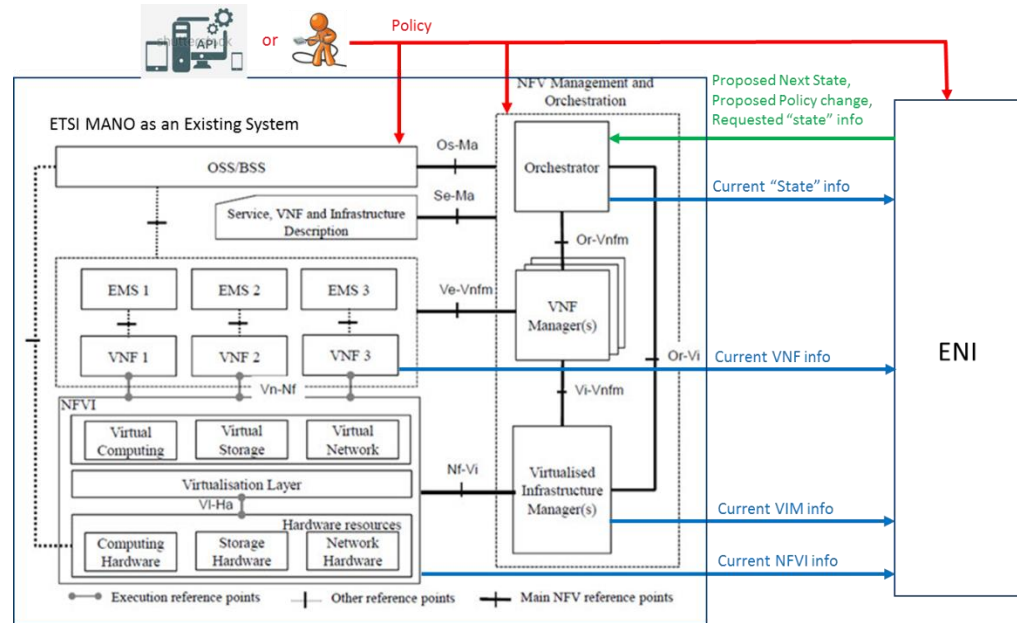
- ❑ **Inter-NFVO communication**
- ❑ **VNF Snapshot management**
- ❑ **Policy Management**

Artificial Intelligence for NFV-MANO

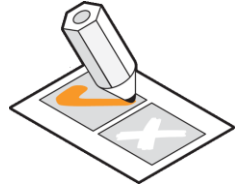
Experiential Networked Intelligence (ENI) is an ETSI Industry Specification Group

Defines a **Cognitive Network Management architecture**, using Artificial Intelligence (AI) techniques and context-aware policies to adjust offered services based on changes in user needs, environmental conditions and business goals.

The ENI engine is expected to **help NFV-MANO** functional blocks to make management decisions.



Which of these statements are true?



1. **Anti-affinity rules between instances of the same VNFC increases the VNF performance.**
2. **Scaling out a VNF is a way to prevent congestion.**
3. **A VNF external connection point is a specific internal connection point made visible to other VNFs.**
4. **A statefull network application can be implemented as a stateless VNF with external state.**
5. **A NIC supporting IPSec is an example of an hardware accelerator.**

Agenda

Part I: Introduction, Architecture, Challenges

Part II: Focus on Management and Orchestration

Part III: NFV, SDN and Service Chaining

Part IV: NFV and 5G



General aspects
NFV Descriptors
Procedures
APIs

NFV Descriptors

The VNF Descriptor (VNFD) and NS Descriptor (NSD) are **deployment templates** that contain information enabling the deployment of VNFs and NSs and the management of their lifecycle, **from a resource viewpoint**.

They do NOT contain **application-related data** and **instantiation data** (e.g. IP addresses)

A PNF Descriptor (PNFD) is not a genuine deployment template. The NFVO is not responsible for deploying PNFs!

However, it provides the NFVO information on the type of **connection points** exposed by a PNF, and thus on the type of Virtual Link it can be connected to.

TOSCA representation specified in ETSI GS NFV-SOL 001
YANG representation specified in ETSI GS NFV-SOL 006

Use of TOSCA for NFV

References: ETSI GS NFV-SOL 001

TOSCA Simple Profile in YAML used to represent NSDs, PNFs and VNFDs in a portable manner.

<http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.2/TOSCA-Simple-Profile-YAML-v1.2.html>

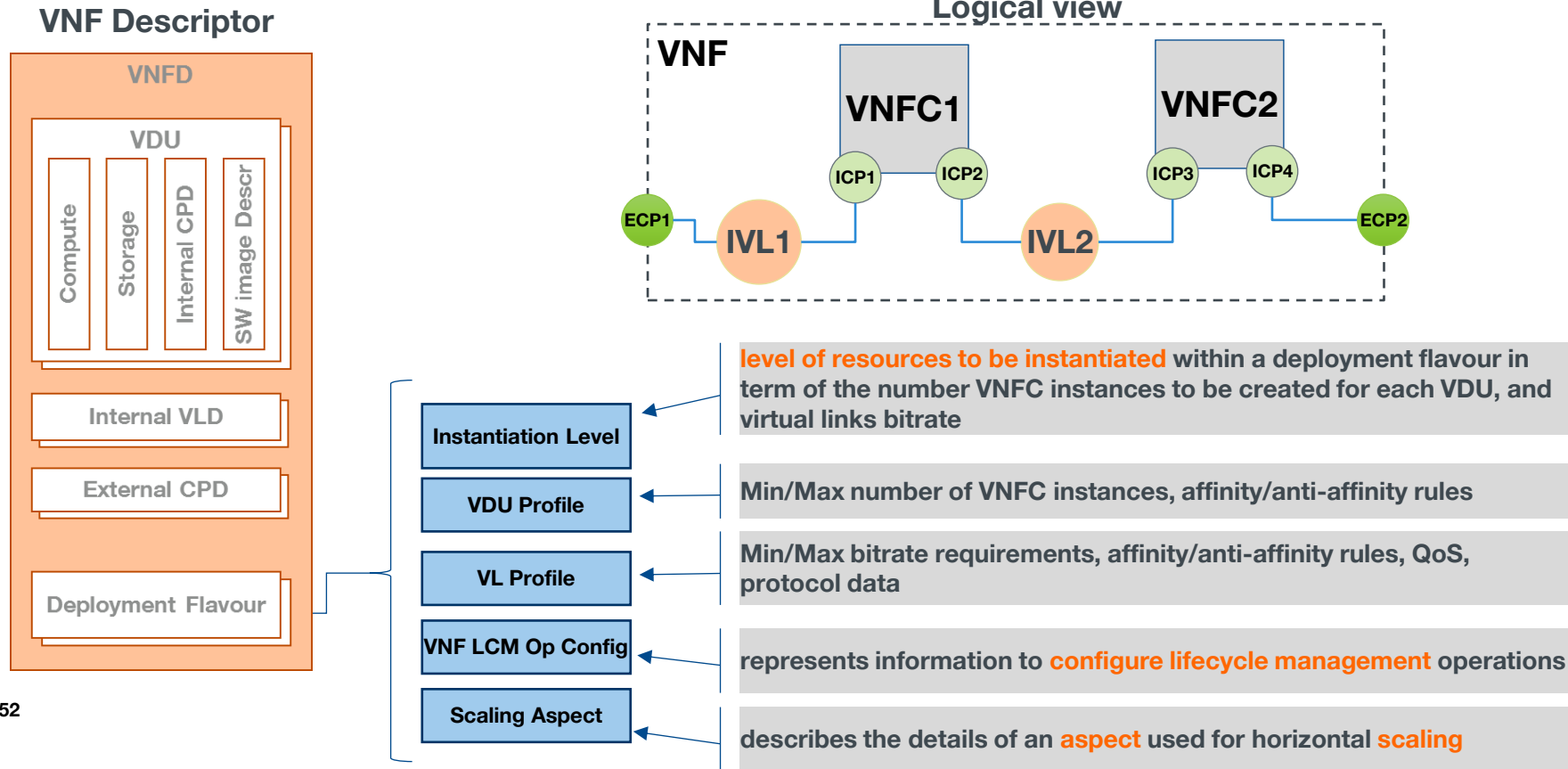
TOSCA = Topology and Orchestration Specification for Cloud Applications

VNFD, PNFs and NSDs are modelled as TOSCA service templates.

VNF Packages are structured according to the TOSCA Cloud Service Archives (CSAR) specification.

```
tosca.nodes.nfv.VnfVirtualLinkDesc:
  derived_from: tosca.nodes.Root
  properties:
    connectivity_type:
      type: tosca.datatypes.nfv.ConnectivityType
      required: true
    description:
      type: string
      required: false
    test_access:
      type: list
      entry_schema:
        type: string
        required: false
    vl_flavours:
      type: map
      entry_schema:
        type: tosca.datatypes.nfv.VlFlavour
        required: true
  capabilities:
    #monitoring_parameters:
      # modeled as ad hoc (named) capabilities in node template
    virtual_linkable:
      type: tosca.capabilities.nfv.VirtualLinkable
```

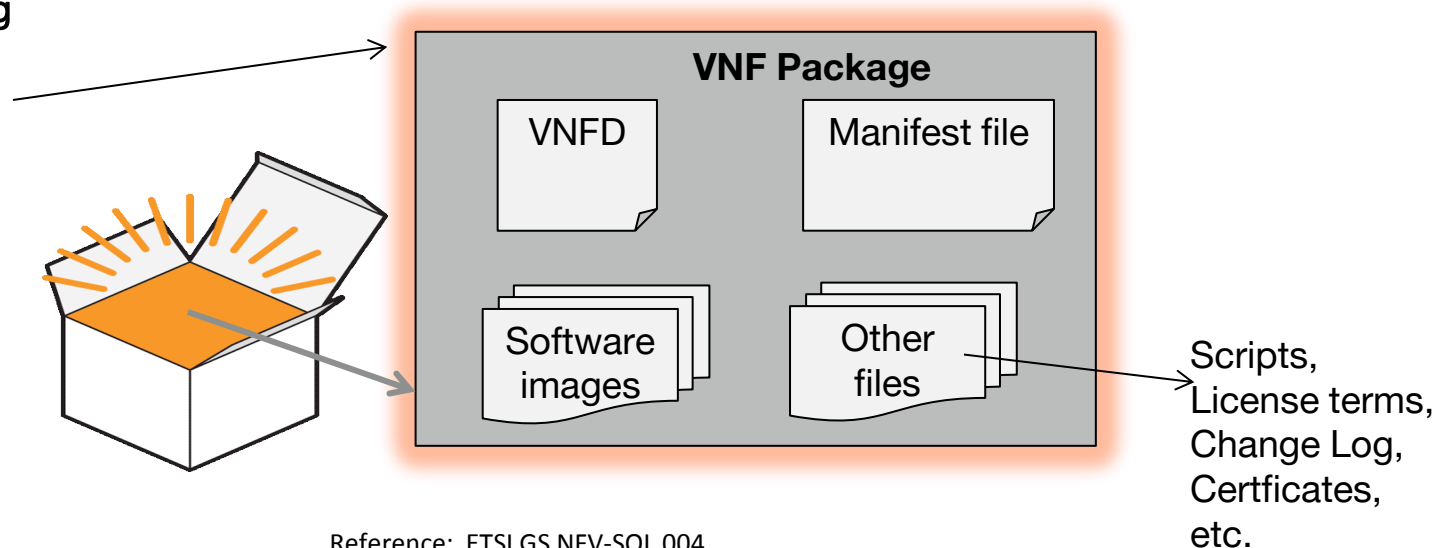
High-level view of a VNF Descriptor (VNFD) contents



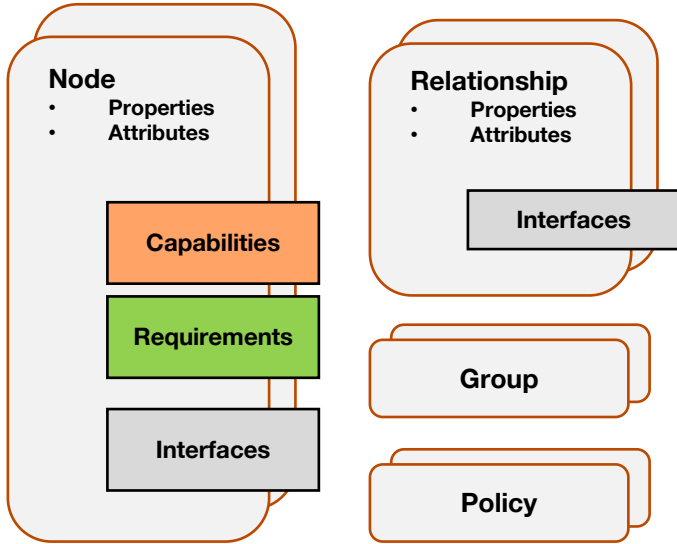
VNF Package

A **file archive** containing all artefacts required by the NFV Management & Orchestration functions to manage the lifecycle of the VNF. It is immutable (protected from modifications).

Format & Encoding
Based on the
TOSCA **Cloud
Service ARchive
(CSAR)**
specification



TOSCA Service Template



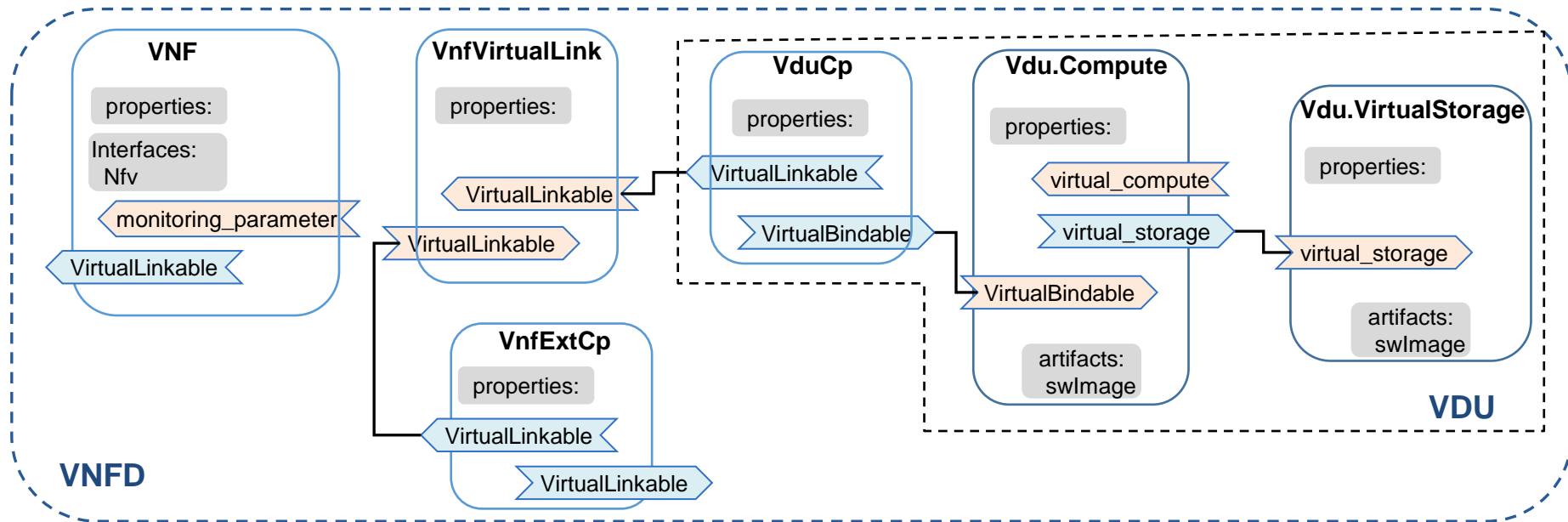
```
tosca_definitions_version: # Required TOSCA Definitions
version string
description: <template_type_description>
metadata: # Optional (and extensible) metadata keyname:
value pairs
```

```
# Some convenience keys left out for this example...
repositories: # list of external repository definitions
which host TOSCA artifacts
imports: # ordered list of import definitions
```

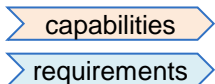
```
# Type definitions
artifact_types: # list of artifact type definitions
data_types: # list of datatype definitions
capability_types: # list of capability type definitions
interface_types: # list of interface type definitions
relationship_types: # list of relationship type
definitions
node_types: # list of node type definitions
group_types: # list of group type definitions
policy_types: # list of policy type definitions
```

```
topology_template:
# topology template definition of the cloud application
or service
```

VNFD service template overview



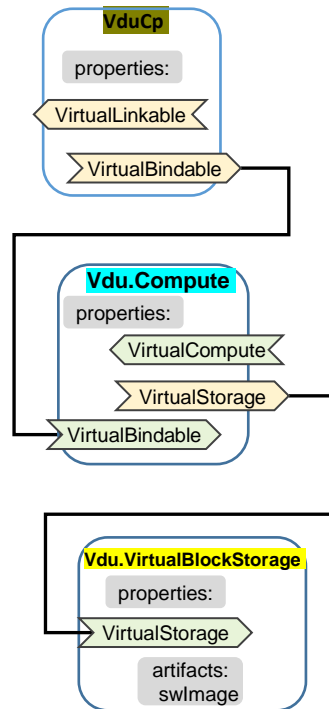
Symbol:



TOSCA VNFD: Compute, storage, connection points

```
...
topology_template:
...
node_templates:
  dbBackend:
    type: toska.nodes.nfv.Vdu.Compute
    properties:
      name: ..
      description: ..
      boot_order: ..
      nfvi_constraints: ..
      configurable_properties:
        additional_vnfc_configurable_properties: {}
      vdu_profile:
        min_number_of_instances: 1
        max_number_of_instances: 4
    capabilities:
      virtual_compute:
        properties:
          virtual_memory:
            virtual_mem_size: 8096 MB
          virtual_cpu:
            cpu_architecture: x86
            num_virtual_cpu: 2
            virtual_cpu_clock: 1800 MHz
    requirements:
      - virtual_storage: mariaDbStorage
```

```
  mariaDbStorage:
    type:
      toska.nodes.nfv.Vdu.VirtualBlockStorage
    properties:
      virtual_block_storage_data:
        size_of_storage: ..
        rdma_enabled: ..
      sw_image_data:
        name: Software of Maria Db
        version: 1.0
        checksum: 9af30fce37a4c5c831e095745744d6d2
        container_format: bare
        disk_format: qcow2
        min_disk: 2 GB
        min_ram: 8096 MB
        size: 2 GB
        operating_system: Linux
        supported_virtualisation_environments:
          - KVM
    artifacts:
      sw_image:
        type: toska.artifacts.nfv.SwImage
        file: maria.db.image.v1.0.qcow2
  dbBackendInternalCp:
    type: toska.nodes.nfv.VduCp
    properties:
      layer_protocols: [ ipv4 ]
      role: leaf
      description: Internal connection point on an VL
      protocol_data: [ associated_layer_protocol: ipv4 ]
      trunk_mode: false
    requirements:
      - virtual_binding: dbBackend
      - virtual_link: internalV1
```



NFV descriptors in YANG

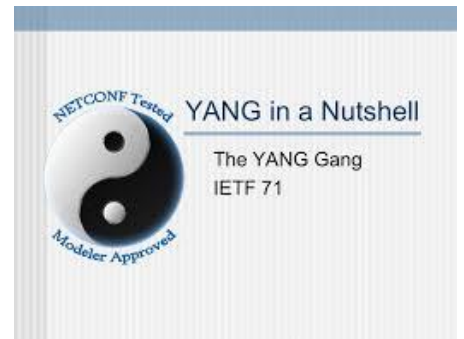
An alternative to TOSCA, specified in IETF RFC 6020 and 7950.

A language rather **intended to model configuration data**, in relation to NETCONF/RESTCONF protocols ... but nothing prevent using it for other purposes.

Use by Open Source Mano (OSM) orchestrator.

ETSI specification: ETSI GS NFV-SOL 006 (work started in September 2017)

Many YANG modules available here: <https://github.com/YangModels/yang>



Agenda

Part I: Introduction, Architecture, Challenges

Part II: Focus on Management and Orchestration

Part III: NFV, SDN and Service Chaining

Part IV: NFV and 5G



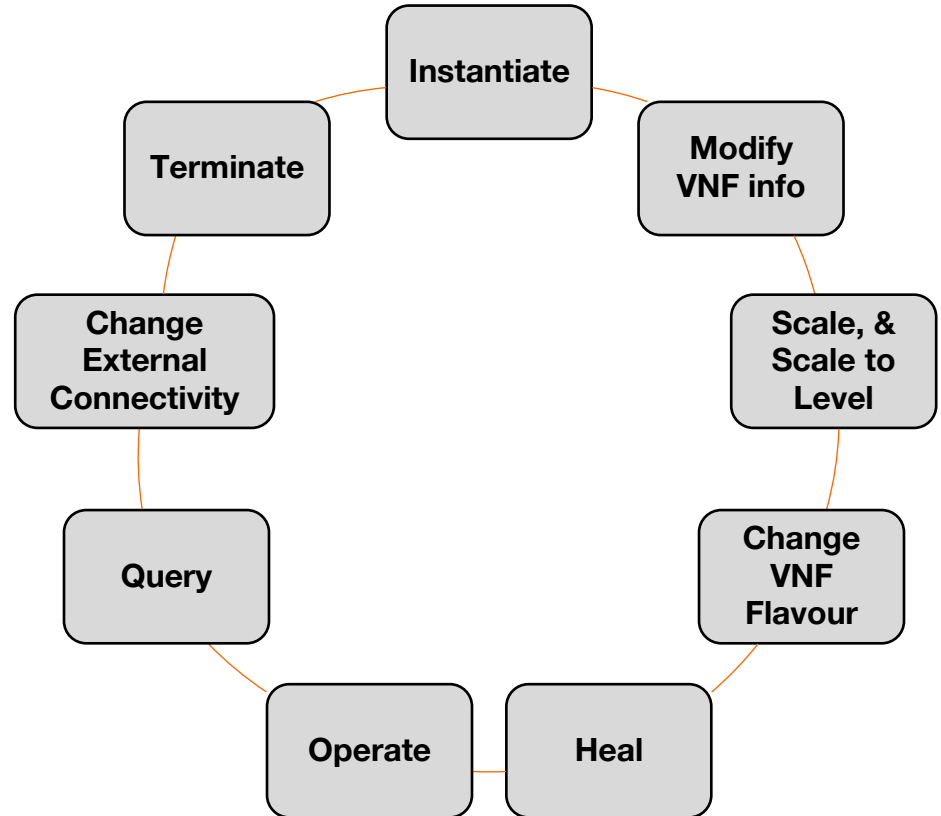
General aspects
NFV Descriptors
Procedures
APIs

VNF Lifecycle Management overview

VNF lifecycle management operations can influence the **allocation of virtualised resources** to a VNF instance, and/or **modify the state** of the VNF instance.

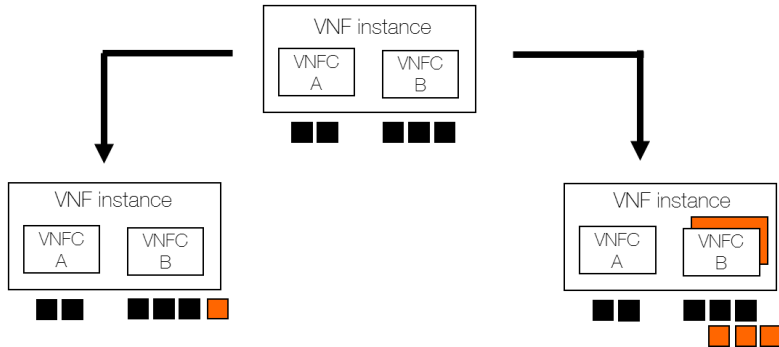
These operations are executed by the **VNFM** upon request of the **NFVO**, the **EM** and the **VNF itself** (with some exceptions).

The VNFM can perform **automating scaling** and **automatic healing** as well (based on information in the VNFD)



VNF Instance scaling, simply explained...

- Scale up / down
 - Increasing the resources allocated to VNFC instance(s) in the VNF instance
- Scale out / in
 - Adding / removing VNF instance(s) to a VNF instance.



Most VNFs only support horizontal scaling

- ❑ **Scaling out/in = Horizontal scaling**
- ❑ **Scaling up/down = Vertical scaling**

Adding/Removing VNF instances is another way to increase/decrease the overall VNF processing capacity.

VNF scaling triggering modes

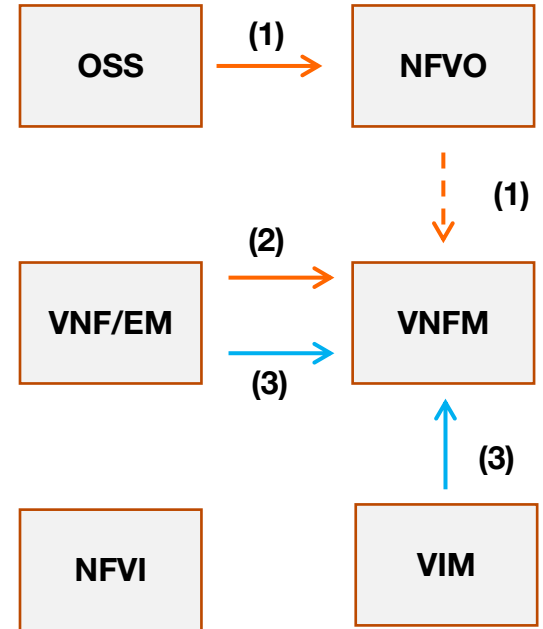
On-Management request: Explicit request (1) from the OSS to the NFVO

On-demand: Explicit request (2) sent to the VNFM by the VNF or its EM

Automatic: Event-based triggering (3) in the VNFM, based on **auto-scaling rules** and LCM scripts available in the VNFD

Examples of events include

- Resource-related events such as NFVI performance threshold crossing notifications (e.g. % CPU utilisation)
- Application-specific events (a.k.a. VNF indicators)



Agenda

Part I: Introduction, Architecture, Challenges

Part II: Focus on Management and Orchestration

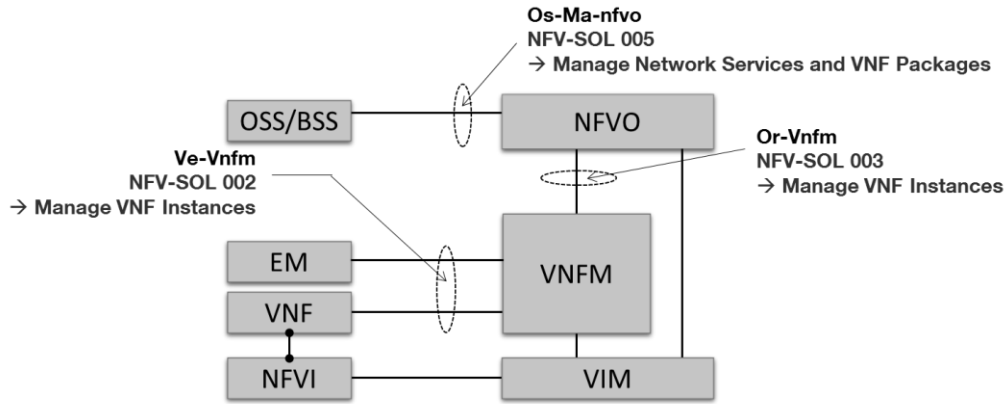
Part III: NFV, SDN and Service Chaining

Part IV: NFV and 5G



**General aspects
NFV Descriptors
Procedures
APIs**

APIs for Management and Orchestration



Use of **RESTful APIs** specified by ETSI except for the VIM northbound interfaces where use of **OpenStack APIs** is assumed.

ETSI APIs specifications are available in dual form

- Text and Tables
- OpenAPI format (a.k.a. Swagger)

- The ETSI architectural framework identifies a number of **reference points**, on which several **interfaces** are produced.
- **1 interface = 1 or 2 APIs**

REST (Representational State Transfer) design applied to ETSI NFV

HTTP-based incarnation of REST

JSON used as the format for resource representations

Manipulation of resources (e.g. vnf instances) using CRUD(*) operations

POST – create resource

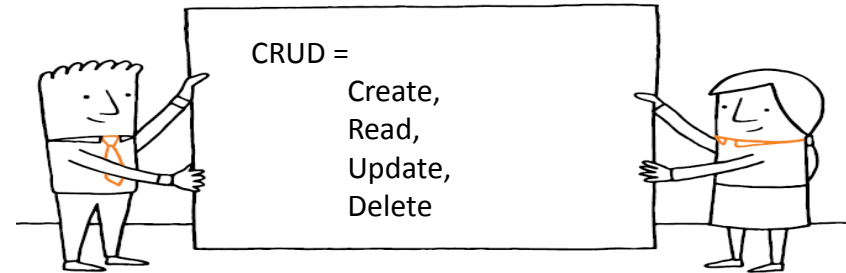
GET – read resource / query resources

PATCH/PUT – update resource

DELETE – delete resource

Special resources for

- notification management (notification endpoint)
- lifecycle operation occurrences
- complex procedures (task resources)
- error handling (task resources)



TASK resources for ETSI NFV APIs

Some procedures (e.g. VNF lifecycle management) are not a good fit to be modelled using CRUD operations.

TASK resources provide a workaround, an **RPC-like solution** within a REST framework.

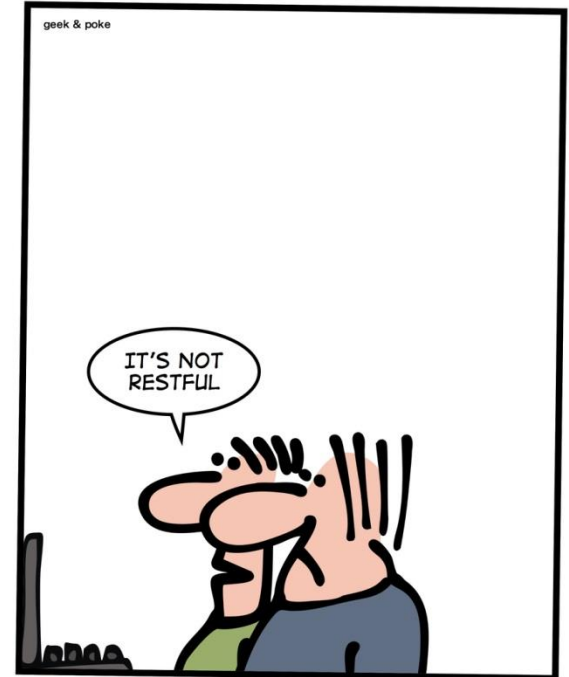
A **TASK resource** is a child of a resource which represents the task/operation to be executed.

E.g. `vnf_instances/{vnfInstanceId}/scale_vnf`

The client **POSTs** a set of parameters to the **TASK resource** to execute the associated operation.

The operation affects the state of the parent resource.

HOW TO INSULT A DEVELOPER



OpenAPI Specification for ETSI NFV APIs

OpenAPI = a language for describing RESTful APIs, previously known as “Swagger”

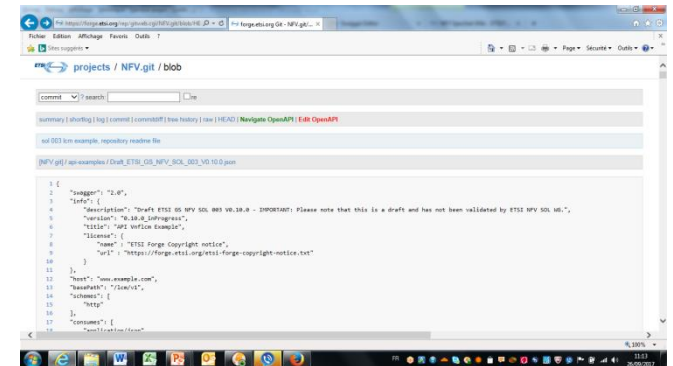
See: <https://www.openapis.org/>

... also a set of tools, to edit, navigate and validate the specifications

See, e.g. <https://forge.etsi.org/swagger/editor/>

OpenAPI files for NFV are available here:

https://nfvwiki.etsi.org/index.php?title=API_specifications#OpenAPIs



```
1 {
2   "swagger": "2.0",
3   "info": {
4     "description": "Draft ETSI NFV SOI M2V V0.10.0 - IMPORTANT! Please note that this is a draft and has not been validated by ETSI NFV SOI M2V.",
5     "version": "0.10.0",
6     "title": "NFV SOI Example",
7     "license": {
8       "name": "ETSI Forge Copyright notice",
9       "url": "https://forge.etsi.org/etsi-forge-copyright-notice.txt"
10    }
11  },
12  "host": "www.example.com",
13  "basePath": "/fims/v1",
14  "schemes": [
15    "https"
16  ],
17  "consumes": [
18    "application/json"
19  ]
20 }
```

Flow of VNF LCM operations

Operations:

Instantiate VNF

Scale VNF

Scale VNF to Level

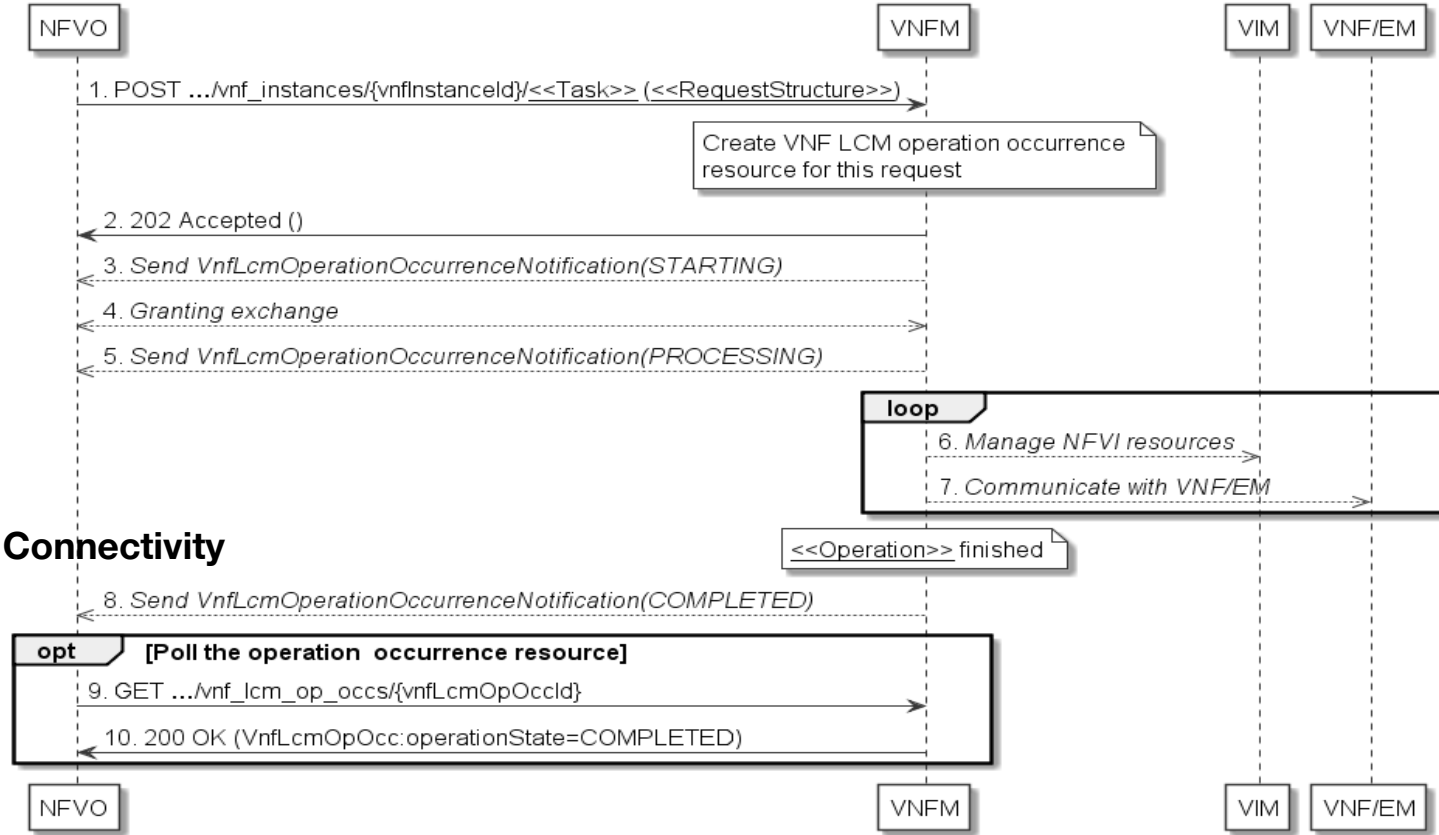
Change VNF Flavour

Operate VNF

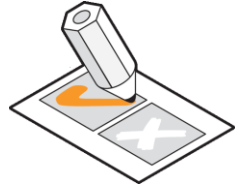
Heal VNF

Change External VNF Connectivity

Terminate VNF



Which of these statements are true?



1. **The Granting procedure enables the NFVO to provide the VNFM with information about the VIM through which virtualised resources are to be allocated**
2. **The NFVO selects physical servers where to deploy VNF instances**
3. **A scaling aspect corresponds to one type of VNFC**
4. **Fault notifications related to virtualised resources can be forwarded to a VNF instance by the VNFM.**
5. **The VNFM and the NFVO communicate with each other using SNMP**

Agenda

Part I: Introduction, Architecture, Challenges

Part II: Focus on Management and Orchestration

Part III: NFV, SDN and Service Chaining

Part IV: NFV and 5G

SDN & NFV

A “new” approach to **implementing and deploying** network functions and services by relocating them from dedicated appliances to pools of generic industry servers

Network Functions Virtualisation

Target: Any type of functional or physical entity within Telco’s networks (e.g. call servers, gateways, firewalls, CDN cache servers, etc.)

Software Defined Networking

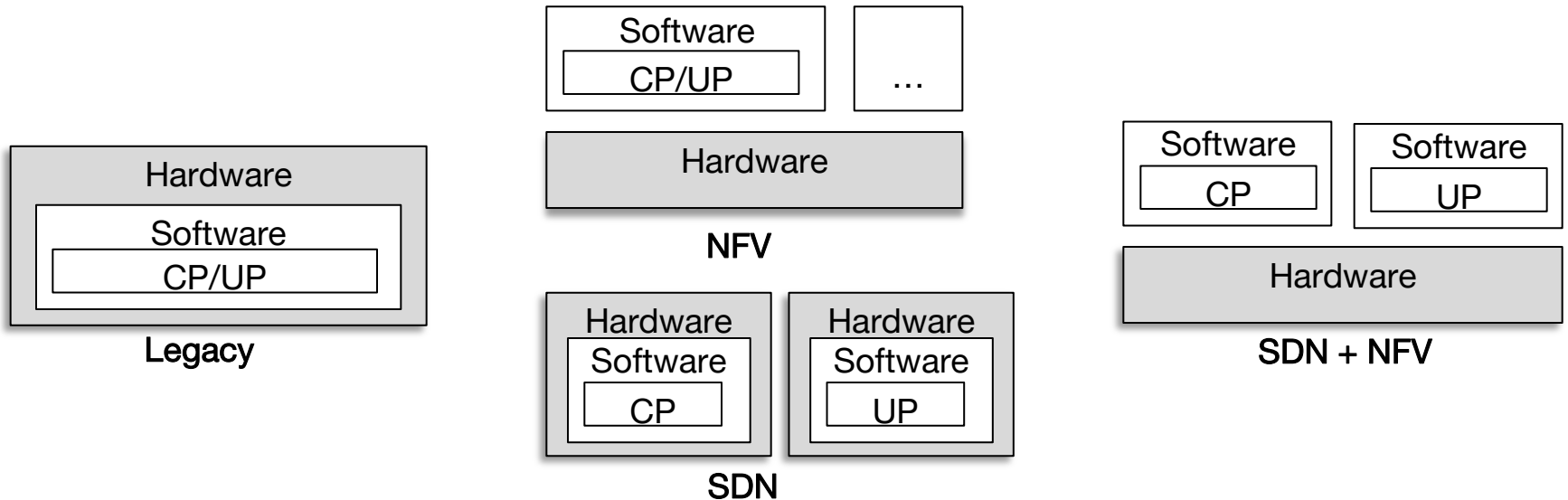
A “new” approach to **programmable networks** that leverages the **separation of control and forwarding planes** in particular

Target: Transport-related functions within Telco’s networks (e.g. routing & forwarding, quality of service management, firewalling etc.)

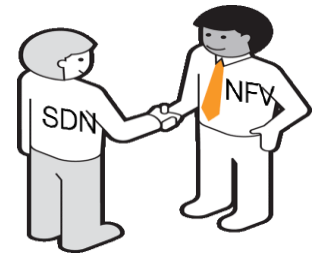
SDN and NFV are orthogonal to each other

SDN decouples (and centralizes) the **control plane** from the **user/data/forwarding plane**.

NFV decouples the **software** from the **hardware**.



NFV & SDN hand-in-hand



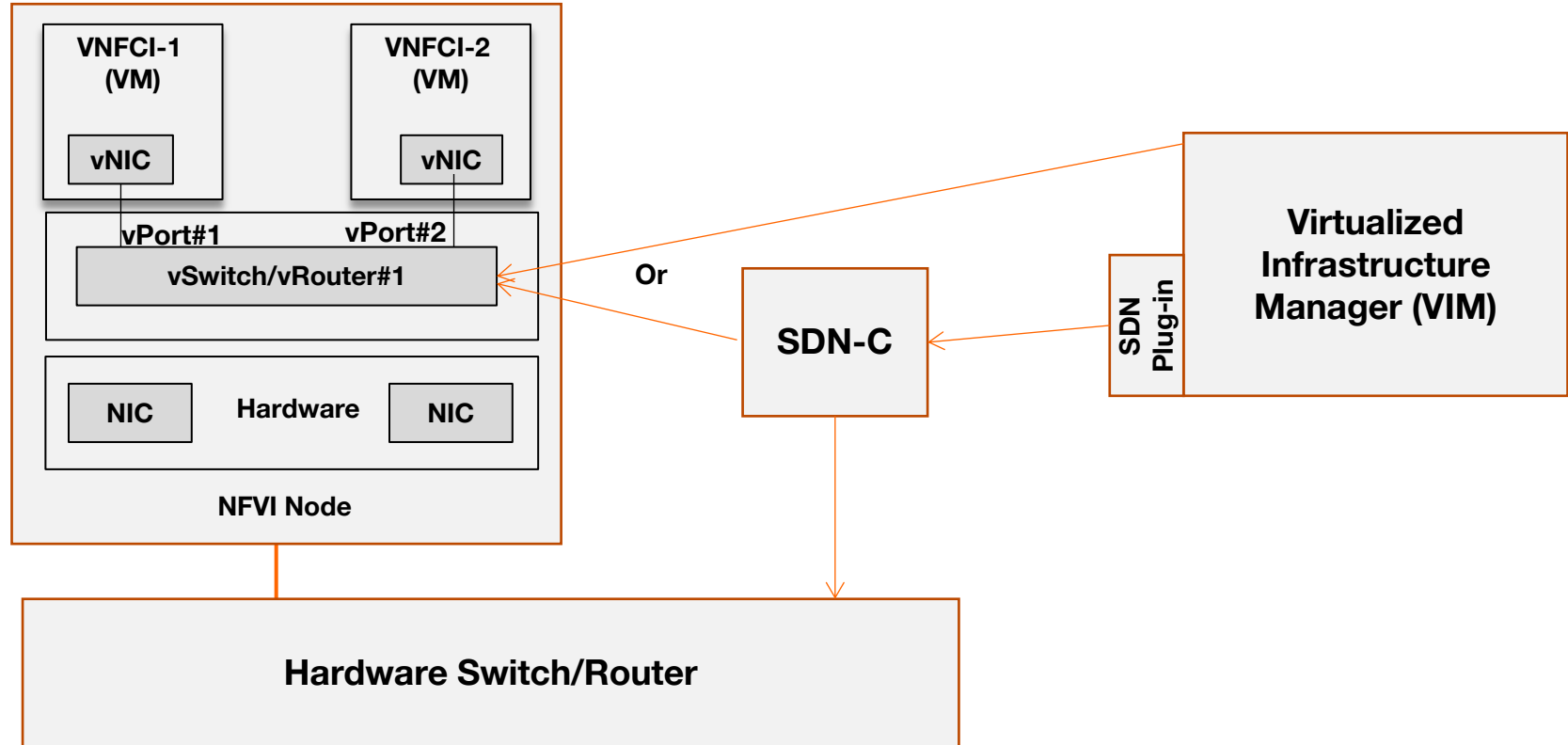
Combining SDN and NFV to enable automation of connectivity services management

Implementing **SDN** using the **NFV** technology

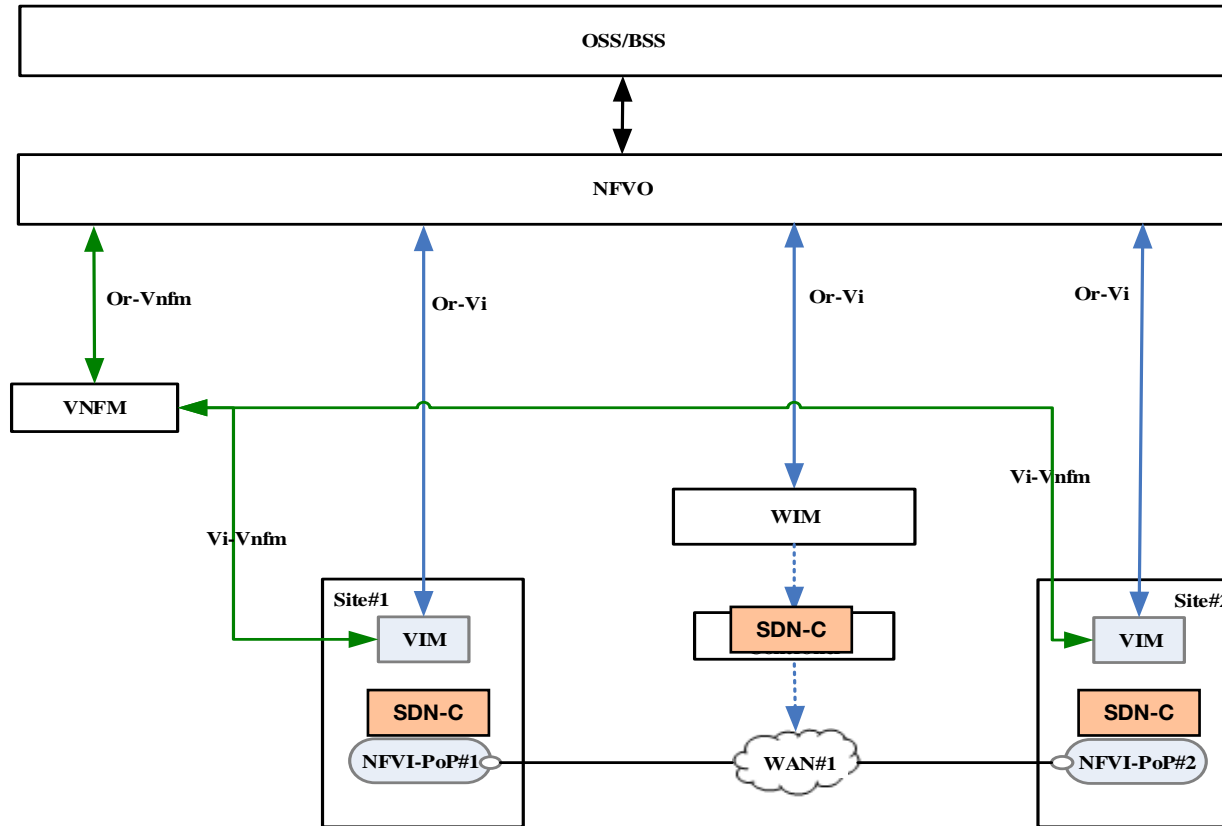
Implementing **NFV** using the **SDN** technology

NFV does not need SDN but may benefit from it
(and vice versa)

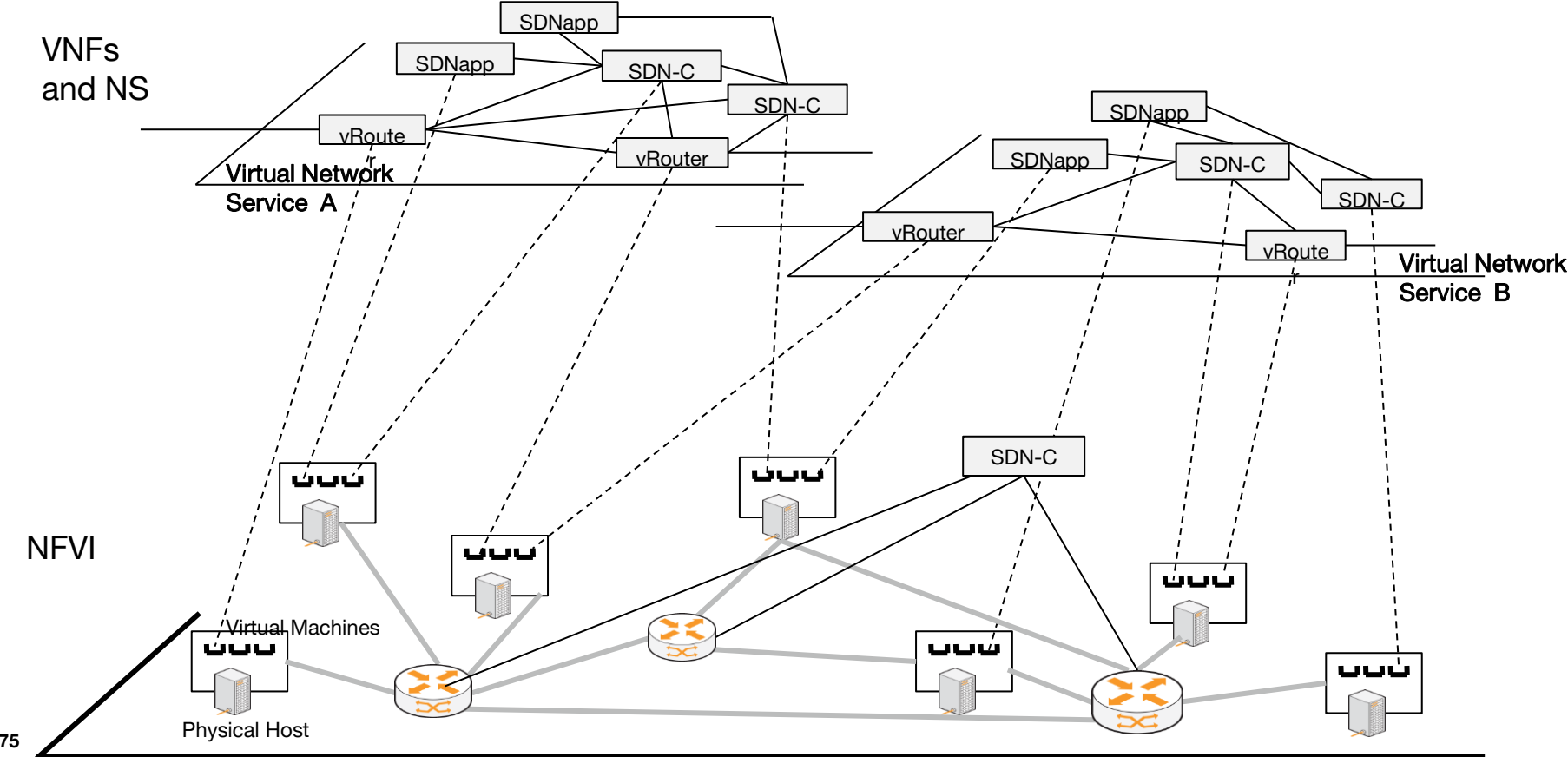
Role of SDN in NFV infrastructure networking



NFV Infrastructure - Multi-site connectivity



Recursive use of SDN in the NFV architecture



NFV, SDN and Service Function Chaining (SFC)

SFC refers to the definition of ordered sets of service functions (service function chains**) and to the mechanisms for the "steering" of traffic flows through them.**

– See IETF RFC 7665

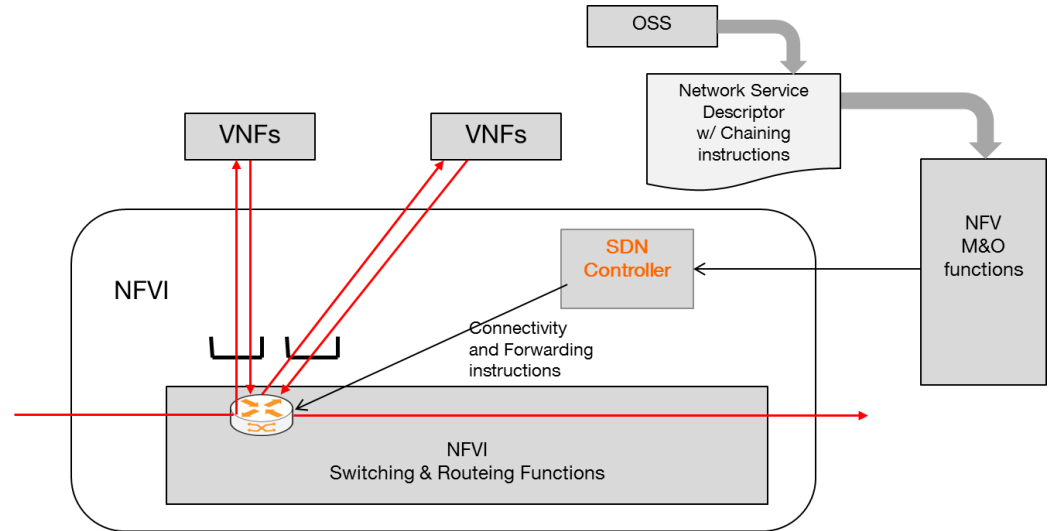
A Service Function can be implemented as a VNF, a set of VNFs, a VNFC, etc.

– See ETSI GS NF-IFA 014 Annex A

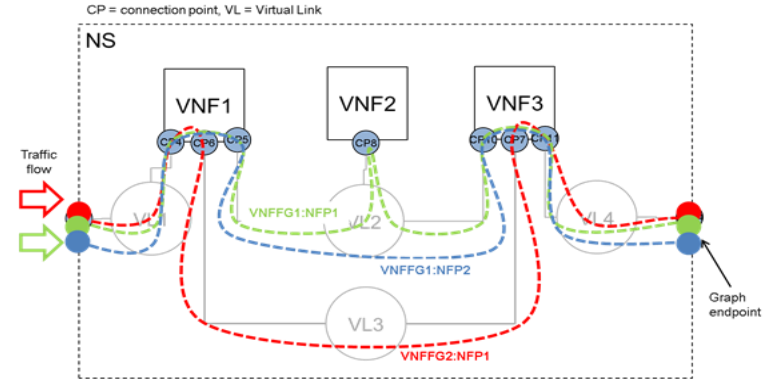
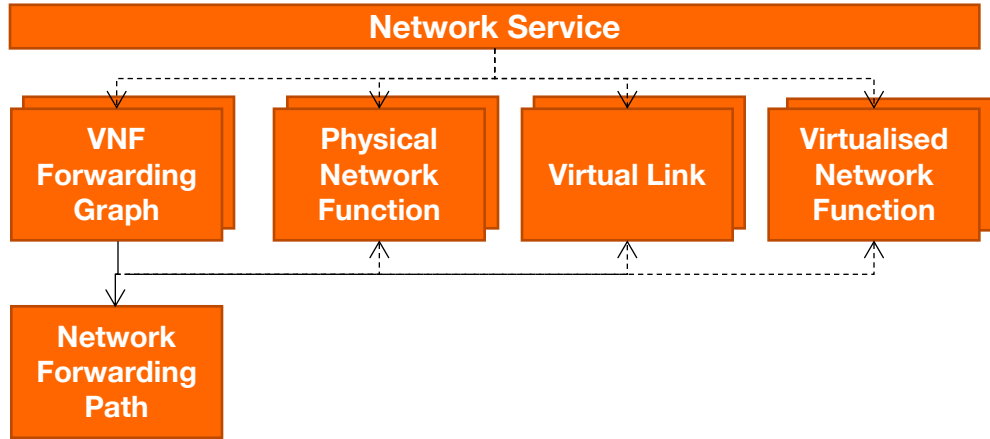
NFV, SDN and Service Function Chaining (SFC)

In an NFV environment, a service function chain is described by a **Network Forwarding Path Descriptor (NFPD)** within a **VNF Forwarding Graph Descriptor (VNFFGD)**.

The contents of these descriptors serves as an input for the NFVO to create the **actual service function paths**, in the NFVI, via the VIM and the associated SDN controllers.



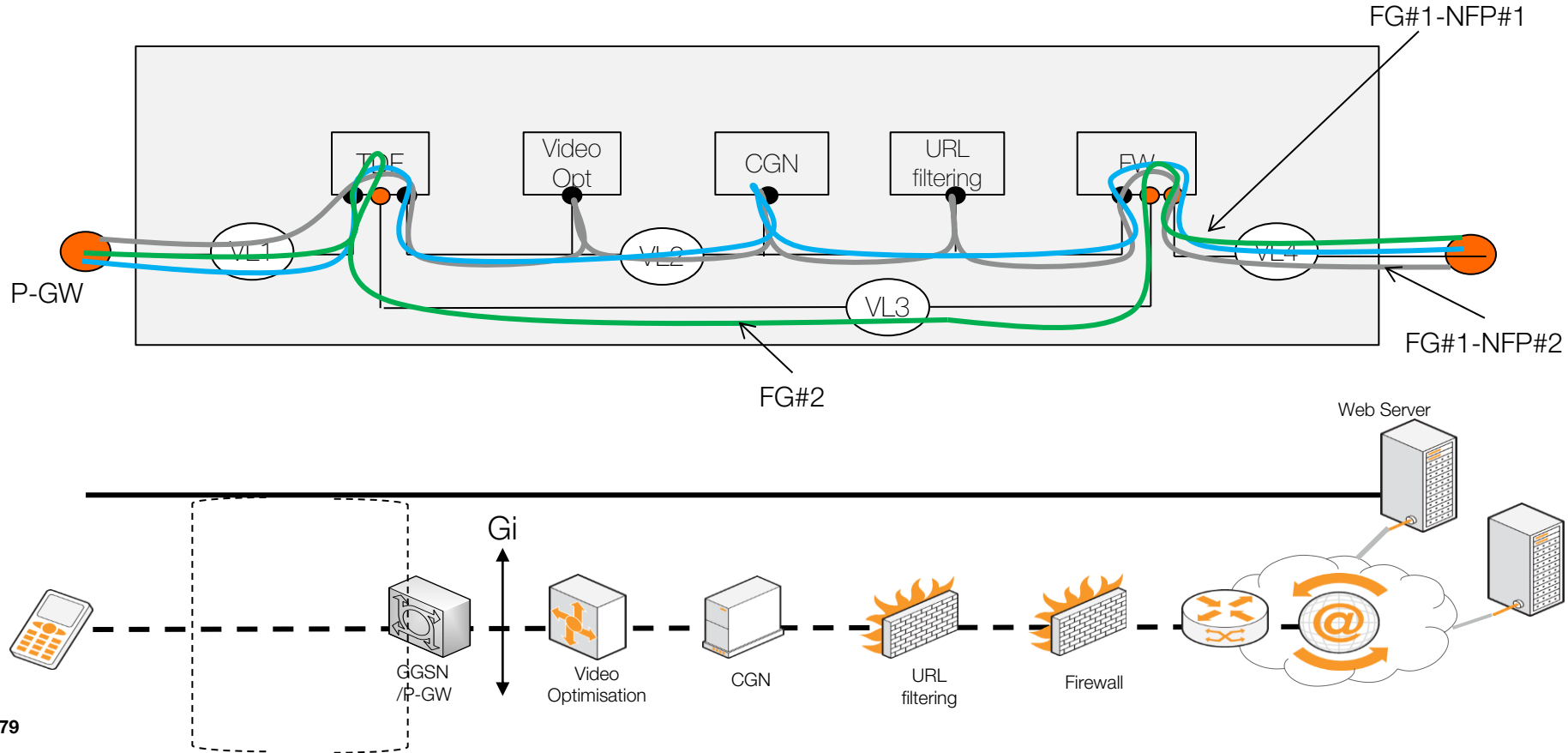
NFV Network Forwarding Paths



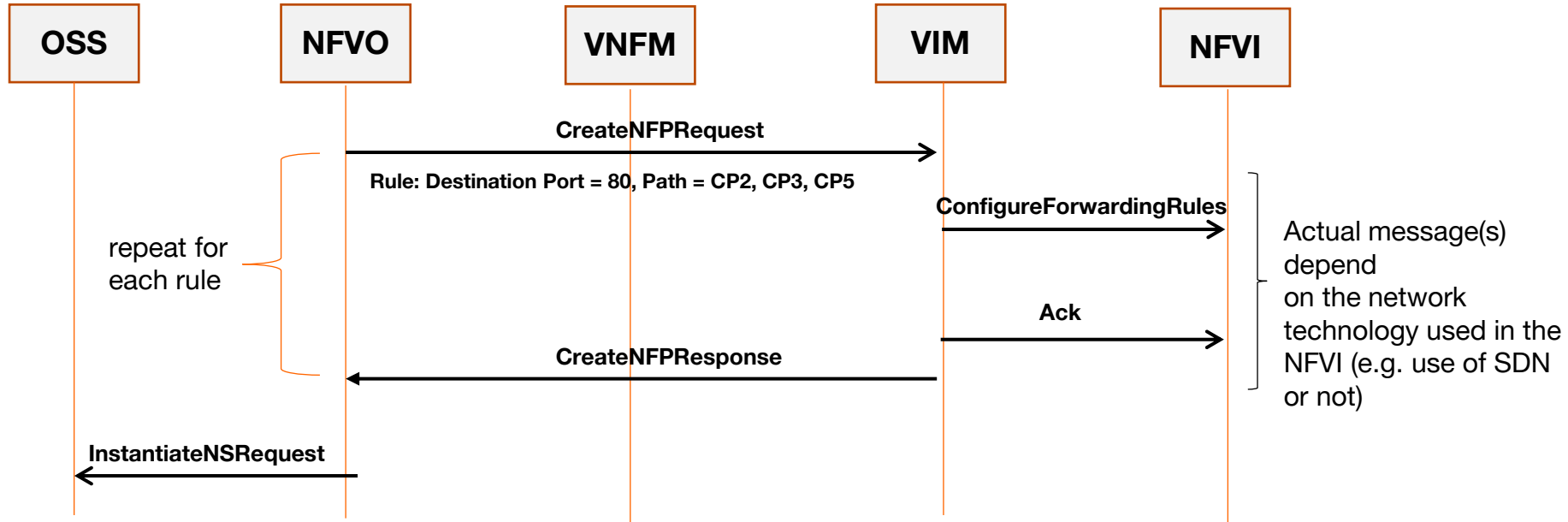
VNF Forwarding Graph: Describes a topology of the Network Service or a portion of the Network Service, by referencing VNFs and PNFs and Virtual Links that connect them.

Network Forwarding Path: Describes a sequence of NF to be traversed for specific traffic flows inside the forwarding graph.

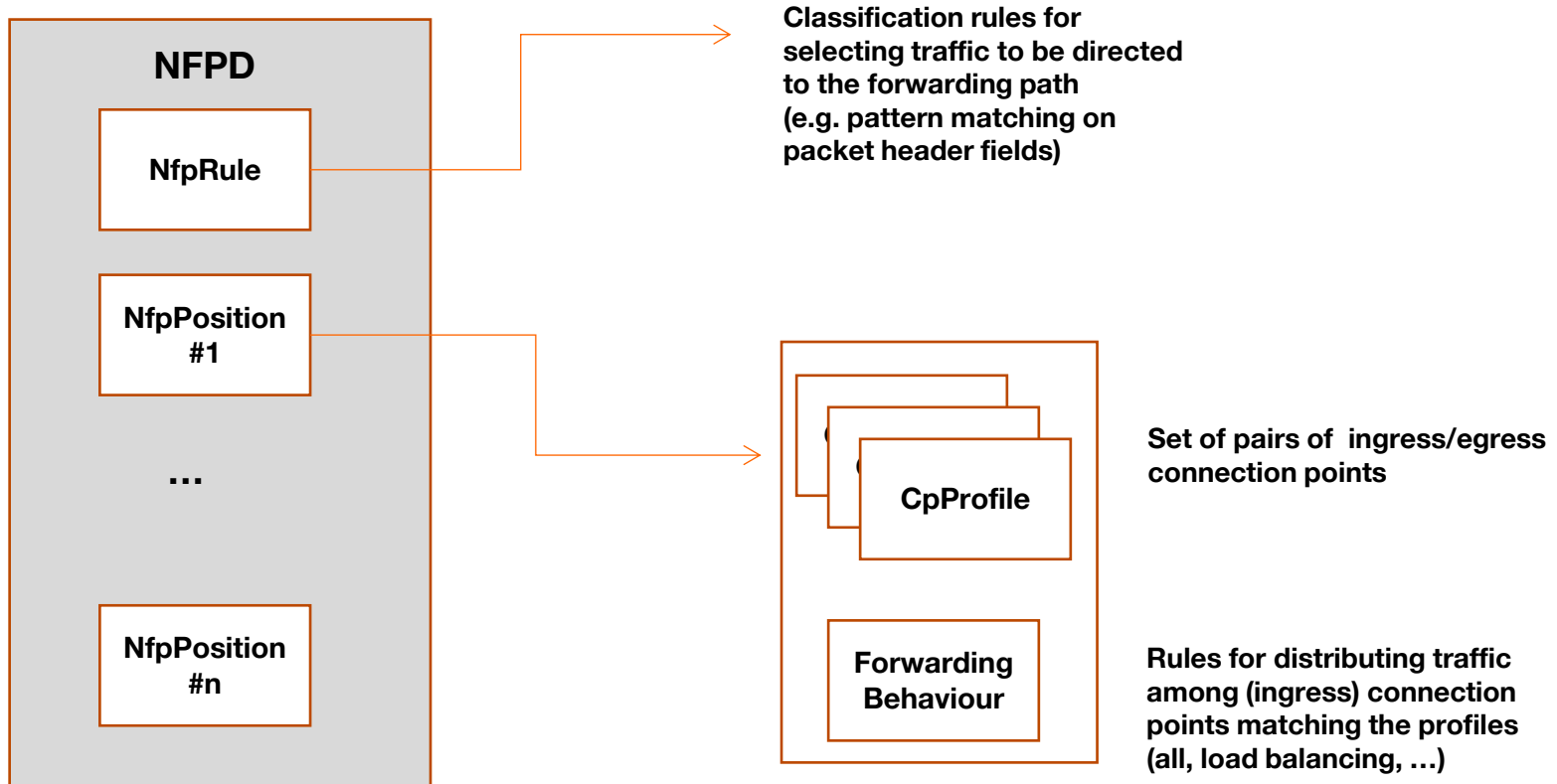
Application to Gi-LAN service chaining



Propagation of network forwarding paths



Contents of an NFPD



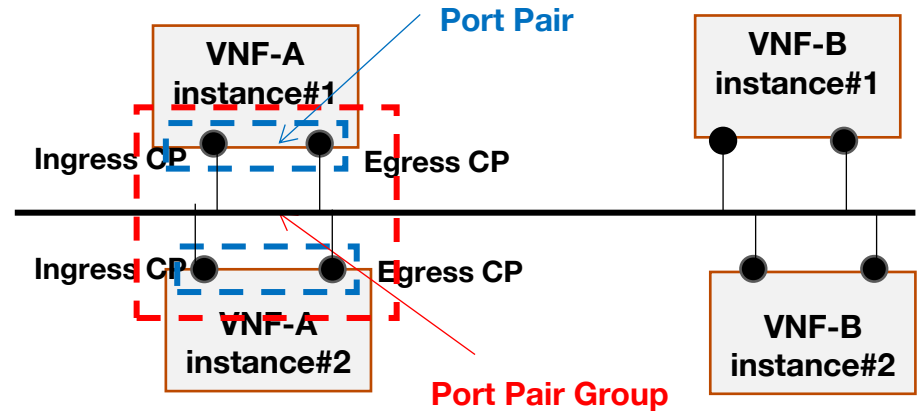
Service Function Chaining with OpenStack

The NFVO derives **Network Forwarding Paths** (NFPs) from the contents of Network Forwarding Path Descriptor (NFPD), embedded in the Network Service Descriptor (NSD).

For each NFP, the NFVO requests the VIM to **create a PortChain** using the OpenStack Neutron SFC API.

Prior to that the NFVO requests the VIM to create **PortPairGroups** and **FlowClassifiers** that the PortChain will use.

The VIM configures the vSwitches according to the PortChain description.



Agenda

Part I: Introduction, Architecture, Challenges

Part II: Focus on Management and Orchestration

Part III: NFV, SDN and Service Chaining

Part IV: NFV and 5G

NFV natively supports Network Slicing

“Network Functions Virtualisation in mobile networks can also be used to create core network instances optimized for specific services, e.g. for Machine-to-Machine communications (M2M).”

NFV White Paper, October 2012

Network Functions Virtualisation

An Introduction, Benefits, Enablers, Challenges & Call for Action

OBJECTIVES

This is a non-proprietary white paper authored by network operators.

The key objective for this white paper is to outline the benefits, enablers and challenges for Network Functions Virtualisation (as distinct from Cloud/SDN) and the rationale for encouraging an international collaboration to accelerate development and deployment of interoperable solutions based on high volume industry standard servers.

CONTRIBUTING ORGANISATIONS & AUTHORS

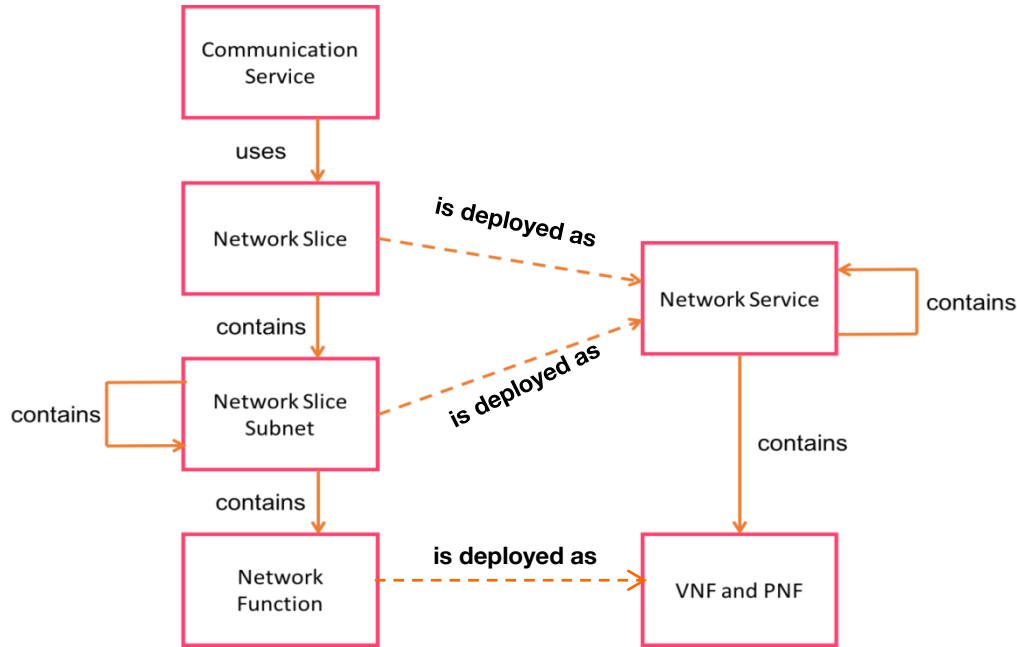
AT&T:	Margaret Chiosi.
BT:	Don Clarke, Peter Willis, Andy Reid.
CenturyLink:	James Feger, Michael Bugenhagen, Waqar Khan, Michael Fargano.
China Mobile:	Dr. Chunfeng Cui, Dr. Hui Deng.
Colt:	Javier Benitez.
Deutsche Telekom:	Uwe Michel, Herbert Damker.
KDDI:	Kenichi Ogaki, Tetsuro Matsuzaki.
NTT:	Masaki Fukui, Katsuhiro Shimano.
Orange:	Dominique Delisle, Quentin Loudier, Christos Koliass.
Telecom Italia:	Ivano Guardini, Elena Demaria, Roberto Minerva, Antonio Manzalini.
Telefonica:	Diego López, Francisco Javier Ramón Salguero.
Telstra:	Frank Ruhl.
Verizon:	Prodip Sen.

PUBLICATION DATE

October 22-24, 2012 at the “SDN and OpenFlow World Congress”, Darmstadt-Germany.

This white paper is available at the following link: http://portal.etsi.org/NFV/NFV_White_Paper.pdf

NFV and Network Slicing



In a resource-centric viewpoint, a **Network Slice** can be represented as a **Network Service instance** or a concatenation of Network Service instances.

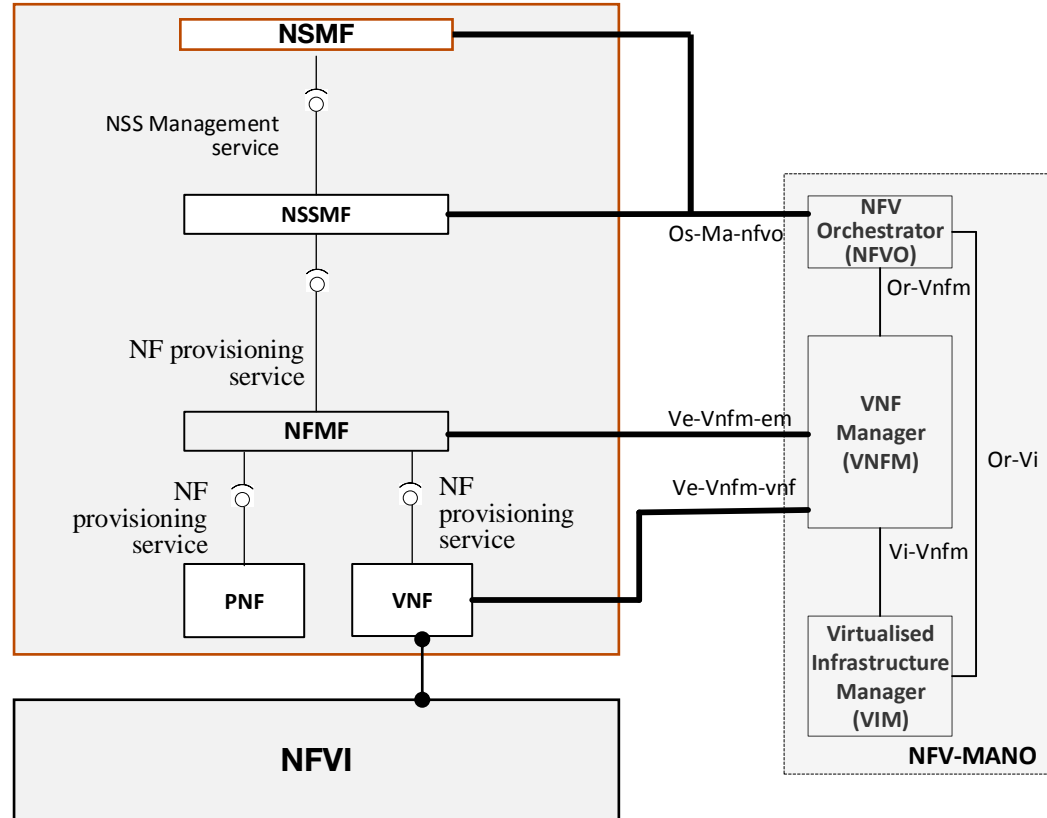
The virtualized resources for a slice subnet and their connectivity to physical resources can be represented by a **nested Network Service**, or one or more VNFs and PNFs directly attached to the Network Service used by the network slice.

Architectural touch points

The NSMF, NSSMF and NFMF consume the management services provided by the NFV-MANO functions.

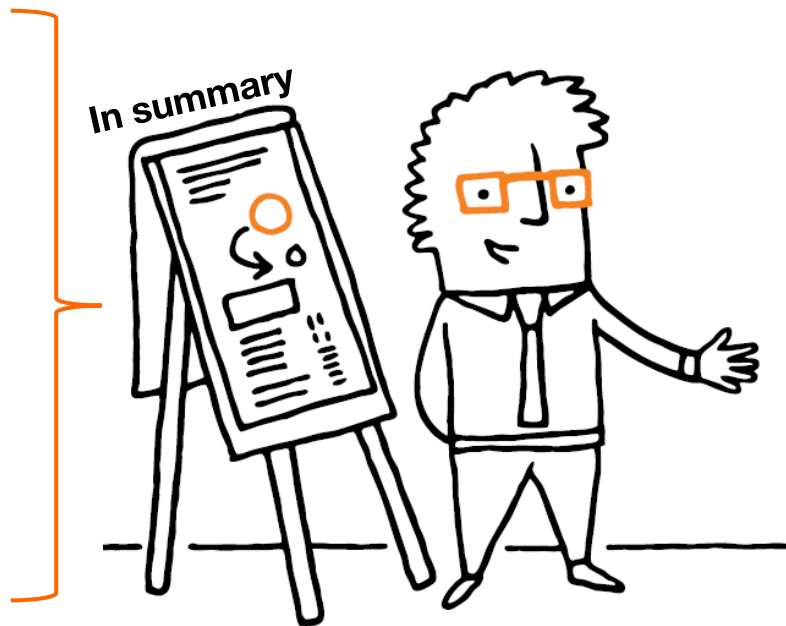
NSMF & NSSMF seen as an **OSS** element from the **NFVO** standpoint.

NFMF seen as an **EM** from the **VNFM** standpoint



Agenda

- Part I: Introduction, Concepts, Challenges**
- Part II: Focus on Management and Orchestration**
- Part III: NFV, SDN and Service Chaining**
- Part IV: NFV and 5G**



Summary

NFV is primarily a new approach to implementing network functions, leveraging cloud computing technologies and progress on the performance of COTS servers.

Management & Orchestration functions are the brain of an NFV system.

Automated data-driven life cycle management of network functions/services

SDN and NFV are different concepts but can bring mutual benefits to each other.

NFV comes with a lot of promises but key challenges should not be neglected: Performance vs. Portability, Integration with legacy OSS, Interoperability between vendors, and also operational processes and skills transformation for both vendors and network operators.

More information

NFV FAQ

https://nfvwiki.etsi.org/index.php?title=NFV_FAQ



On-line Webinars/Tutorials

VNF Package:

<https://www.brighttalk.com/webcast/12761/265769/nfv-tutorial-on-vnf-package-specification>

APIs and many more on YouTube!

https://www.youtube.com/playlist?list=PLINY888NYhG_C2GueQjhsVGHuTg8hRjzTR

ETSI Deliverables

All draft specifications and reports:

<https://docbox.etsi.org/ISG/NFV/Open/Drafts>

Easy access to published specifications

Specification” tab at

<http://www.etsi.org/technologies-clusters/technologies/nfv>

or

https://docbox.etsi.org/ISG/NFV/Open/Publications_pdf

Useful tips

A GR (e.g. ETSI GS NFV-IFA 009) is an informative document.

A GS (e.g. ETSI GS NFV-SOL 004) is a normative document if the 1st digit of the version identifier is equal or greater than 2).

ETSI GS NFV-IFA xxx documents are Stage 2 specifications

ETSI GS NFV-SOL xxx documents are Stage 3 specifications