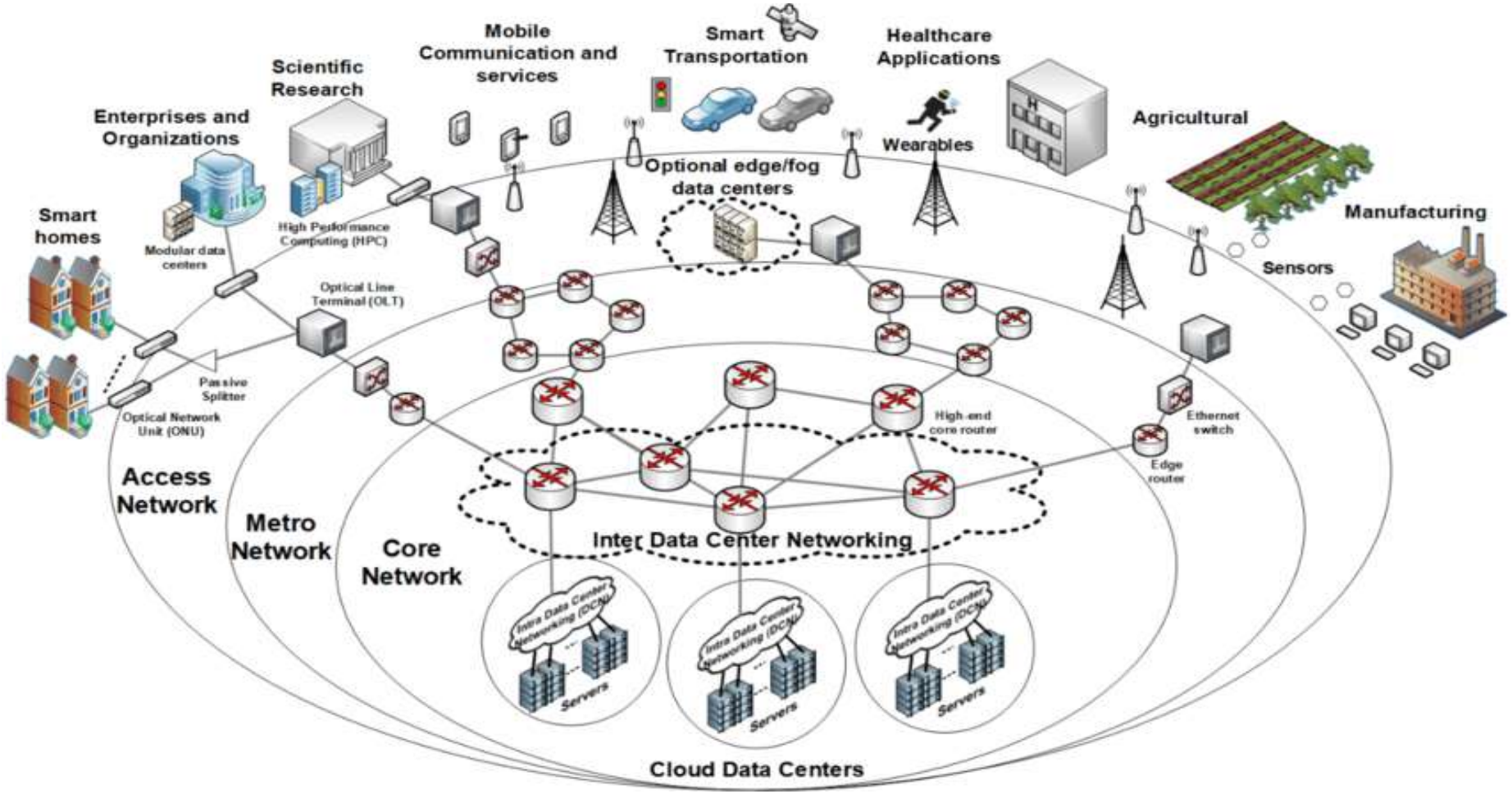
Chapter 3

Network Function Virtualization (NFV)

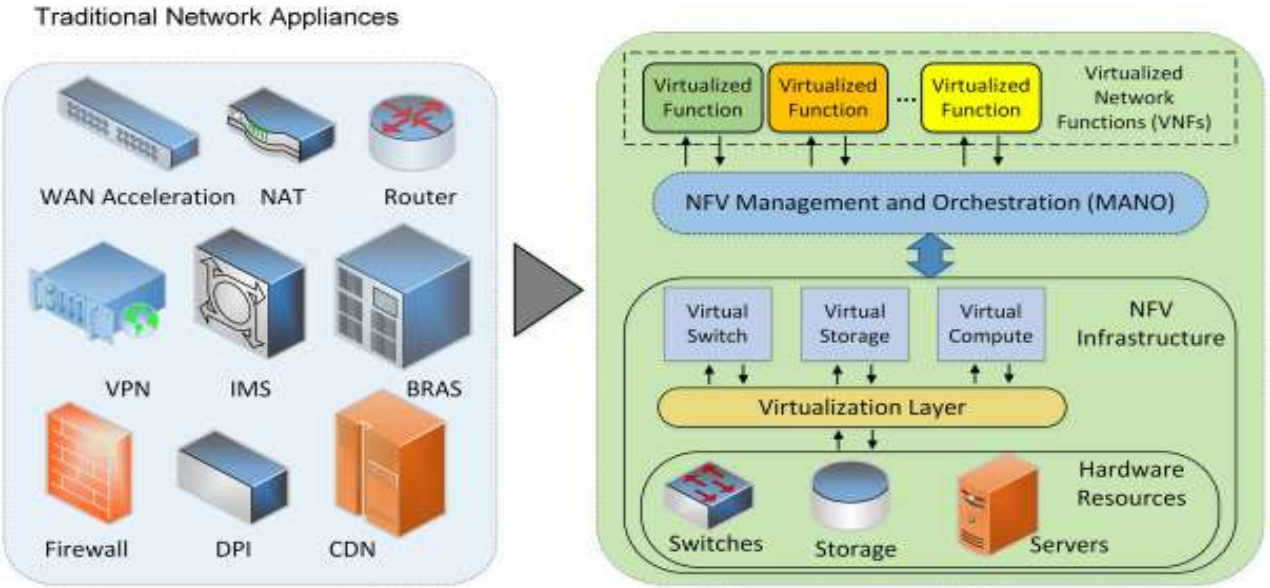
Rami Tawil

Evolution from Hardware-Based to Virtualized Network Functions



Scaling and upgrading required costly physical installations.

Evolution from Hardware-Based to Virtualized Network Functions

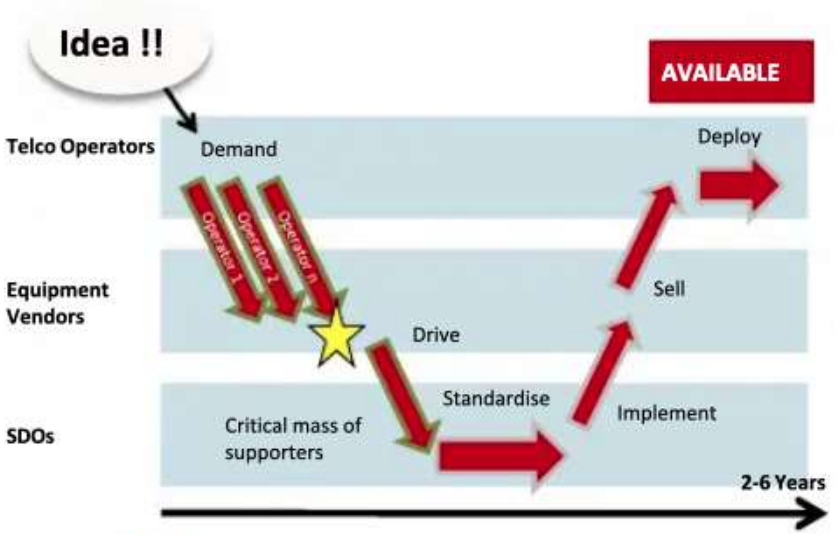


CDN: Content Delivery Network
IMS: IP Multimedia Subsystem
NAT: Network Address Translation
VPN: Virtual Private Network
BRAS: Broadband Remote Access Server
DPI: Deep Packet Inspection

- NFV replaces dedicated hardware with software-based Virtual Network Functions (VNFs) running on commodity servers.
- This transition improves flexibility, scalability, and cost-efficiency.

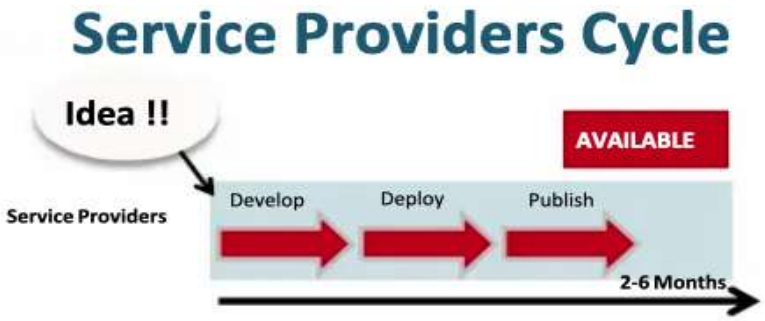
Implementing new idea

Hardware-Based



2-6 years

Virtualized Network Functions



2-6 months

Motivation and Advantages of NFV

- Reduce dependency on proprietary network hardware.
- Lower capital and operational costs (CAPEX and OPEX).
- Enable rapid deployment of new network services.
- Increase scalability through on-demand resource allocation.
- Simplify management and automation using orchestration frameworks.
- Enhance innovation by decoupling software from hardware vendors.

Problem Statement

Complex carrier networks

with a large variety of proprietary nodes and hardware appliances.

Launching new services is difficult and takes too long

Space and power to accommodate

Requires just another variety of box, which needs to be integrated.

Operation is expensive

Rapidly reach end of life

Due to existing procure-design,-integrate-deploy cycle.

- Network functionalities are based on specific HW&SW
- One physical node per role Traditional Network model

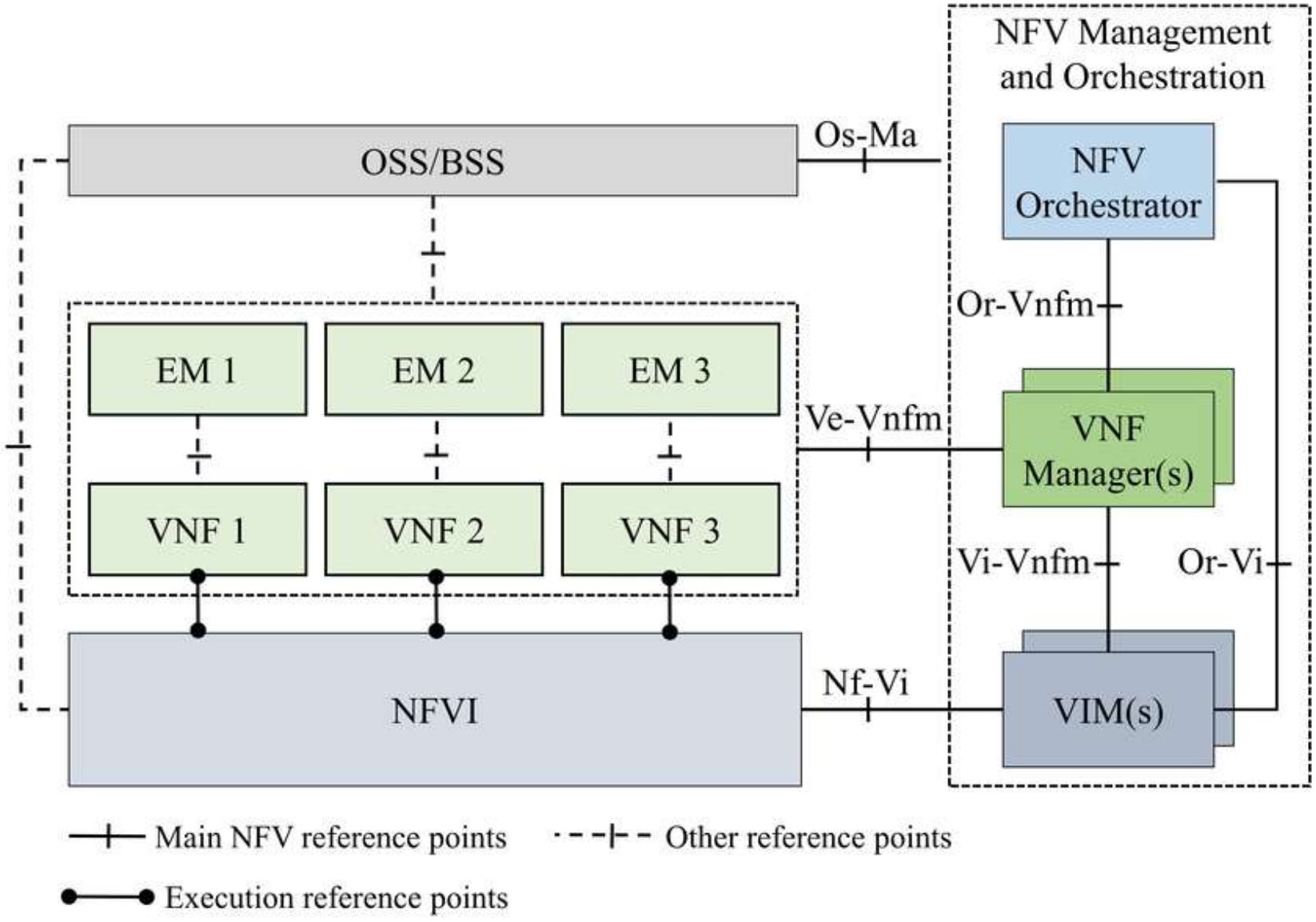
NFV Use Cases: 5G, Cloud Services, IoT, and Enterprise Networks

- **5G Networks**: Enables core network functions (EPC, AMF, SMF) as virtualized services.
- **Cloud Services**: Supports scalable and flexible virtualized data centers.
- **IoT**: Facilitates dynamic resource management for billions of connected devices.
- **Enterprise Networks**: Allows deployment of virtual firewalls, VPNs, and load balancers on demand.

NFV Architecture and Standards

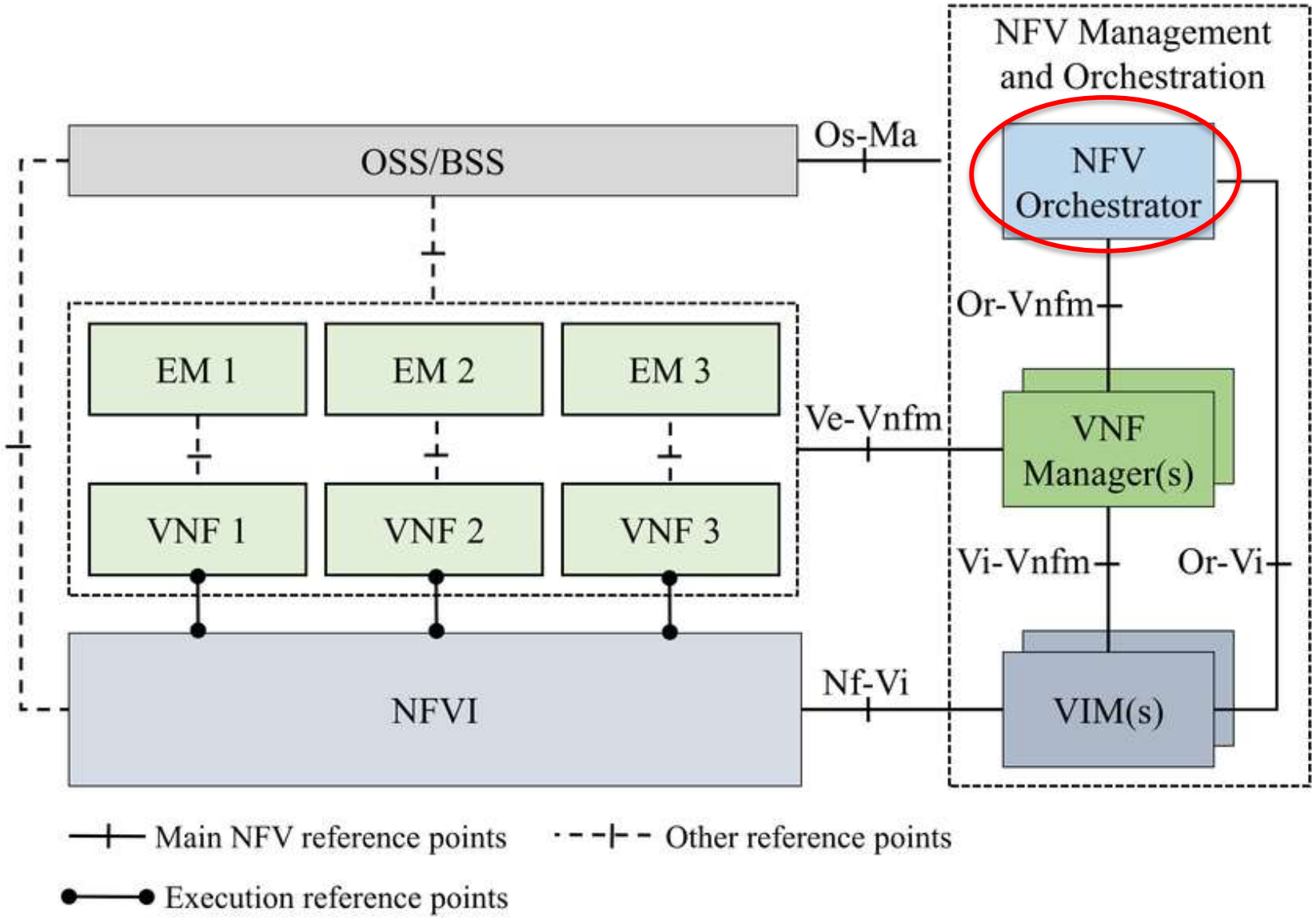
- European Telecommunications Standards Institute (ETSI) NFV Architectural Framework
- Core components: NFVI, VNFs, MANO
- Interfaces and reference points
- Example deployments

ETSI NFV Architectural Framework



NFV MANAGEMENT AND ORCHESTRATION

ETSI NFV Architectural Framework



NFV Reference Points

NFV Orchestrator

- The central brain and automation engine in an NFV architecture.
- Primary role: manage the end-to-end lifecycle of Network Services *(chains of Virtualized Network Functions (VNFs))* by coordinating resources and processes across the entire NFV infrastructure.

NFV Reference Points

NFV Orchestrator

Key Responsibilities and Functionalities

1- Network Service Lifecycle Management (core function)

- **Instantiation:** reads a blueprint (Network Service Descriptor) and automatically coordinates with the VNF Manager(s) and VIM to create, connect, and launch all the VNFs in the correct order.
- **Scaling:** can automatically scale a service in or out based on policies and traffic demands (e.g., adding more virtual firewall instances during peak load).
- **Healing:** If a service fails, it can initiate recovery actions, like restarting a VNF or moving it to a healthy server.
- **Update/Upgrade:** It manages coordinated software upgrades for the entire service without causing downtime.
- **Termination:** It cleanly shuts down and decommissions the entire service, freeing up all resources.

NFV Reference Points

NFV Orchestrator

Key Responsibilities and Functionalities

2- Resource Orchestration & Global Inventory

- Maintains a global view of all available resources (compute, storage, network) across multiple data centers and cloud sites by talking to one or more VIMs.
- It makes intelligent placement decisions for VNFs based on policies
  - "place this VNF in the data center closest to the user" or "ensure these two VNFs are on separate physical servers for redundancy".

3- Policy Management and Enforcement

- Implements high-level business and operational policies, such as Service Level Agreements (SLAs), security rules, and affinity/anti-affinity rules
  - "these VNFs must not run on the same physical host”.

NFV Reference Points

NFV Orchestrator

Key Responsibilities and Functionalities

4- Service Catalog Management

- Maintains a catalog of available VNFs and pre-defined Network Service blueprints, which are templates that define the service's structure, dependencies, and resource requirements.

5- Interaction with OSS/BSS

- Provides a northbound interface for the traditional OSS/BSS.
    - When the OSS requests a new service for a customer, the NFVO receives that request and executes the complex orchestration process to deliver it.

Service Catalog Management in Action

1.Catalog Contents:

1. The **Service Catalog** contains a list of all available **VNFs** and **Network Services**, such as:
    1. Virtual EPC (vEPC)
    2. Virtual IMS (vIMS)
    3. Virtual Firewall (vFW)
    4. Load Balancer (vLB)
    5. Network Slice Template for 5G services

2.Blueprints:

1. Each service has a **blueprint** (template) that defines:
    1. **Structure**: What VNFs are included (e.g., vMME, vSGW, vPGW for EPC)
    2. **Dependencies**: Which functions depend on others
    3. **Resources**: Required CPU, memory, and bandwidth
    4. **Configuration**: Network topology and interfaces

3.Use Case:

1. When Vodafone launches a new **5G enterprise service** for a client:
    1. The orchestrator selects the appropriate **Network Service Blueprint** (e.g., “5G Slice for IoT”).
    2. The blueprint automatically triggers deployment of all required VNFs (firewall, load balancer, EPC components) from the catalog.
    3. This ensures a consistent, tested configuration is used across all deployments.

4.Benefits:

1. Reduces **service deployment time** from weeks to minutes.
2. Ensures **standardization** of network services.
3. Supports **multi-vendor VNFs** seamlessly.

Interaction with OSS/BSS

1.Customer Service Request (BSS):

1. A customer (for example, a business client) orders a **VPN** or a **5G enterprise slice** through AT&T's customer portal (part of the **Business Support System — BSS**).
2. The BSS sends a **service order request** to the **OSS**.

2.Service Order (OSS → NFVO):

1. The **Operations Support System (OSS)** translates the service order into **technical service parameters** (e.g., bandwidth, latency, location).
2. OSS sends this information to the **NFV Orchestrator (NFVO)** using the **northbound interface**.

3.Orchestration (NFVO):

1. The **NFVO** triggers the orchestration process:
    1. Selects a **Network Service Blueprint** from the **Service Catalog** (e.g., a VPN or 5G slice template).
    2. Allocates compute, storage, and network resources from the **NFV Infrastructure (NFVI)**.
    3. Deploys and configures the required **VNFs** (e.g., virtual routers, firewalls, gateways).

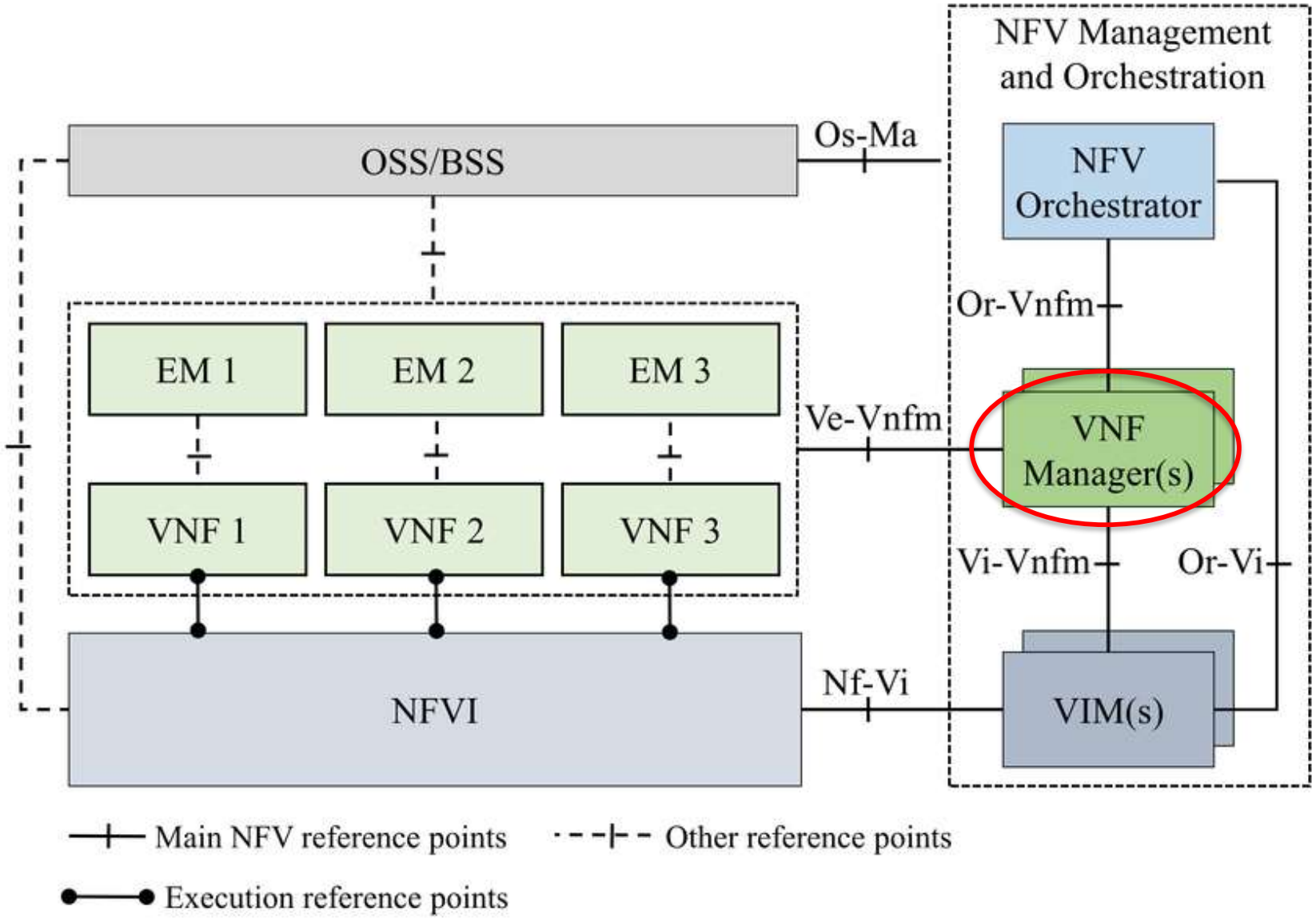
4.Status & Feedback:

1. The NFVO sends **status updates** (success, error, progress) back to the OSS/BSS.
2. Once deployment completes, the OSS updates the **customer account** and activates billing automatically through the **BSS**.

Benefits

- **Automated end-to-end service delivery** — no manual provisioning needed.
- **Seamless integration** between customer-facing systems (BSS) and network control (NFVO).
- **Improved time-to-market** for new telecom services (from weeks to minutes).

ETSI NFV Architectural Framework



NFV Reference Points

VNF Manager (VNFM)

Responsible for the *lifecycle management of individual instances of Virtualized Network Functions (VNFs).*

In simpler terms, if the NFV Orchestrator (NFVO) is the **conductor** of the entire orchestra (managing the full network service), the VNFM is the *section leader* for a specific group of instruments

NFV Reference Points

VNF Manager (VNFM)

Key Responsibilities and Functionalities

1- VNF Instance Lifecycle Management

- **Instantiation:** receives a request from the NFVO and executes the steps to create and boot a new VNF instance (e.g., a virtual firewall).

  This often involves interacting with the VIM to allocate resources.

- **Scaling:** performs the scaling actions (scale-in/out, scale-up/down) on a VNF based on triggers from the NFVO or directly from the VNF/EMS.
  - **For example**: it can add a new virtual CPU to a VNF (scale-up) or deploy a new instance of the VNF (scale-out)
- **Healing:** monitors the health of the VNF and can take corrective action if it fails
  - **For example**: restarting the VNF software or rebuilding the VNF on a new host.
- **Update/Upgrade:** manages the process of updating or patching the VNF's software.
- **Termination:** gracefully shuts down and deletes the VNF instance, ensuring all

NFV Reference Points

VNF Manager (VNFM)

Key Responsibilities and Functionalities

2- Configuration and Event Reporting

- Handle the initial configuration of the VNF after it's instantiated.
- Collects performance measurements (PM) and fault information from the VNF and its underlying virtual resources, forwarding relevant events and data to the NFVO.

3- Policy Enforcement for a VNF

- The VNFM enforces lifecycle management policies specific to a VNF.
    - For example, if a policy states, "If CPU usage > 80% for 5 minutes, scale out," the VNFM is the component that executes that scaling action.

NFV Reference Points

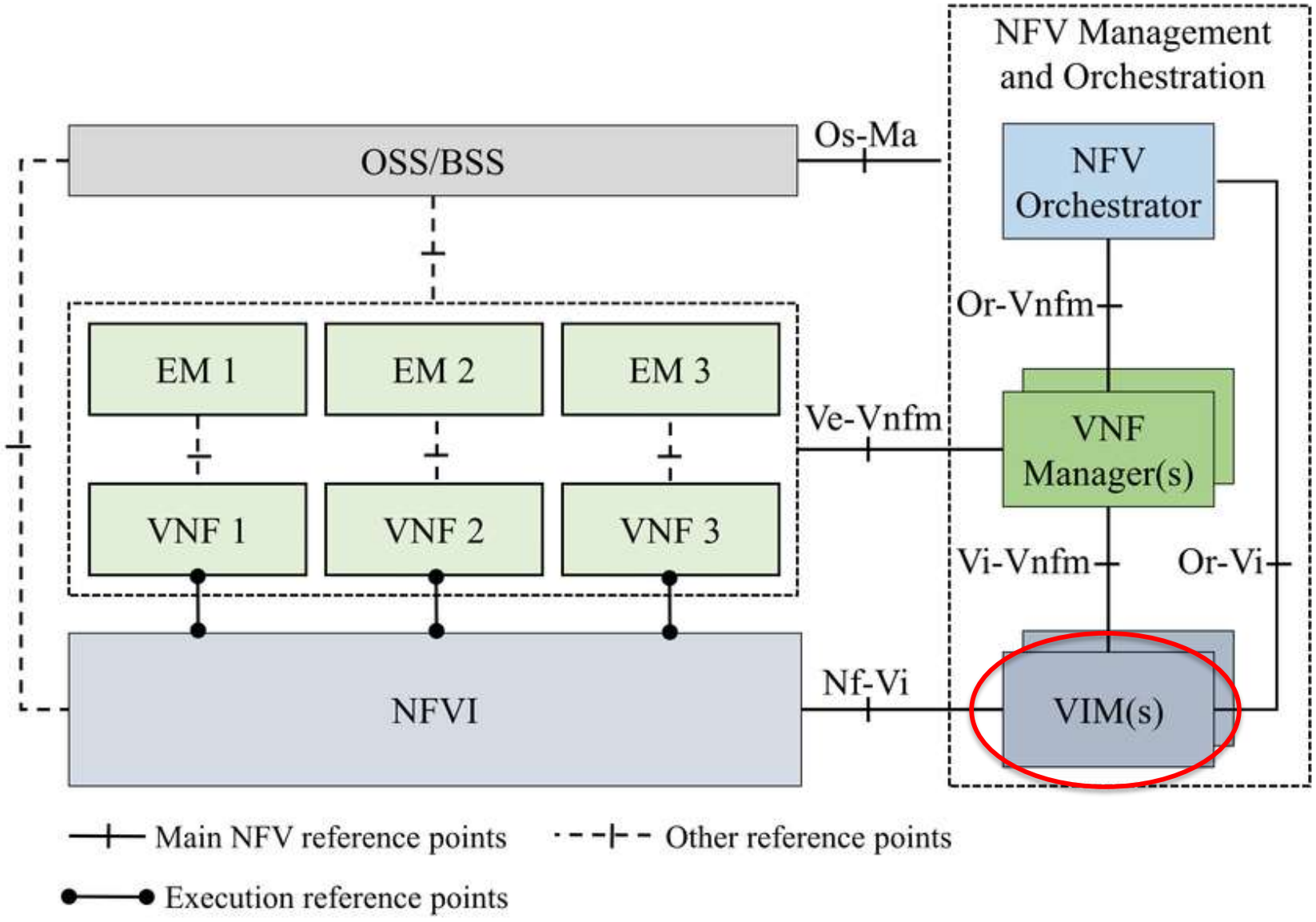
VNF Manager (VNFM)

Key Responsibilities and Functionalities

4- Interaction with the VNF and its EMS

- Communicates with the VNF's built-in management system or its traditional Element Management System (EMS) to perform configuration and receive VNF-specific alarms and metrics

ETSI NFV Architectural Framework



NFV Reference Points

Virtualized Infrastructure Manager (VIM)

Responsible for *controlling and managing the NFV Infrastructure (NFVI)*—the compute, storage, and networking resources that form the cloud platform upon which VNFs run.

So, if the NFVI is the *physical data center* (servers, disks, switches) and its virtualization layer, the VIM is the *operating system and resource manager* for that data center.

NFV Reference Points

Virtualized Infrastructure Manager (VIM)

Key Responsibilities and Functionalities

1- Resource Management and Inventory

- **Discovery and Inventory:** maintains a real-time inventory of all physical and virtual resources under its control
    - How many CPU cores, how much RAM, available storage capacity, and network topology.
- **Allocation and Scheduling:** responsible for allocating specific compute, storage, and network resources to VNFs upon request from the VNFM or NFVO.
    - Decides *which* physical server will host a new VNF instance.

NFV Reference Points

Virtualized Infrastructure Manager (VIM)

Key Responsibilities and Functionalities

2- Orchestration of Virtual Resources

- Doesn't just allocate resources in isolation; it orchestrates them to create complex virtual environments. This includes:
    - Creating and managing Virtual Machines (VMs) or containers.
    - Creating and connecting virtual networks (e.g., VLANs, VxLANs).
    - Attaching virtual storage volumes to VMs.

NFV Reference Points

Virtualized Infrastructure Manager (VIM)

Key Responsibilities and Functionalities

3- Infrastructure Performance and Fault Management

- Continuously monitors the health and performance of the NFVI.
- Collects metrics like physical server CPU utilization, hypervisor performance, network port statistics, and storage I/O.
- Detects hardware and hypervisor failures (e.g., a server power supply failure, a disk array error) and generates alerts.

NFV Reference Points

Virtualized Infrastructure Manager (VIM)

Key Responsibilities and Functionalities

4- Image Management

- Stores and manages the software images (e.g., QCOW2, VMDK files) that are used to boot VNFs. When a VNF needs to be instantiated, the VIM retrieves the correct image and uses it to create the VM.

5- Implementation of Network Policies

- The VIM enforces low-level network connectivity and security policies defined by higher-level orchestrators. For example, it configures the virtual switches to ensure that the "WAN" port of a virtual firewall is connected to the external network and the "LAN" port is connected to the internal service network.

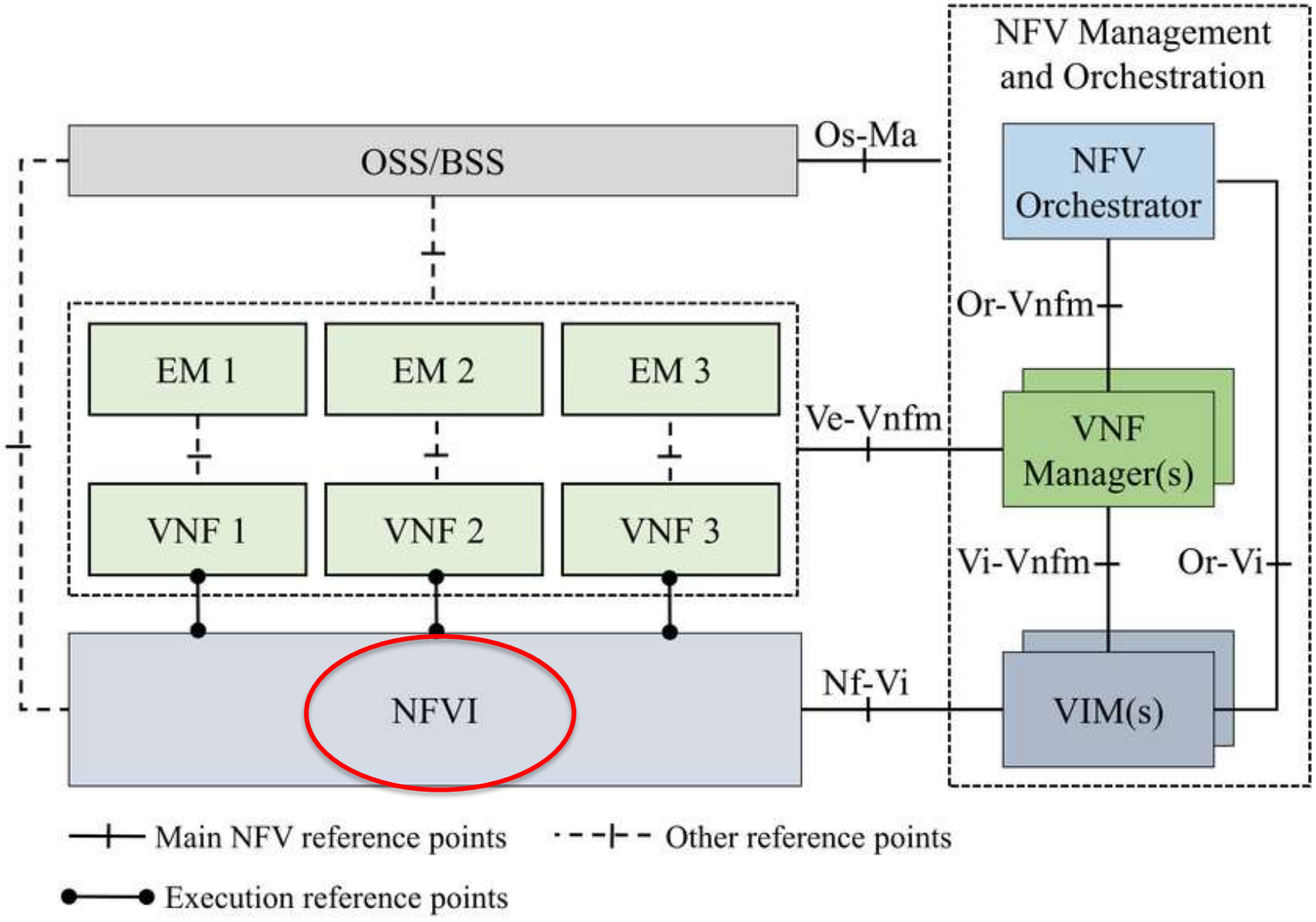
NFV Reference Points

Virtualized Infrastructure Manager (VIM)

VIMs are essentially cloud management platforms. The most common examples are:

- **OpenStack** (and its components like Nova, Neutron, Cinder)
- **VMware vCloud Director** or the **vCenter Server** suite
- **Kubernetes** (when managing container-based VNFs)
- **Public Cloud APIs** (e.g., from AWS, Microsoft Azure, Google Cloud)

ETSI NFV Architectural Framework



NFV Reference Points

NFV Infrastructure (NFVI)

- Foundational layer of hardware and software resources that provides the *virtualized environment upon which Virtualized Network Functions (VNFs) are deployed and run.*
- It is the "cloud" in the telecom network, abstracting physical resources to create a *flexible*, *on-demand pool of compute*, *storage*, and *networking*.
- So, if VNFs are the **apps** on your smartphone, the NFVI is the **smartphone itself**

NFV Reference Points

NFV Infrastructure (NFVI)

NFVI is composed of two essential layers that work together:

1. Physical Layer (Physical Resources)

- **Compute Hardware:** High-volume servers (often using Commercial Off-The-Shelf - COTS - hardware) that provide the processing power (CPUs).
- **Storage Hardware:** Hard disk drives (HDDs), solid-state drives (SSDs), and storage area networks (SANs) that provide capacity for VNF images and data.
- **Networking Hardware:** Physical switches, routers, and network interface cards (NICs) that interconnect the servers and provide links to the external network.

2. Virtualization Layer (Virtualization Software)

- **Hypervisor** or Virtual Machine Monitor (VMM), such as: KVM, VMware ESXi, runs on the physical servers and is responsible for creating and running the Virtual Machines (VMs) or containers that host the VNFs.
- It allocates physical resources (CPU, RAM) to these VMs.

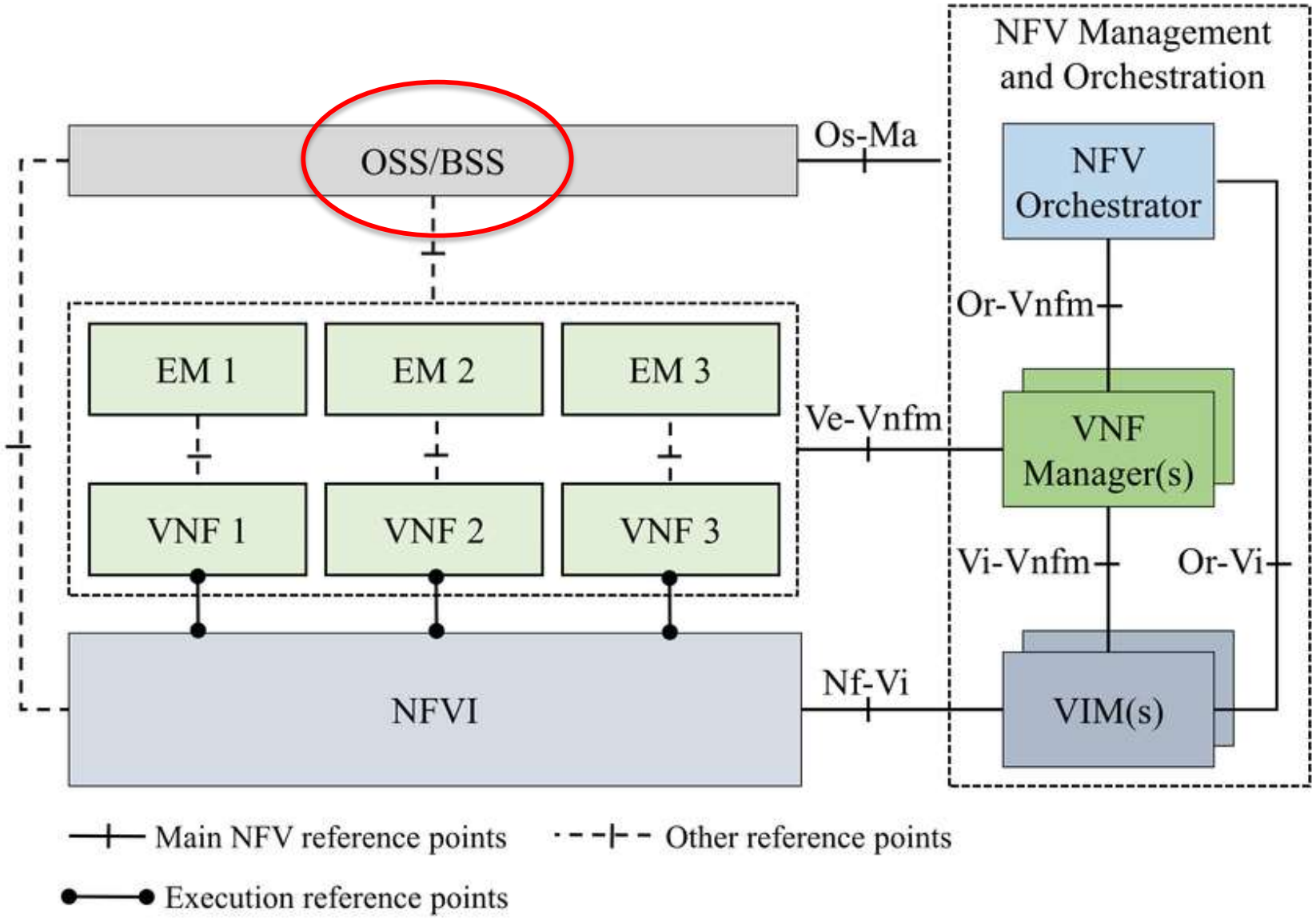
NFV Reference Points

NFV Infrastructure (NFVI)

Key Responsibilities and Characteristics

- **Resource Pooling:** Aggregates all physical resources into a shared pool that can be dynamically allocated to any VNF as needed.
- **Abstraction:** Hides the complexity and specifics of the underlying hardware from the VNFs. A VNF doesn't need to know what brand of server it's running on; it just sees the virtual resources (vCPU, vRAM, vNIC) it has been assigned.
- **Elasticity:** The infrastructure can scale resources up and down on demand, allowing VNFs to be scaled seamlessly to handle changing traffic loads.
- **Isolation:** Ensures that multiple VNFs from different vendors or services can run securely side-by-side on the same physical hardware without interfering with each other.

ETSI NFV Architectural Framework



OSS

OPERATIONAL SUPPORT SYSTEM

ETSI NFV Architectural Framework

- **OSS**
  - The "brain" that tells the equipment what to do and ensures everything is running smoothly (manage, monitor, analyze, and control the network operations).
- **Key functions**
  - **Network Inventory:** Keeping a detailed record of every piece of physical and logical network equipment.
  - **Fault Management:** Continuously monitoring the network for failures (e.g., a fiber cut, a router crash, a failing circuit).
  - **Configuration Management:** Remotely configuring network devices (e.g., setting up a new customer's internet connection, updating software on a cell tower).
  - **Performance Management:** Collecting and analyzing performance data (e.g., bandwidth utilization, latency, packet loss, signal strength).
  - **Service Provisioning & Activation:** The process of automatically setting up and activating a new service for a customer (e.g., activating a new fiber line or a 5G mobile plan).

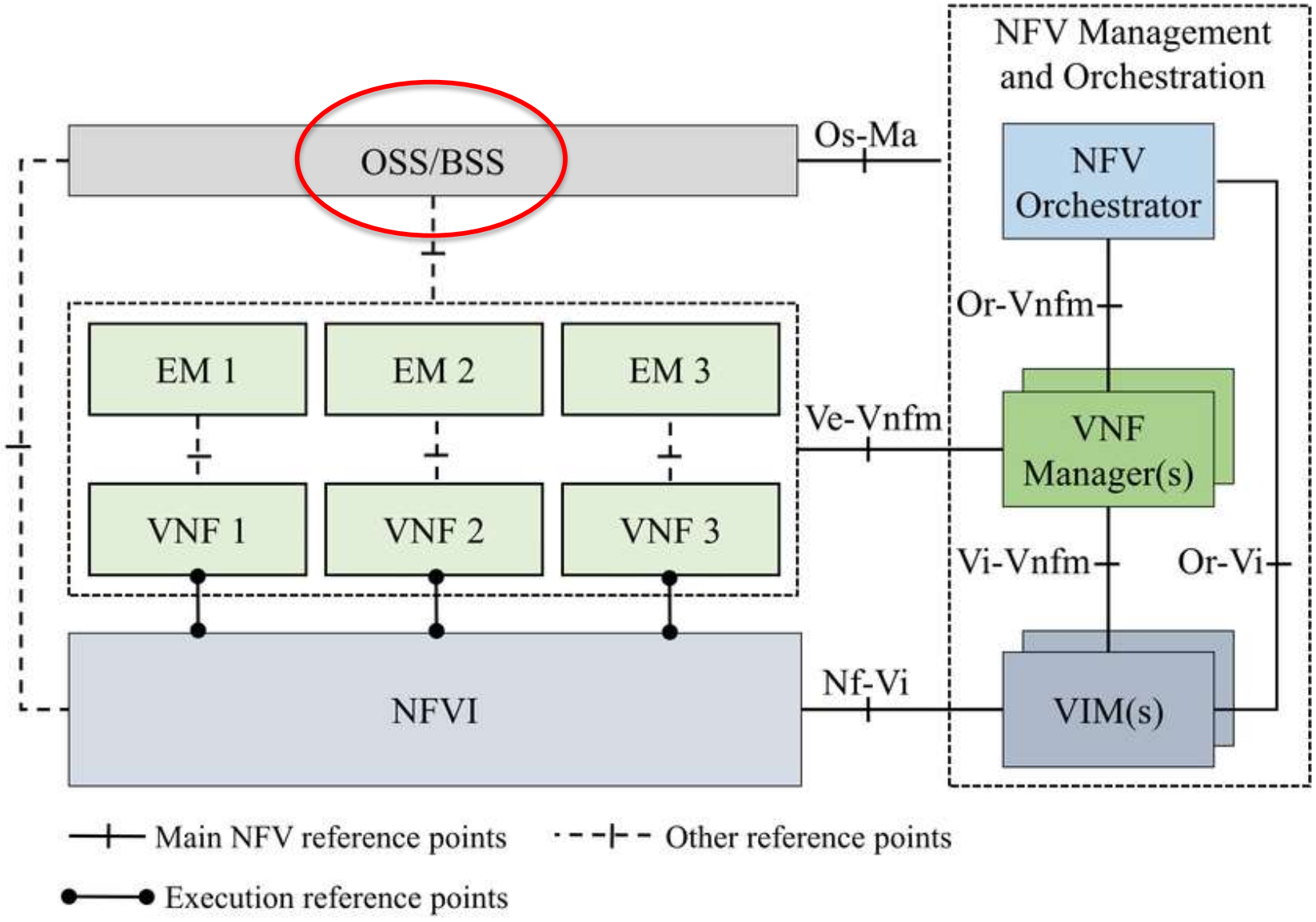
ETSI NFV Architectural Framework

- **OSS Example (Activating a New Home Fiber Internet Customer)**

Alex," who orders a 1 Gbps fiber internet package from "Telecom Giant Inc." The OSS platforms work behind the scenes to make this happen

  - **Step 1:** Order Receipt and Service Design
  - **Step 2:** Network Inventory Check (OSS Function)
  - **Step 3:** Service Provisioning & Activation (OSS Function)
  - **Step 4:** Work Order Dispatch
  - **Step 5:** Installation and Activation
  - **Step 6:** Ongoing Monitoring & Assurance (OSS Function)

ETSI NFV Architectural Framework



BSS

BUSINESS SUPPORT SYSTEM

ETSI NFV Architectural Framework

- **BSS**
  - Describe the business and/or customer-facing functionality.
- **Key functions**
  - **Customer Relationship Management (CRM):** The single view of the customer, managing all interactions, sales history, and contact information.
  - **Order Management:** Capturing and managing the customer's order for a new service, upgrade, or change.
  - **Billing and Revenue Management**: Generating invoices, processing payments, applying discounts...
  - **Product Catalog:** Defining the sellable products and services (e.g., "1 Gbps Internet," "Unlimited Mobile Plan," "Streaming TV Bundle") and their rules.
  - **Fulfillment Orchestration:** Taking the customer's order and triggering the correct processes in the OSS to have the service delivered.

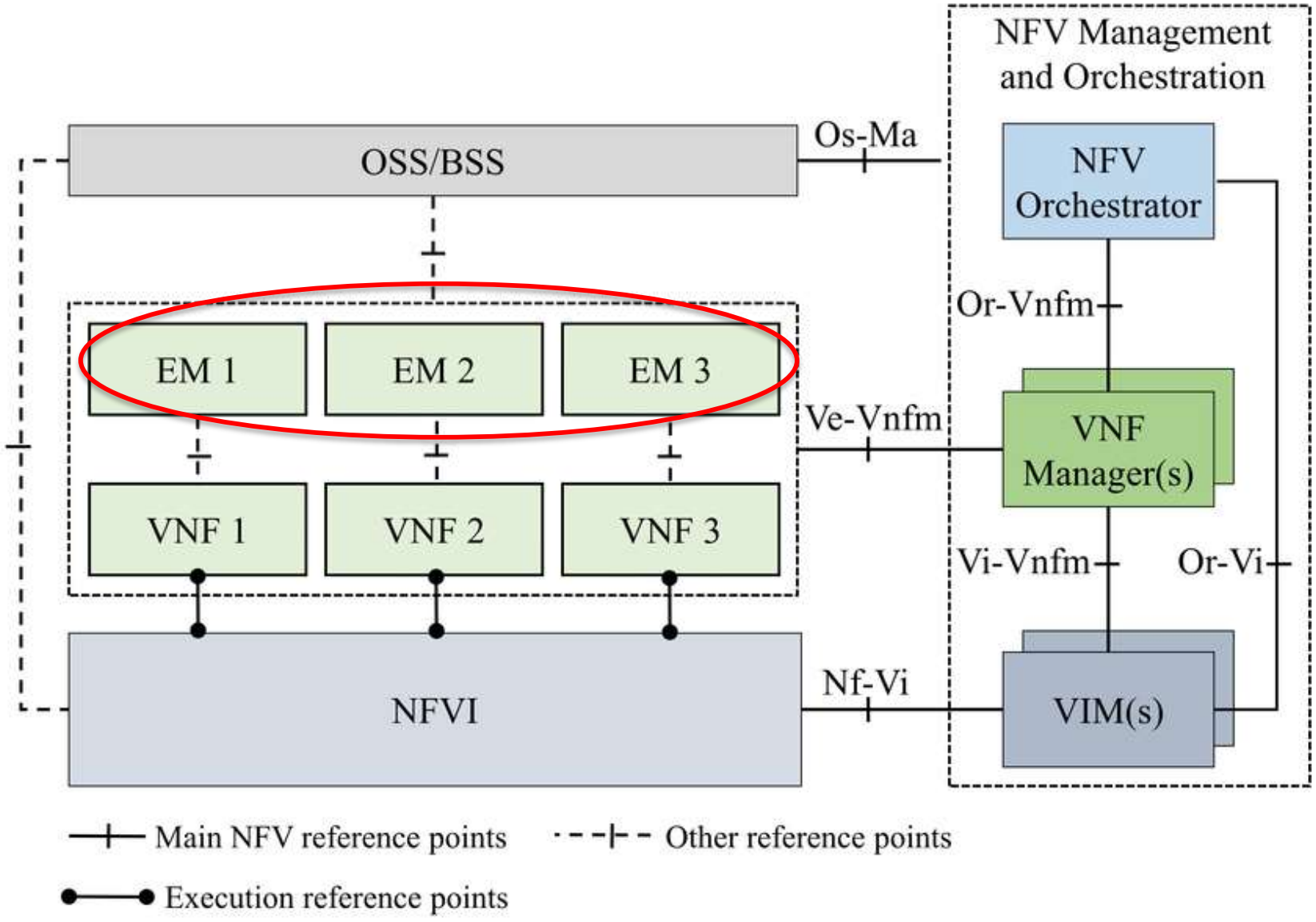
ETSI NFV Architectural Framework

- **BSS example (The Same Home Fiber Internet Customer)**

Let's revisit our customer, "Alex," ordering from "Telecom Giant Inc." We'll now see the crucial role the BSS plays in his journey.

  - **Step 1:** Product Discovery & Shopping Cart (BSS Function)
  - **Step 2:** Customer Creation & Credit Check (BSS Function)
  - **Step 3:** Order Submission & Orchestration (BSS & OSS Handshake)
  - **Step 4:** Billing Cycle Activation (BSS Function)
  - **Step 5:** Invoicing and Payment (BSS Function)
  - **Step 6:** Ongoing Customer Support & Retention (BSS Function)

ETSI NFV Architectural Framework



EMS

ELEMENT MANAGEMENT SYSTEM

ETSI NFV Architectural Framework

- **EMS**
  - Manages specific types of one or more network elements within a telecommunication management network (TMN).
  - Manage functions and capabilities.
  - Communicates upward to higher-level systems of network management (NMS), in order to manage the traffic between itself and other network elements.
- **Key functions**
  - **Device-Level Fault Management:** Receiving real-time alarms and events from a specific device (e.g., "Port 5 Failure," "CPU Overload," "Transmitter Laser Fault").
  - **Device Configuration**: Directly configuring settings on the device (e.g., setting up VLANs, adjusting radio power, provisioning a port speed).
  - **Performance Monitoring**: Collecting detailed performance metrics from the device (e.g., bytes in/out on an interface, optical power levels, error counts, memory utilization).
  - **Software/Firmware Management**: Pushing software updates and patches to the devices it manages.
  - **Log Management**: Collecting and storing logs from the device for audit and debugging purposes.

ETSI NFV Architectural Framework

- **EMS example (Troubleshooting a Network Slowdown in a Specific Area)**

  Let's return to "Telecom Giant Inc." The NOC (Network Operations Center) gets several calls from businesses in a downtown office building complaining that their internet is slow.

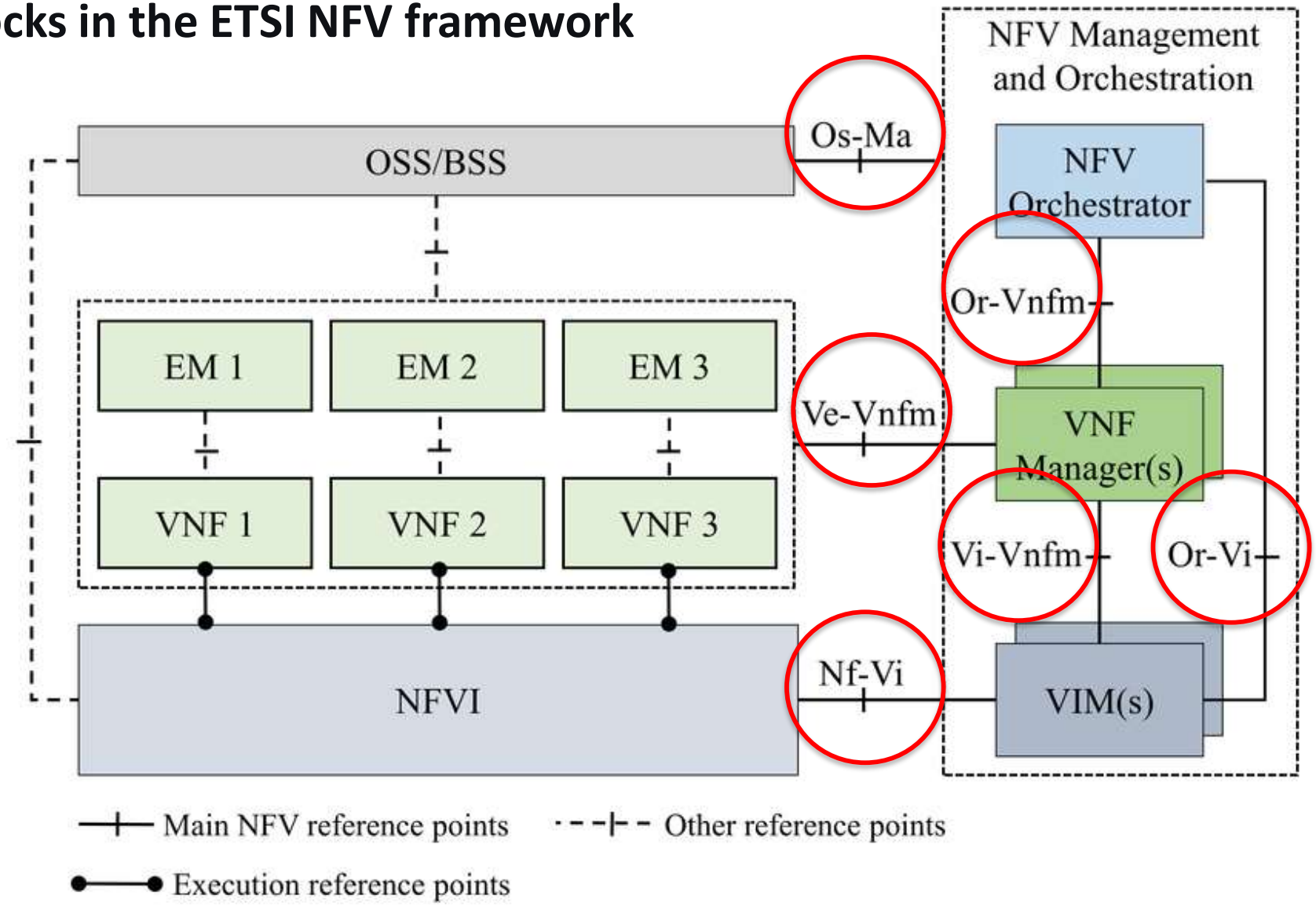
  - **Step 1:** The OSS Raises the Alert
  - **Step 2:** Drilling Down with the EMS
  - **Step 3:** Identifying the Root Cause (EMS Function)
    - Performance Data, Fault Data, and Configuration Data
  - **Step 4:** Taking Action via the EMS
    - Apply temporary Quality of Service (QoS) policy
  - **Step 5:** Proactive Maintenance with the EMS
    - Inventory, Software Upgrade, and Configuration Backup.

ETSI NFV Architectural Framework

Think of it in terms of hierarchy:

- **BSS** manages the *customer and the business*.
- **OSS** manages the *end-to-end services and network-wide operations*.
- **EMS** manages *individual devices or a group of similar devices*.

Standardized interfaces between the functional blocks in the ETSI NFV framework



NFV Reference Points

Orchestrator – VNF Manager (**Or-Vnfm**)

Main Functionalities:

- **VNF Instantiation Request:** The NFVO instructs the VNFM to instantiate (create) a new VNF instance.
- **VNF Lifecycle Management Commands:** The NFVO can request the VNFM to perform actions like **scale** (in/out, up/down), **heal** (e.g., restart a failed VNF component), **update/upgrade**, or **terminate** a VNF.
- **VNF Lifecycle Change Notification:** The VNFM informs the NFVO about changes in the VNF's state (e.g., "VNF instantiation is complete," "Scaling operation failed," "VNF is now terminated").
- **Policy Enforcement:** The NFVO can provide the VNFM with policies related to the VNF's lifecycle (e.g., "Scale out when CPU usage > 80%").

NFV Reference Points

Virtualized Infrastructure Manager – VNF Manager (**Vi-Vnfm**)

Main Functionalities:

- **Resource Allocation for VNF Lifecycle**: When the VNFM needs to instantiate or scale a VNF, it uses this interface to ask the VIM to allocate the actual compute, storage, and network resources.
- **Performance & Fault Information**: The VNFM can query the VIM for performance data (e.g., CPU utilization of the VM hosting the VNF) and fault information (e.g., "The underlying VM has crashed") to make healing and scaling decisions.
- **Virtualized Resource Change Notification**: The VIM can notify the VNFM about events related to the resources a VNF is using (e.g., a hypervisor evacuation).

NFV Reference Points

Orchestrator – Virtualized Infrastructure Manager (**Or-Vi**)

Main Functionalities:

- **Resource Reservation & Allocation:** The NFVO uses this interface to reserve and allocate blocks of resources from the VIM to fulfill Network Services.
- **Global Resource Inventory:** The NFVO queries the VIM to discover the available capacity and inventory of the entire NFVI (e.g., how many total CPU cores are free, what networks are available). This is essential for placement decisions.
- **Infrastructure Performance and Fault Management:** The NFVO receives aggregated performance and fault information about the NFVI from the VIM (e.g., "Datacenter 'A' is over-utilized," "Physical server 'X' has failed").
- **Root Cause Analysis:** By correlating VIM-level faults with service-level issues, the NFVO can determine if a service failure is due to a VNF software bug or an underlying hardware failure.

NFV Reference Points

NFVI-Virtualized Infrastructure Manager (**Nf-Vi**)

Main Functionalities:

- **Hardware Resource Abstraction:** The VIM uses this interface to communicate with the hypervisors (e.g., KVM, VMware) and physical hardware to actually create, delete, and manage virtual machines (VMs) and containers.
- **Physical Resource Inventory:** The VIM discovers the physical compute, storage, and networking hardware via this interface.
- **Hardware-Level Fault and Performance Management:** The hypervisors and hardware drivers send real-time performance metrics (CPU, memory, I/O) and fault alerts (hard disk failure, network link down) up to the VIM through this interface.

NFV Reference Points

Operation Support System (OSS)/Business Support Systems (BSS) – NFV Management and Orchestration (**Os-Ma**)

Main Functionalities:

- **Network Service Lifecycle Management:** The OSS uses this interface to request the NFVO to instantiate, scale, update, or terminate a complete Network Service (a chain of VNFs).
- **Policy Exchange:** The OSS/BSS provides the NFVO with high-level business and service policies (e.g., service level agreements - SLAs).
- **Service-Related Information Reporting:** The NFVO provides the OSS/BSS with data about the Network Service, including its current state, performance data for SLA reporting, and fault information for ticketing.
- **Billing Data:** Usage records for VNFs and Network Services can be sent to the BSS for billing purposes.

NFV Reference Points

VNF/ Element Management System (EMS) – VNF Manager (**Ve-Vnfm**)

Main Functionalities:

- **VNF Configuration:** The VNFM can trigger initial configuration on the VNF after instantiation via the EMS.
- **Software Image Management:** The VNFM can instruct the EMS to update the VNF's software.
- **VNF-Specific Fault and Performance Management:** The EMS reports VNF-specific alarms (e.g., "Firewall policy table is full," "Session limit reached") and performance metrics (e.g., "Packets inspected per second") to the VNFM. The VNFM can then decide to trigger a healing or scaling action.
- **Lifecycle Coordination:** The VNFM informs the EMS of impending lifecycle operations (e.g., "I am about to scale this VNF, prepare yourself").

A Simple Analogy: The Power Grid

Let's use a powerful analogy to tie all the NFV components together. Imagine providing electricity to a city.

- **The NFVI is the entire Power Grid Infrastructure.**
  - The **Physical Layer** is the power plants, coal/gas, high-voltage transmission lines, and electrical substations.
  - The **Virtualization Layer** is the technology that converts various energy sources (coal, gas, solar) into a standardized form (AC electricity) and manages its flow.
- **The VNFs are the Electrical Appliances** in homes and businesses (e.g., a refrigerator, an air conditioner, a factory machine). Each appliance provides a specific function.
- **The VIM is the Local Power Station Manager & Grid Operator.**
  - They control the flow of electricity from the power plants to the neighborhoods. They can switch power sources and reroute energy to

A Simple Analogy: The Power Grid (cont)

- **The VNFM is the Appliance Specialist/Technician.**
  - If your air conditioner (Firewall VNF) breaks, the technician (VNFM) is called to fix or replace it. If you need more cooling, they install a bigger unit (scale up).
- **The NFV Orchestrator is the City Planning Department.**
  - They don't manage individual appliances or power lines. They approve the construction of a new housing development (new Network Service) and ensure that the power grid, water, and roads are coordinated to support it. They work with the Grid Operator (VIM) to ensure capacity is available.

Summary

Module	Real-World Function	Example in the Scenario
OSS/BSS	Handles customer orders, billing, and policy	Customer requests VPN via portal
NFV Orchestrator (NFVO)	Plans and coordinates the full network service	Selects and deploys “Enterprise VPN” blueprint
VNF Manager (VNFM)	Manages the lifecycle of VNFs	Instantiates and configures vRouter, vFirewall, vLoadBalancer
VIM	Allocates compute, storage, and network resources	Deploys VMs and sets up virtual networks
NFVI	Executes VNFs on physical infrastructure	Hosts all VNFs in data centers
EMS	Manages individual VNFs	Configures rules, monitors performance, sends alarms

OSS/BSS — Customer Request and Business Orchestration

- **Technologies Used:**
  - **BSS:** Salesforce CRM, Amdocs, or NetCracker
  - **OSS:** ServiceNow OSS, NetAct
  - **Interface:** *Os-Ma* (REST API / SOAP over HTTPS)
  - **Data formats:** TMF Open APIs (TMF640 – Service Ordering, TMF641 – Service Inventory)

OSS/BSS — Customer Request and Business Orchestration

- **Interface:** *Os-Ma* (REST API / SOAP over HTTPS)

- REST is a style of building web services that uses:
  - **HTTP methods** (GET, POST, PUT, DELETE)
  - **URLs**
  - **JSON format** (sometimes XML)

REST Example:

* Get student with ID 5:

GET https://school-api.com/students/5

* Response (JSON):

```
{
  "id": 5,
  "name": "Sara",
  "grade": 8
}
```

OSS/BSS — Customer Request and Business Orchestration

- **Interface:** *Os-Ma* (REST API / SOAP over HTTPS)
- SOAP is a **strict, formal** protocol for exchanging messages using XML.
  - More Secure than REST
  - Slower and Heavier

SOAP Request Example:

* Get student info:

```
POST https://school-api.com/studentService
Content-Type: text/xml
```

* Body (XML):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:stu="http://school.com/student">
  <soapenv:Header/>
  <soapenv:Body>
    <stu:getStudent>
      <stu:id>5</stu:id>
    </stu:getStudent>
  </soapenv:Body>
</soapenv:Envelope>
```

OSS/BSS — Customer Request and Business Orchestration

- **Interface:** *Os-Ma* (REST API / SOAP over HTTPS)

SOAP Response Example:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <getStudentResponse>
      <id>5</id>
      <name>Sara</name>
      <grade>8</grade>
    </getStudentResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

OSS/BSS — Customer Request and Business Orchestration

- **Data formats:** TMF Open APIs
  - **TMF640 – Service Ordering**: used when a customer or another system wants to place, update, or track a service order.
    - Submit service orders
    - Modify or cancel orders
    - Check order status
    - Attach service characteristics
    - Include related customer and service information
  - **TMF641 – Service Inventory**: returns the list of active, planned, or terminated service instances owned by the customer.
    - Query service inventory
    - Get detailed service instance information
    - Get service hierarchy (parent/child services)
    - Track service activation state
    - View service characteristics

OSS/BSS — Customer Request and Business Orchestration

- **Data formats:** TMF Open APIs

Creating a Service Order (TMF640)

POST /serviceOrder

```
{ "externalId": "SO-2025-001",
  "description": "Fiber broadband installation",
  "requestedStartDate": "2025-11-15T10:00:00Z",
  "orderItem": [
   {
    "id": "1",
    "action": "add",
    "service": {
     "name": "Fiber 100Mbps",
     "serviceSpecification": {
      "id": "fiber_spec_100",
      "href": "/serviceSpecification/fiber_spec_100"},
     "serviceCharacteristic": [
      {
       "name": "Speed",
       "value": "100Mbps"},
      {
       "name": "InstallationType",
       "value": "HomeVisit"
      }]}}]}
```

Request body

```
{
 "id": "7890",
 "state": "acknowledged",
 "orderItem": [
  {
   "id": "1",
   "state": "acknowledged"
  }
 ]
}
```

Response body

OSS/BSS — Customer Request and Business Orchestration

- **Data formats:** TMF Open APIs

Query Service Inventory (TMF641)

GET /service?serviceState=active


```
[{
  "id": "service_001",
  "name": "Fiber 100Mbps",
  "serviceState": "active",
  "serviceSpecification": {
   "id": "fiber_spec_100” },
  "serviceCharacteristic": [
   {
    "name": "Speed",
    "value": "100Mbps” },
   {
    "name": "RouterIncluded",
    "value": "YES"}]},
 {
  "id": "service_002",
  "name": "Mobile Voice",
  "serviceState": "active",
  "serviceCharacteristic": [
   {
    "name": "Minutes",
    "value": "300"}]}]
```

OSS/BSS — Customer Request and Business Orchestration

- Technical Steps:

1- Customer order created in **BSS (Order Management System)** → generates an XML/JSON order message:

```
{
  "serviceType": "5G_Enterprise_VPN",
  "bandwidth": "10Gbps",
  "QoS": "Gold",
  "locations": ["Paris", "London", "Berlin"]
}
```

2 - BSS → OSS via TMF640 API: triggers “Order Fulfillment.”

3 - OSS translates service request into a Network Service Descriptor (NSD) request to the NFV Orchestrator.

4 - Sends this request to NFVO using the **Os-Ma** interface:

```
POST /nfvo/ns-instances

Content-Type: application/json
```

Output: The NFVO now receives a structured service request ready for orchestration.

NFV Orchestrator (NFVO) — Service Orchestration and Automation

*The **NFVO** receives the VPN request and automatically orchestrates the deployment of all required Virtual Network Functions (VNFs).*

- Technical Used:
  - **Platform:** OSM (Open Source MANO) or ONAP (Open Network Automation Platform)
  - **Interfaces:** *Or-Vnfm* (to VNFM), *Or-Vi* (to VIM)
  - **Blueprints:** Network Service Descriptors (YAML-based NSDs)

- Technical Steps:

1- The NFVO reads the **NSD** named enterprise_vpn.yaml:

```
nsd:
  name: Enterprise_VPN_Service
  constituent-vnfd:
    - vnfd-id: vRouter
    - vnfd-id: vFirewall
    - vnfd-id: vLoadBalancer
```

2- Checks available resources via **Or–Vi** API to the **VIM** (e.g., OpenStack Keystone/Nova).

NFV Orchestrator (NFVO) — Service Orchestration and Automation

- Technical Steps:

3- Sends VNF instantiation requests to VNFM via Or–Vnfm:

```
POST /vnfm/vnfs
{
  "vnfd_id": "vFirewall",
  "flavor": "medium",
  "location": "London"
}
```

4- NFVO coordinates all VNFs, chaining them (Service Function Chaining) through SDN controllers (e.g., OpenDaylight).

Output: A network service graph is instantiated — VPN tunnel + virtual firewall + load balancer.

NFV Manager (NFVM) — Lifecycle and Policy Control

*The **VNFM** manages the lifecycle of the deployed VNFs — creating, configuring, monitoring, and scaling them.*

- Technical Used:
  - **VNFM Platform:** Nokia CloudBand, Huawei CloudMSE, or OSM VNFM
  - **Interface:** *Ve–Vnfm* (to EMS), *Vi–Vnfm* (to VIM)
  - **Protocols:** NETCONF/YANG, RESTCONF, Ansible automation scripts
- Technical Steps:

1- VNFM receives instantiateVNFRequest from NFVO.

2- Requests VM allocation from **VIM**:

```
POST /vims/resources
{
  "vCPU": 8, "vRAM": "16GB", "Storage": "200GB"
}
```

3- Once VM is up, VNFM boots the **VNF image** (vFirewall.qcow2).

NFV Manager (NFVM) — Lifecycle and Policy Control

- Technical Steps:

4- Connects to the **VNF's EMS** via **NETCONF**:

```
<edit-config>
 <firewall-policy>
  <rule id="1" action="allow" src="10.0.0.0/8" dst="172.16.0.0/12"/>
 </firewall-policy>
</edit-config>
```

5- Enforces policies (from NFVO): e.g., *"Scale out when CPU > 80%"*.

6- Monitors alarms from the EMS (cpuUsage, linkFailure) and sends status updates back to NFVO.

Output: All VNFs are deployed, configured, and monitored automatically.

Virtualized Infrastructure Manager (VIM) — Resource Management

*The **VIM** (OpenStack) provisions and manages compute, storage, and network resources across data centers.*

- Technologies Used:
  - **Platform:** OpenStack (Nova, Neutron, Cinder, Glance)
  - **Interfaces:** *Nf–Vi* (to NFVI), *Or–Vi* (to NFVO), *Vi–Vnfm* (to VNFM)
  - **Protocols:** OpenStack APIs (REST), gRPC for telemetry

- Technical Steps:

1- NFVO sends resource request to VIM:

```
POST /v2.1/servers
{
  "name": "vFirewall-01",
  "imageRef": "vFirewall.qcow2",
  "flavorRef": "m1.medium",
  "networks": [{"uuid": "vpn-net"}]
}
```

Virtualized Infrastructure Manager (VIM) — Resource Management

- Technical Steps:

2- VIM allocates the virtual machine using **Nova**.

3- Configures virtual networks with **Neutron** (creates VLANs/VxLANs).

4- Attaches storage via **Cinder** and retrieves image via **Glance**.

5- Monitors VM health using **Ceilometer** or **Monasca** telemetry.

Output: All VNFs are running on provisioned infrastructure with full network connectivity.

NFV Infrastructure (NFVI) — Execution Environment

*The **NFVI** provides the virtualized physical environment for all VNFs.*

- Technologies Used:
  - **Compute:** Dell EMC PowerEdge servers with Intel Xeon CPUs
  - **Storage:** SAN/NAS
  - **Network:** Cisco Nexus switches, Mellanox NICs
  - **Virtualization Layer:** KVM hypervisor with SR-IOV and DPDK for high packet throughput

- Technical Steps:

1. Physical servers host multiple KVM instances — one per VNF.
2. DPDK accelerates packet forwarding for VNFs like vFirewall and vRouter.
3. SR-IOV enables direct NIC access, reducing latency.
4. Each VNF runs in isolation with dedicated vCPUs and vNICs.
5. Hardware metrics (CPU, RAM, I/O) are collected via Prometheus exporters and sent to VIM for resource management.

Output: VNFs execute efficiently with carrier-grade performance and isolation.

Element Management System (EMS) — Device-Level Management

*Each **VNF** (router, firewall, load balancer) has an EMS that manages its local operations.*

- Technologies Used:
  - **EMS Software:** FortiManager (for vFirewall), Cisco Prime (for vRouter), HAProxy Manager (for vLoadBalancer)
  - **Protocol:** SNMP, NETCONF, Syslog
  - **Interface:** *Ve–Vnfm*

- Technical Steps:

1. EMS collects telemetry data:
   - CPU/memory usage
   - Interface traffic (ifInOctets, ifOutOctets)
   - Error counters
2. Sends alarms (e.g., link failure) to VNFM via Syslog or SNMP traps.
3. Receives configuration commands from VNFM (e.g., adjust firewall ACLs).
4. Performs firmware upgrades on VNFs when instructed:

```
PUT /firmware/update
{
  "image": "vFirewall_v2.3.1.img"
}
```

Output: EMS ensures each virtual function runs optimally and reports real-time status for automation.