A REFERENCE ARCHITECTURE FOR NETWORK FUNCTION VIRTUALIZATION

by

Ahmed M. Alwakeel

A Dissertation Submitted to the Faculty of

The College of Engineering and Computer Science

In Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

Florida Atlantic University

Boca Raton, FL

May 2020

A REFERENCE ARCHITECTURE FOR NETWORK FUNCTION VIRTUALIZATION

by

Ahmed M. Alwakeel

This dissertation was prepared under the direction of the candidate's dissertation advisor, Dr. Eduardo Fernandez, Department of Computer & Electrical Engineering and Computer Science, and has been approved by the members of the supervisory committee. It was submitted to the faculty of the College of Engineering and Computer Science and was accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

SUPERVISORY COMMITTEE:

Eduardo B. Fernandez, Ph.D., P E.
Dissertation Advisor

Mohammad Ilyas, Ph.D., P.E.

Imadeldin Mahgoub, Ph.D.

Shihong Huang, Ph.D.

Hanqi Zhuang, Ph.D.
Interim Chair, Department of Computer and Electrical Engineering and Computer Science

Stella N. Batalama, Ph.D.
Dean, The College of Engineering and Computer Science

Robert W. Stackman Jr., Ph.D.
Dean, Graduate College

MARCH 05, 2020
Date

ACKNOWLEDGMENTS

This work would not have completed without the help of my advisor professor Eduardo B. Fernandez. Mentoring, support ,and guidance helped me and guided me toward completing this dissertation. I sincerely thank Dr. Fernandez for representing me in Asia PLoP 2018 conference in Tokyo, and in the 26th PLoP 2019 in Canada as well as in UK for ARES 2019. I would also like to express sincere gratitude to my committee members, Dr. Mohammad Ilyas, Dr. Imadeldin Mahgoub and Dr. Shihong Huang for all of their valuable and constructive feedback during my research. I thank the members of Secure Systems Research Group for their comments and feedback during our weekly meetings. Finally, I want to express deep appreciation for my family and close friends for all of their support and encouragement during my Ph.D. This work would not be possible without the support I had from my parents and my lovely wife.

ABSTRACT

Author: Ahmed M. Alwakeel

Title A Reference Architecture for Network Function Virtualization

Institution: Florida Atlantic University

Dissertation Advisor: Dr. Eduardo B. Fernandez

Degree: Doctor of Philosophy

Year: 2020

Cloud computing has provided many services to potential consumers, one of these services being the provision of network functions using virtualization. Network Function Virtualization is a new technology that aims to improve the way we consume network services. Legacy networking solutions are different because consumers must buy and install various hardware equipment. In NFV, networks are provided to users as a software as a service (SaaS). Implementing NFV comes with many benefits, including faster module development for network functions, more rapid deployment, enhancement of the network on cloud infrastructures, and lowering the overall cost of having a network system. All these benefits can be achieved in NFV by turning physical network functions into Virtual Network Functions (VNFs). However, since this technology is still a new network paradigm, integrating this virtual environment into a legacy environment or even moving all together into NFV reflects on the complexity of adopting the NFV system. Also, a network service could be composed of several components that are provided by different service providers; this also increases the complexity and heterogeneity of the system. We apply abstract architectural modeling to describe and analyze the NFV architecture. We use architectural patterns to build a flexible NFV architecture to build a Reference Architecture (RA) for NFV, that describe the system and how it works. RAs are proven to be a powerful solution to abstract complex systems that lacks semantics. Having an RA for NFV helps us understand the system and how it functions. It also helps us to expose the possible vulnerabilities that may lead to threats toward the system. In the future, this RA could be enhanced into SRA by adding misuse and security patterns for it to cover potential threats and vulnerabilities in the system. Our audiences are system designers, system architects, and security professionals who are interested in building a secure NFV system.

Dedicated to my Parents, Mahmoud and Fatima. My wife Raheeq and my daughter Elian. My brothers Mohammed, Maher and Majdei. My sisters Maha, Manal and Marwah.

A REFERENCE ARCHITECTURE FOR NETWORK FUNCTION VIRTUALIZATION

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

1.1 OVERVIEW

Network Function Virtualization (NFV) is a recent technology to provide network solutions through completely decoupling network functions from the physical devices that provide them. Unlike legacy networks, NFV functions are performed in virtual machines (VM) created by the cloud infrastructure. It has become desirable for network operators, developers, and even regular consumers due to its ability to quickly develop new services without hardware restrictions. In other words, virtualization is broadly described as the process of separating resources for services from the physical delivery of that service [1].

NFV promises the following benefits [2]:

1. Independence: the software is no longer integrated with hardware. As a result, their evolution will be independent of each other [3].
2. Flexibility and speed: the decoupling of software from hardware helps to reassign and share the same infrastructure resources, which allows for the performing of different functions at various times. As a result, the deployment of network functions and their connections become faster and more flexible [4]

3. Scalability: in conventional legacy network systems, telecoms must be up to date with new network standards and requirements, which requires time, planning, and money. However, in NFV, the decoupling of software from hardware allows for dynamically scaling the actual performance of virtualized network functions with finer granularity and minimal effort [5].
4. Reduced energy consumption: with the ability to scale up or reduce the resources, Telcos will be able to reduce the OPEX needed to run network devices. Similarly, energy consumption at the customer end will be reduced significantly due to not having to install dedicated hardware to deliver network functions [6],[7].

Although NFV has many benefits, it comes with many new challenges. A common concern for NFV users is the ability to adapt to the new virtual system that NFV provides. In NFV functions, which are provided through software and rely on virtualization, security challenges are the concerns of anyone who uses a cloud-based system. The concern is that any virtual function could be compromised and controlled by an external entity. The entire system could be potentially destroyed or severely damaged, causing users to have unstable delivery of service or even entirely losing access to the network.

1.2 MOTIVATION

In 2012 The European Telecommunication Standards Institute (ETSI) held a meeting in Germany with several other telecom network leaders in the industry such as AT&T, BT, Orange, Telefonia, Verizon, and others where the concept of

NFV was introduced for the first time, and the standards for it were defined. Although ETSI defined a framework for NFV systems in [8], their framework is vague and lacks semantics, making implementing complex systems like NFV a challenging and time-consuming task.

Moreover, the NFV framework contains different stakeholders and several architectural components, including the legacy network architecture that could be part of the cloud ecosystem [9], which makes the NFV architecture very complex. Also, a network service could be composed of several components that are provided by different vendors; this also increases the complexity and heterogeneity of the system.

An effective way to study and analyze such complex systems is through abstraction. Many researchers have used abstraction to represent complex systems. In [9], the authors used abstraction to represent the cloud ecosystem and its security. In [10], the authors used abstraction to show how to design a mobile shared workspace, while in [11], the authors applied it to build a security reference architecture for cloud systems. Here, we use architectural models to provide a better understanding of NFV architecture and its main architectural components, and the interactions among these components. Understanding how different components interact with each other will help us to achieve our goal, which is to build a Reference Architecture (RA) for NFV. Understanding how a system works and interacts with other systems will help us in designing it, and also help us to ensure the use of NFV system to its full potential.

1.3 CONTRIBUTIONS

The following contributions were made in this dissertation:

1. As part of the RA creation process, we defined the main use cases and stakeholders of the NFV system as well as showing how stakeholders preform fundamental tasks in NFV system.

2. We created four architectural patterns for NFV system that describe the following:
   - A pattern of Management and Orchestration (MANO) in NFV that shows how different management tasks of the system are performed. This pattern also shows how MANO unit handles users' requests.
   - A pattern of Network Function Virtualization Infrastructure (NFVI) that shows the infrastructure layer of NFV system and how different physical components contribute to creating the system.
   - A pattern of Virtual Machine Environment (VME), which shows how physical components are transferred into virtual components to achieve virtualized network functions.
   - A pattern of Virtual Network Function (VNF), which shows how virtual functions are created and managed. This pattern also shows the lifecycle of VNF in the NFV system.

3. We developed a Reference Architecture (RA) that describes how the different components of the system interact with each other and with

external systems such as virtual data centers. We achieved this by combining the four patterns previously mentioned as well as adding additional components to it to serve the system.

1.4 DISSERTATION ORGANIZATION

This dissertation is divided into eight different chapters, most of the topics covered in this dissertation have already been published or submitted for publication so there may be some replication in the content. Chapter one gives a brief introduction to the topic of the dissertation as well as the motivation to write it and the achievements we accomplished during the time of preparation of this dissertation. In Chapter Two, a background review of the topic of NFV is given with concentrating on NFV architectural framework by ETSI [2]. A brief background on some architectural modeling techniques such as patterns and reference architectures are discussed, and how they impact such systems. A presentation of NFV UML Use Cases is given in Chapter Three. The reference architectural model is presented in chapter four; all the patterns related to the NFV system that we designed will be presented in this chapter. Chapter Five will cover the validation aspect of our models. In Chapter Six, we discuss analysis of activities for use cases to illustrate possible security threats. Chapter Seven will go over related work by other researchers in the same field. Finally, in Chapter Eight, a conclusion for this dissertation will be given as well as some ideas for possible future work.

2 BACKGROUND

2.1 NETWORK FUNCTION VIRTUALIZATION (NFV)

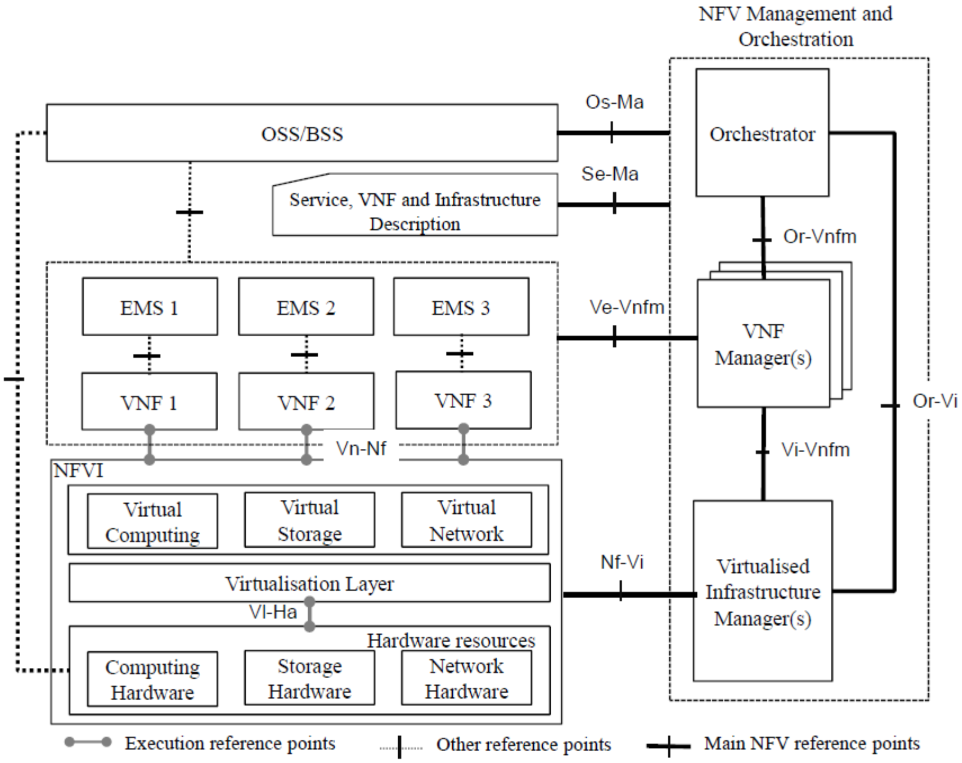


Figure 2.1 ETSI NFV framework

- *The Management and Orchestration (MANO) unit:* could be considered as the heart of the NFV system. This unit takes care of the management and orchestration aspects of the entire set of VNFs. MANO contains three main parts.

  - *Virtualized Infrastructure Manager (VIM)* is the first part of MANO that controls and manages the interaction of the VNF and Network Function Virtualization Infrastructure (NFVI) as well as computes and stores network resources; It also has the necessary deployment and monitoring capability for the virtualization layer.
  - *Virtual Network Function (VNF) manager* is the second part of MANO, which is responsible for managing the lifecycle of VNF instances and initializing, updating, querying, scaling and terminating these instances.
  - *Orchestrator* is the last part of MANO that manages the lifecycle of network services that includes policy management, instantiation, performance management and Key Performance Indicator (KPI), which is a measurable value used to evaluate how effectively a company is achieving key business objectives. Moreover, VNF descriptor is one of the components of NFV-MANO which takes care of giving descriptions regarding different VNF deployment and operational requirements [2]. These three components, as well as other components in the NFV architecture, interact with each other through reference points.

- *Network Function Virtualization Infrastructure (NFVI)*: is the foundation platform of NFV that contains the hardware resources as well as the virtual instantiations that build up the infrastructure on which VNFs are deployed, managed and executed. The NFVI can be part of the cloud Infrastructure-as-a-Service (IaaS), which cloud providers use to create Virtual Data Centers (VDCs) [12], containing all the necessary virtualized computing, storage, and networking to run as a physical data center. These VDCs are provided to NFV providers, which in turn use them to provide network services to NFV consumers. The resources of a VDC provided to a NFV provider should be isolated from other providers; such isolation enables NFV providers to securely share the same cloud infrastructure. In terms of NFV services, the VNFs are deployed over virtual machines (VMs) within a VDC. The NFVI consists of three main components. First, the hardware resources that contain compute facilities, which are normally Commercial-Off-The-Shelf (COTS) appliances, the storage hardware could be in form of direct-attached hard disks, external storage-area-networks (SAN) or network-attached storage (NAS) [13],and the network hardware may consist of switches/routers that provide processing and connectivity capabilities to VNFs through the virtualization layer. Second, the virtualization layer, which lies on top of the hardware resources layer and contains the Virtual Machine Monitor (VMM) (also known as the hypervisor), which has three main roles: decouples the virtual resources from the underlying physical resources, provides isolation among VMs, and emulates the hardware resources [14]. Third, the virtual infrastructure lies on top of the virtualization layer and contains the virtualized resources, which are abstractions of the hardware resources; these abstractions are virtual machines, diagram with some vague semantics which, as indicated earlier, is not precise enough to be the basis of security analysis.

- *Virtual Network Functions (VNFs)*: are software packages that represent the implementation of the legacy non-virtual network functions that could be deployed on the NFVI. A single VNF could be composed of several internal components, such as packet data network gateways (PGW), residential gateways, firewalls, etc. In order to reduce management and complexity in deploying it [15]. On the other hand, a VNF could also contain only one component in order to increase scalability and reusability, as well as to have a faster response due to its simplicity; keeping in mind that a single VNF could be deployed and distributed across several VMs [15]. Normally, the virtual network services provided by TSPs are composed of several VNF based on the users' needs.

- *Element Management System (EMS):* it takes care of performing the typical management functionality for each VNF connected to it, i.e., it performs FCAPS (Fault, Configuration, Accounting, Performance and Security management). There could be one EMS for every VNF, or a single EMS could manage multiple VNFs.

- *Operation Support System and Business Support System (OSS/BSS):* is responsible for various functions related to the NFV service such as billing, support ticketing, helpdesk support as well as service fulfillment and assurance.

- *Service, VNF and Infrastructure descriptions:* provide information regarding the VNF deployment template, VNF forwarding graph, service-related information and NFV infrastructure information [2].

Moreover, ETSI has defined some reference points [16] to connect the components of the system as seen in figure 2.1 these are the main reference points of the framework:

- Virtualization Layer - Hardware Resources (VI-Ha): it interfaces the virtualization layer to hardware resources to create an execution environment for VNFs and collect relevant hardware resource state information for managing the VNFs.
- VNF - NFV Infrastructure (Vn-Nf): it presents the execution environment provided by the NFVI to the VNF, and guarantees the hardware-independent lifecycle, performance and portability requirement for the VNF. It also interconnects VNF components to storage as well as to other components within the same or another VNF.
- Orchestrator - VNF Manager (Or-Vnfm): it is used for exchanges between the orchestrator and the VNF manager to support resource requests, such as authorization, validation and allocation to the VNF manager, send

configuration information to the VNF manager, and also used to collect state information of the VNF.

- Virtualized Infrastructure Manager - VNF Manager (Vi-Vnfm): it is used to send resource allocation requests by the VNF manager, and by the VIM to report the availability and status of infrastructure resources.
- Orchestrator - Virtualized Infrastructure Manager (Or-Vi): this reference point is used for exchanges between the orchestrator and the VIM to request resource reservation and/or allocation by the orchestrator, as well as used by the VIM to report the availability and status of infrastructure resources.
- NFVI - Virtualized Infrastructure Manager (Nf-Vi): this reference point interfaces between NFV Infrastructure and the VIM of MANO; it is used by the VIM to manage the hardware resources and virtualization layer of the NFVI, manage resource allocation requests, and forward virtualized resource state information.
- VNF/EMS - VNF Manager (Ve-Vnfm): this interface is used for exchanges between the VNF and EMS with the VNF manager to support requests for VNF lifecycle management, exchanging configuration and state information.
- OSS/BSS - NFV MANO (Os-Ma): it is used for exchanges between the OSS/BSS and the NFV MANO to support several functions such as, requests for network service lifecycle management, VNF lifecycle

management, forwarding of NFV related state information, policy management exchanges, and data analytics exchanges.

- Service, VNF, and Infrastructure Description - NFV MANO (Se-Ma): the service, VNF, infrastructure description uses this reference point to provide information regarding the VNF deployment template, VNF forwarding graph, service-related information, and NFV infrastructure information models, which are used by the MANO.

2.2 ARCHITECTURAL MODELING

We discuss now some architectural modeling techniques that could be used to achieve our goal. Architectural models are an effective way to represent systems and describe their components as well as capturing design decisions. We use these models to build our RA.

2.2.1 PATTERNS

A Pattern can be defined as a solution to a recurrent problem in a specific context [17]. There are several types of software patterns such as design and architectural patterns, which are used to build flexible and extensible systems; security patterns are used to build secure systems by describing the way to control their threats, fix their vulnerabilities, and provide security attributes [18]. Misuse patterns are used to describe how attacks are performed from the point of view of an attacker [16]; they also define the environment where the attack is performed, what security mechanisms are needed as countermeasures to stop it, and how to find forensic information to trace the attack once it happens.

In this work, we use patterns to represent the main components of the NFV system. Patterns are powerful methods of presenting a detailed solution not only for software but also hardware and physical components that together create ecosystems. They are usually described templates with predefined sections. In our work, we use the POSA (Pattern-Oriented Software Architecture) template [19]. The descriptions of patterns may include both UML modeling techniques and formal languages descriptions. The sections of the POSA template include:

- *Intent*: this section describes a summary of the solution.
- *Context*: defines the environment where the pattern will be applied. Sometimes relevant characteristics of the context are discussed here as well.
- *Problem*: here, we give a description of what happens before we have a solution. The forces that affect the possible solution are listed here, as well.
- *Solution*: in this section, the suggested solution is presented. Usually, one or more UML diagrams are used to explain the solution.
- *Implementation*: in this section, some details about implementing the solution of the pattern are given. Real-life examples may be also presented in this section.
- *Consequences*: in this section, the advantages and liabilities of implementing the solution are discussed.
- *Related Patterns*: in this section, a list of other patterns related to the proposed pattern are listed.

2.2.2 REFERENCE ARCHITECTURE

A Reference Architecture (RA) is an abstract of some architecture related to one or more domains without implementation aspects [20], [21], [22]. An RA should illustrate the fundamental concepts of a system units and the interactions among these units as an architectural solution in particular domain. Several features make RAs powerful tools to represent systems. Some of RAs features are configurability, extendibility, and reusability [21]. In addition to class and sequence diagrams, an RA may include a set of use cases (UC), and a set of Roles (R) corresponding to its stakeholders (actors) [23]. Types of RAs include those for the technology domain (describe platforms and other design artifacts [20], application domain (describe different kinds of applications), and problem domain (similar to domain models, but oriented to software). RA stakeholders include groups, individuals, organizations, and systems that have an interest in the system and affect the design and development of the system [21]. An RA can be evolved into a Security Reference Architecture (SRA) by adding security patterns to neutralize its identified threats [21],[24],[25].

3 THE MAIN UML USE CASES OF NETWORK FUNCTION VIRTUALIZATION

3.1 INTRODUCTION

In this chapter, we take the first step toward building a Reference Architecture for the NFV system, which is defining the Unified Modeling Language (UML) use cases for NFV as well as identify their primary actors. Section 3.2 includes UML use cases of the system where a full written description of the primary use cases is given. Section 3.3 describes stakeholders who interact with the system and which role they play. Finally, in section 3.4, we provide a further description of the selected use cases that didn't appear in the patterns in Chapter 4. These use cases will be described using UML sequence diagrams.

3.2 MAIN USE CASES FOR NFV

In this section, we first give a general overview of the system functions using UML use case diagrams, as shown in Figures 3.1 and 3.2. A use case model is an analysis methodology to clarify, identify, and organize the main activities of the system and which actor interacts with which activity. It also describes how the system should respond under various conditions to a request from the actors [26]. An actor represents a user or automated system that may interact with the system. Generally, an actor is a role rather than being a specific person; an actor can be distinguished through its tasks, and a single actor could be associated with one or more use cases and vice versa. The next two subsections will present the main actors of the system and the primary use cases of the system.

3.3 ACTORS OF THE SYSTEM

I. **Cloud Service Provider (Cloud SP):** provides the cloud services that act as a host for the NFV service. The Cloud SP is responsible for setting up Service Level Agreements (SLA) with the Telco. The cloud SP may have other customers not involved in NFV services.

II. **NFV Provider (NFVP):** is the telecommunication company (Telco) who provides complete networks, and is responsible for setting up Service Level Agreements (SLA) with its consumers.

III. **NFVP Administrator:** is the admin who works in the Telco, and handles the provision of network functions and resources, monitoring the SLA, etc.

IV. **NFVP Designer:** works in Telco and responsible for designing and building the network service packages and functions that are provided to consumers.

V. **NFVP Operator:** works in Telco and responsible for the operation of the VNFs, monitoring, troubleshooting, testing, and implementing the network services, and takes care of consumers requests.

VI. **Consumer:** is a person who receives network services from the NFVP.

VII. **Consumer Administrator (Con-Admin):** is the admin who works in the consumer company, and responsible for several managerial tasks such as monitoring the SLA between the company and the NFVP.

VIII. **Consumer Operator (Con-Op):** is the network operator who works in the consumer company, and responsible for handling the technical tasks.

IX. **NFV management and orchestration (MANO):** is a unit in charge of automatically controlling and managing the resources of the service as well as the interactions of VNFs with the consumers. Furthermore, it manages business aspects of the system such as billing and payment.

X. **Virtual Machine Environment (VME):** is a unit responsible for creating and managing all the resources related to virtual machines in the system.

3.4 USE CASES (UCs) OF THE SYSTEM

UC1- Establish Cloud SLA: Set up an SLA between the Cloud SP and the NFVP Administrator, which governs how cloud services are delivered and managed, and states the level of availability, performance, service continuity, as well as measurable target values characterizing the levels of services.

Actors: Cloud SP, NFVP Administrator.

UC2- Create a VDC: The NFVP, represented by the NFVP Operator, leases a virtual infrastructure from the cloud SP to provide network services to its Consumers.

Actors: Cloud SP, NFVP Operator.

UC3- Establish network SLA: Set up an SLA between the NFVP Administrator and the Consumer Administrator (Con-Admin), which governs how communication services are delivered and managed, and states the level of availability, performance, service continuity, security, as well as measurable target values characterizing the levels of services.

Actors: NFVP Administrator, Con-Admin.

UC4- Open Account: The Con-Op opens an account in order to use the network services provided by the NFVP.

Actors: Con-Op, NFVP Operator.

UC5- Close Account: The Con-Op can close an account if the company he is working in does not need the account anymore or the account can be closed directly by the NFVP Operator in case the company violates terms of service or even for security reasons.

Actors: Con-Op, NFVP Operator.

UC6- Create Network Package: The NFVP Designer creates and designs network graphs, or even single network functions, that can be chosen and used by Consumers.

Actors: NFVP Designer.

UC7- Monitor Network: MANO monitors the available and allocated resources, the status of the service, any unauthorized or suspicious activity, and faults that may impact the service as a part or whole.

Actors: MANO.

UC8- Bill for service: The MANO issues bill for the network services provided to the Consumers.

Actors: MANO, Con-Op.

UC9- Pay Bill for Service: The Con-Op pays bills for the used services. The payment notice is sent to the MANO.

Actors: Con-Op, MANO.

UC10- Request Network: The Con-Op requests a network service; the request is received by the NFVP Operator.

Actors: Con-Op, NFVP Operator.

UC11- Request a VNF: The Con-Op requests a VNF; a VNF could be a single network function such as vFirewall, or it could contain a set of functions. The request is received by the NFVP Operator.

Actors: Con-Op, NFVP Operator.

UC12- Request Modify Network: The Con-Op requests to modify a network they are using, such as: optimizing, configuring, scaling the network, add a vFirewall to increase its security, etc.

Actors: Con-Op, NFVP Operator.

UC13- Request Modify VNF: The Con-Op requests to modify a VNF, by adding or removing a virtual function, such as adding a load balancer to the network.

Actors: Con-Op, NFVP Operator.

UC14- Request Modify VNFC: The Con-Op requests to modify a VNF, such as increase the number of ports on a virtual router.

Actors: Con-Op, NFVP Operator.

UC15- Request Stop VNF: The Con-Op requests to stop a VNF.

Actors: Con-Op, NFVP Operator.

UC16- Pause VNF: The Con-Op requests to pause a VNF for specific time.

Actors: Con-Op, NFVP Operator.

UC17- Terminate VNF: The Con-Op requests to terminate a VNF.

Actors: Con-Op, NFVP Operator.

UC18- Request Terminate Network: The Con-Op requests to terminate the whole network service either because the Consumer requested it or because the Consumer failed to pay for the intended service, or even for security reasons.

Actors: Con-Op, NFVP Operator.

UC19- Request Reliability Level: the Consumer wants to ensure a degree of reliability by requesting a reliability level (high, medium, or low level) in his network service. The request is optional and done by the Con-Op and sent to the MANO.
Actors: Con-Op, MANO.

UC20- Consume Network: The Consumer uses the network service provided by the NFVP. Once the Consumer starts using the service, the MANO starts metering its usage in order to bill the Consumer for the provided network services.
Actors: Consumer, MANO.

UC21- Collect and Forward Consumers' Requests: The MANO collects resource allocation, state information, and other several requests from the Consumers, and forwards them to the designated units.
Actors: MANO

UC22- Modify Resources: The VME includes the hypervisor, which is responsible for managing the VMs and modifying the virtual resources necessary to comply with the Consumers modification requests.
Actors: VME.

UC23- Manage Virtual Resources: The hypervisor in the VME manages the virtual resources, such as: virtual computing, storage, and network resources for the NFV infrastructure. The MANO reports the allocation and status information to and from the VME.

Actors: VME, MANO.

UC24- Monitor VM: The MANO monitors the VMs’ performance and checks its status, as well as detects any unusual activity on it, such as a VM monopolizing resources, and trigger alerts in case of a problem facing a VM.

Actors: VME, MANO.

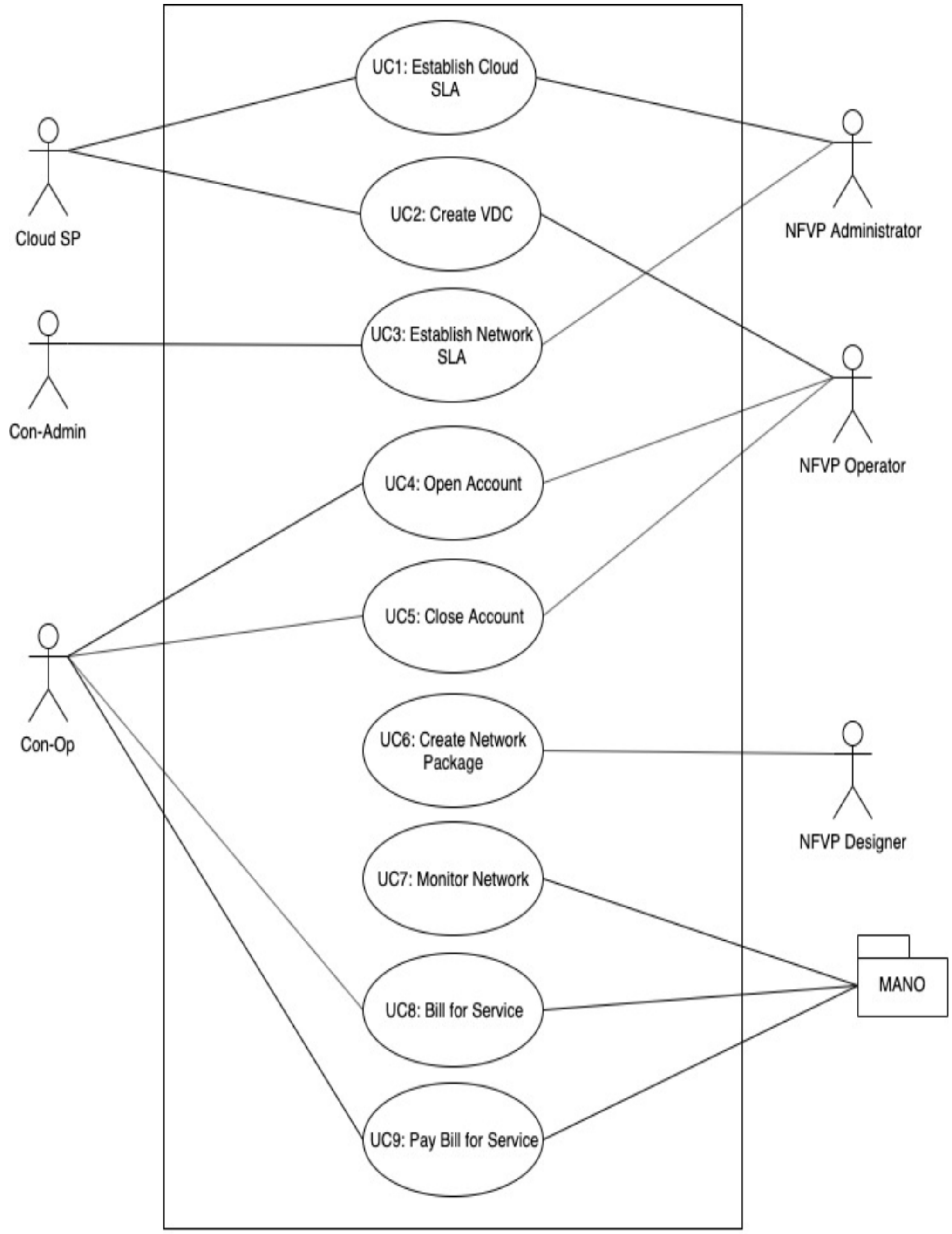


Figure 3.1 Use Cases (UCs) for NFV Architecture Part 1

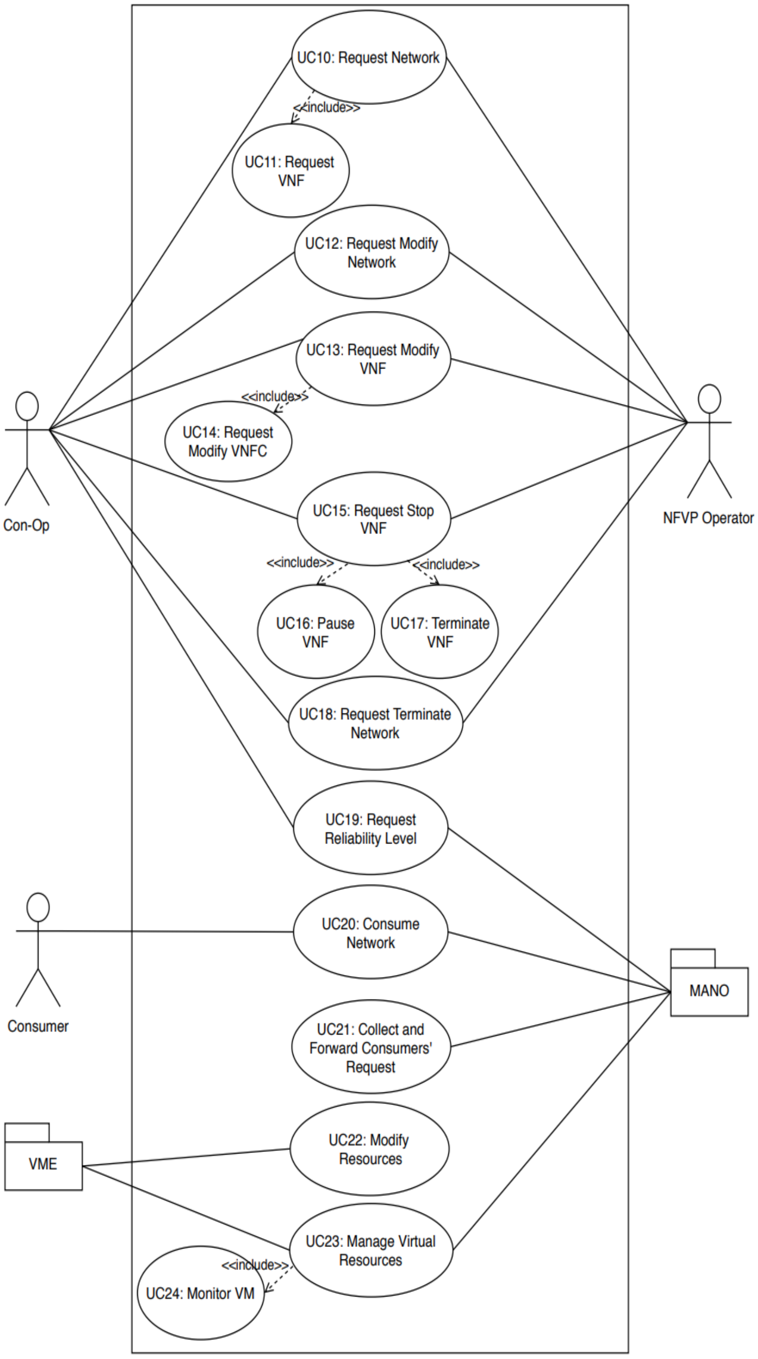


Figure 3.2 Use Cases (UCs) for NFV Architecture Part 2

3.5 DESCRIPTIONS OF NFV USE CASES

In this subsection, we will cover in detail selected use cases of the NFV system. The use cases in this subsection are general use cases related to the system, and they were not described in any of the patterns we developed in Chapter 4. Therefore, we will give some details to them in this chapter.

UC3- Establish Network SLA: this scenario shows the process of establishing an SLA for network service (figure 3.3).

Summary: the NFVP Administrator generates a new SLA and forwards it to the Con-Admin. If they both agree on the SLA, the Con-Admin confirms acceptance.

Actors: NFVP Administrator.

Precondition: the Con-Admin has an account.

Description:

1. The NFVP Administrator generates an NFV SLA.
2. NFVP Administrator sends the SLA to the portal.
3. The portal forwards the SLA to the Con-Admin.
4. The Con-Admin inspects the received SLA.
5. The Con-Admin inspects SLA, accepts it and sends its acceptance through the portal.
6. The portal forwards SLA acceptance to the NFVP Administrator, which registers it.

Exceptions: in case the two parties don’t agree they need to negotiate a new or modified SLA.

Postcondition: the SLA is established between the two parties.

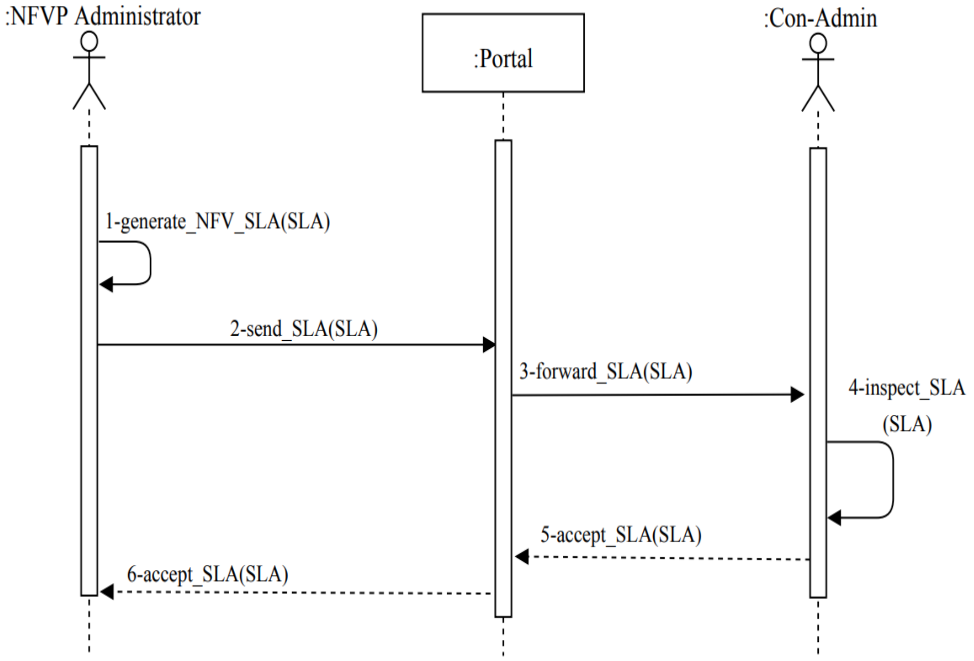


Figure 3.3 Sequence Diagram for the UC3 Establish Network SLA

UC8- Bill for service: The MANO unit handles the billing for the services provided to Consumers.

Summary: the MANO calculates the cost of network services provided for Consumers, and sends the bill to the Con-Op.

Actor: MANO

Precondition: the Consumer is subscribed to a network service and has used it for some amount of time, and it has reached to the next billing cycle.

Description:

1. MANO retrieves Consumer_ID, service usage time and date.
2. MANO calculates usage for the service provided to Consumers based on the service start/end time and date, as well as the Consumer ID.
3. MANO generates the bill based on the calculation.

4. MANO sends the bill to Portal.
5. The Portal sends and shows the bill to Con-Op.
6. Postcondition: the Con-Op received a bill for the used services.

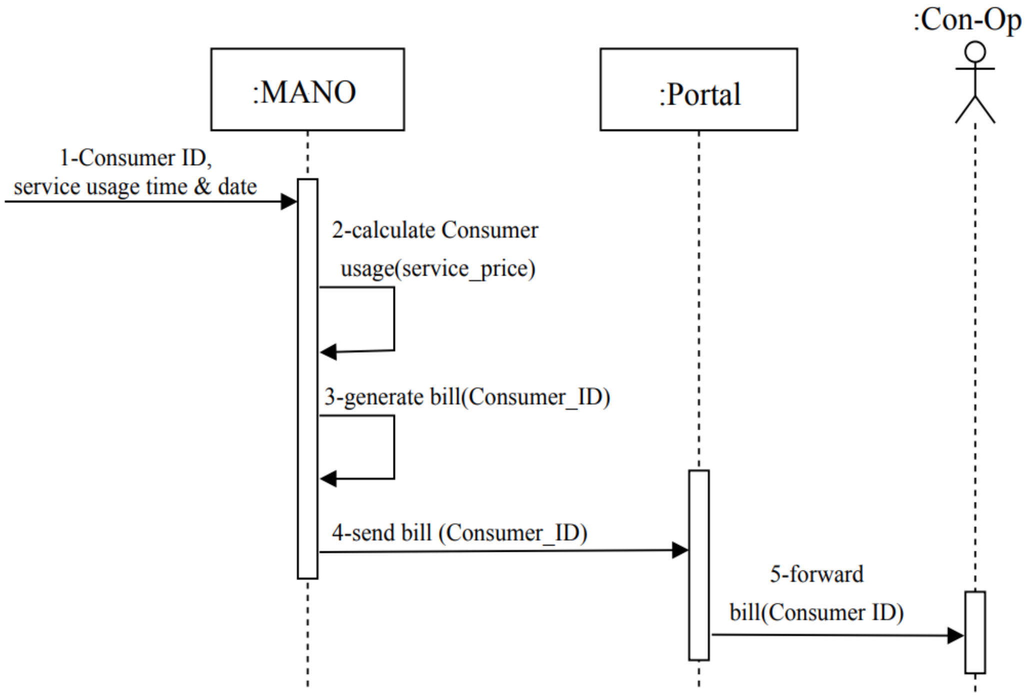


Figure 3.4 Sequence Diagram for the UC8 Bill for Service

UC4- Open Account: in order to use NFV services, consumers must open an account with the service provider.

Summary: consumer operator opens an account to be able to ask for network services.

Actors: Con-op

Precondition: NFVP operator is accepting new accounts.

Description:

1. Con-op sends an account request to the service portal.
2. Portal displays an account form.

3. Con-op submits user information.

4. User information gets forwarded to the NFV.

5. User information gets forwarded to the MANO.

6. MANO validates user info.

7. MANO creates a new user account.

8. New account information gets forwarded to MANO.

9. MANO forwards user account to NFV.

10. NFV forwards user account to the service portal.

11. Service portal displays user account to the consumer.

Exception: user credit document is not accepted.

Postcondition: user gets an account that he can use to request network services

4 BUILDING REFERENCE ARCHITECTURE FOR NFV

4.1 INTRODUCTION

We see the ETSI framework as a block diagram with some vague semantics, which, as indicated earlier, is not precise enough to be the basis of architectural and security analysis. Thus, we use architectural modeling, using patterns to study the ETSI framework, which helps us to analyze the NFV system components and understand how they work.

The pattern diagram in figure 4.1 shows four architectural patterns, which are NFVI pattern [27], VME pattern [28], VNF pattern [29], and MANO pattern [30]. A Pattern can be defined as a solution to a recurrent problem in a specific context [17]. Each pattern describes one of the leading architectural components of NFV described by the ETSI, noting that all our patterns follow POSA template, as shown in subsection 2.1.1. The NFVI is considered the infrastructure of the NFV system and provides the necessary resources for the VNFs [31]. The VME is part of the NFVI, and we describe it in detail due to its essential role in virtualization. Also, it contains the hypervisor that is responsible for creating and managing the VMs needed to run the VNFs. The MANO is the management unit that manages and orchestrates the NFV system. The VNFs are a software implementation of the network functions that provide the functionality of NFV. Combining these patterns provides us with an RA for NFV.

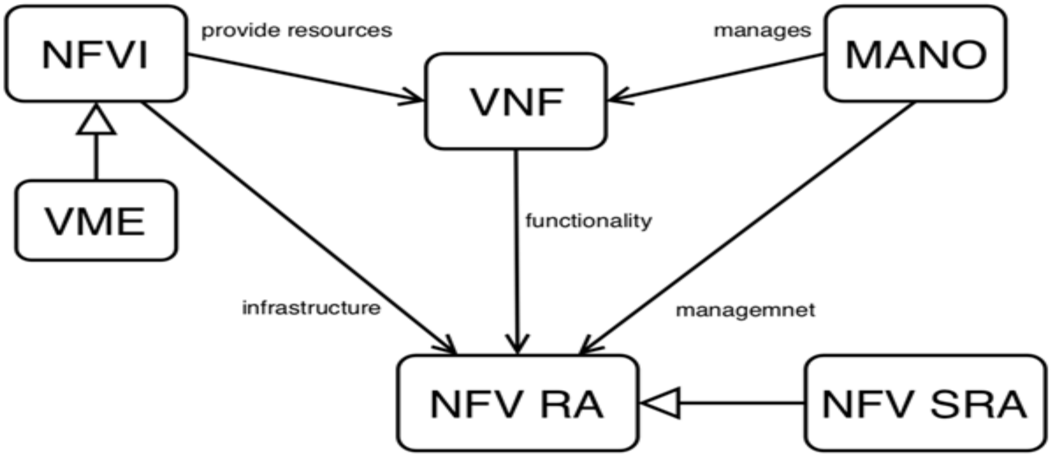


Figure 4.1 Pattern Diagram of NFV RA

4.2 NETWORK FUNCTION VIRTUALIZATION INFRASTRUCTURE (NFVI) PATTERN

4.2.1 Intent

The NFVI pattern describes an architectural layer that contains the actual physical components as well as the virtualization layer that performs the implementation of network functions as virtual functions.6

4.2.2 Example

Dave works in a networking company that provides solutions to his customers. He is in charge of improving the efficiency and flexibility of the networks they provide. He is looking for a more cost-effective solution than using hardware functions as they do now, which they consider inflexible and costly.

4.2.3 Context

Telecommunication providers need to provide a variety of network services such as routers and firewalls to a variable number of customers.

4.2.4 Problem

Providing a variety of network functions in a flexible and cost-effective way is needed to be able to compete in the market. How can we achieve this objective? The solution to this problem is affected by the following forces:

I. Heterogeneity: We need to provide a variety of different types of network functions including some that do not have equivalent physical components.

II. Availability: The network service should be available at most times.

III. Scalability: The number and type of customers can change up or down along time. The number of functions they require also changes along time.

IV. Location: different physical components could be distributed in different locations which may introduce connectivity issues if we used hardware devices.

V. Upgradability: When used together with physical components an upgrade should not introduce issues for virtual functions.

VI. Security: We need to provide a good level of security to our network functions.

VII. Performance: Virtual functions should provide a good level of performance with no latency or noticeable reduction in performance.

VIII. Cost: Using VNFs should be cheaper than using legacy network systems.

4.2.5 Solution

Define an architectural unit (NFVI) that contains a virtualization layer that provides network functions using virtualization of hardware resources. The fundamental component of the NFVI is the hypervisor which creates and manages virtual machines that can be used to create networks for NFV consumers. The hypervisor also provides isolation among the virtualized entities.

Structure

Figure 4.2 shows the class diagram of the NFVI pattern which contains the hardware resources (compute, storage, network), The Virtual Machine Image (VMI), is a template used to create new machines and may contain an operating system, data files, and applications; The VMI Repository contains VMIs. The Virtual Machine Environment (VME) contains the VMM (Virtual Machine Monitor or hypervisor), which in turn manages the resources for the cloud service provider, and the VIM that manages the hypervisor within the VME. The VME is a pattern and is described in more detail in [28]. Also, the diagram shows the boundary of the Virtual Data Center (VDC) that is dedicated to the NFV service. The VDC contains the VIM and the virtual resources of the NFVI.

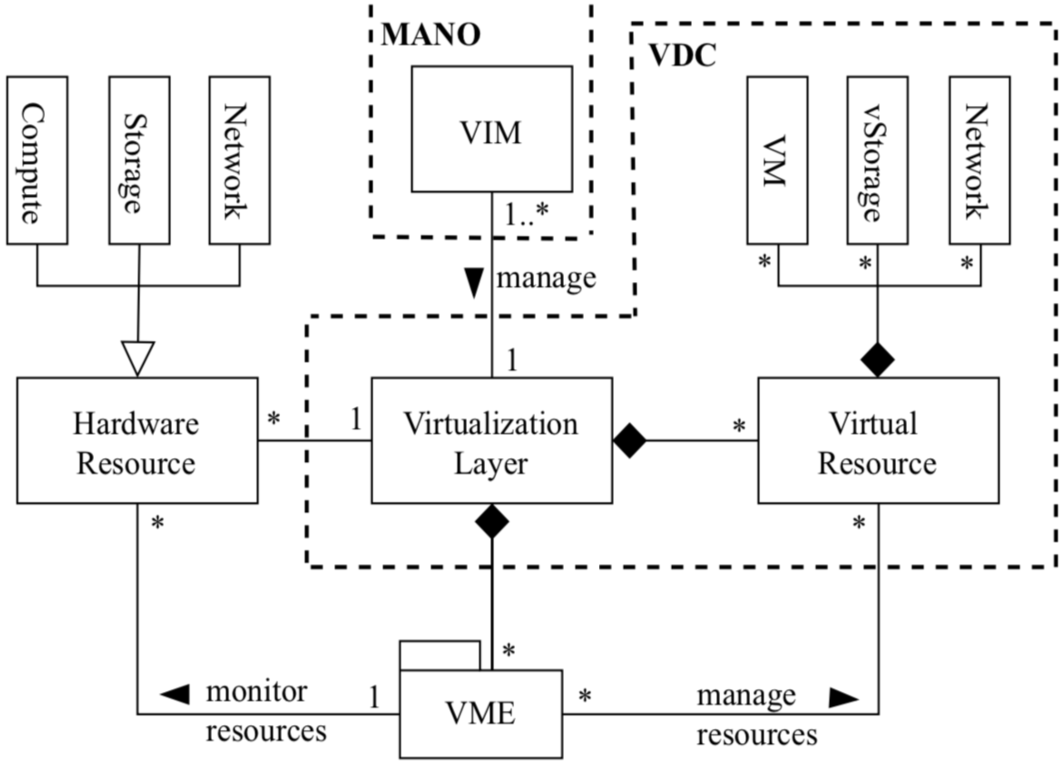


Figure 4.2 class diagram of NFVI pattern

4.2.6 Dynamics

NFVI includes several use cases including “Create a set of Virtual Network Functions”, “Assign virtual resources to user accounts”, and “Modify virtual resources”. In this subsection, we discuss two use cases in detail: “Create a set of Virtual Network Functions”, and “Stop NFV Service”.

UC1: Create a set of Virtual Network Functions for a TSP

Actor: VIM.

Precondition: The customer has an active account.

Description:

1. VIM receives an external request from a TSP to create a service composed of a set of VNFs.
2. VIM sends a create service request with details about the needed resources for the service.
3. The hardware unit handles the request and checks for available resources.
4. Hardware resource details get forwarded to the virtualization layer.
5. The virtualization layer creates a virtual function based on the available hardware resources.

Postcondition: A set of VNFs has been created for a TSP.

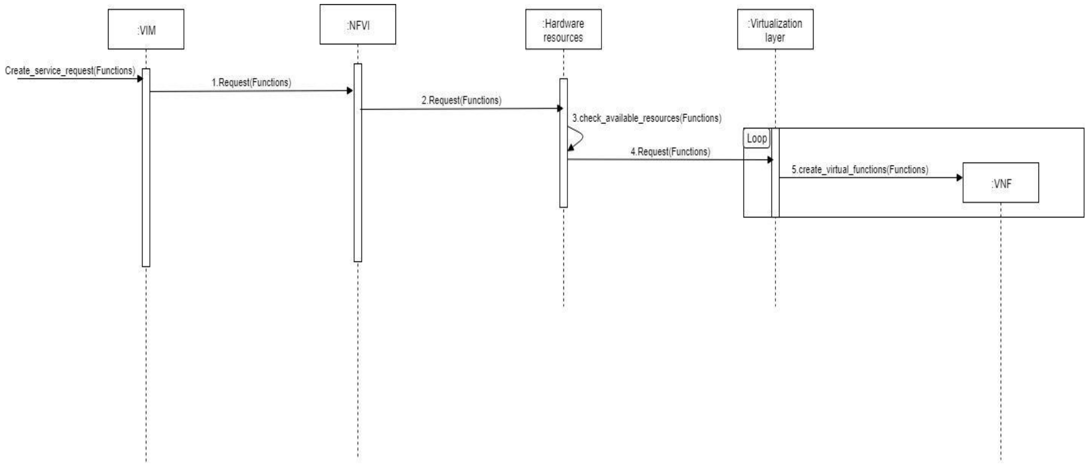


Figure 4.3 Sequence diagram for Use Case “Create a set of Virtual Functions”

UC2: Stop Service for a customer

Actor: VIM.

Precondition: the customer has an active set of VNFs

Description:

1. VIM receives an external request to stop VF.
2. VIM forwards the request to NFVI to handle the request.
3. NFVI sends a stop request to the virtual resources.

4. Once received Virtual resources pause the VF.
5. After successfully passing the VF, the SP stops the service to their customer.

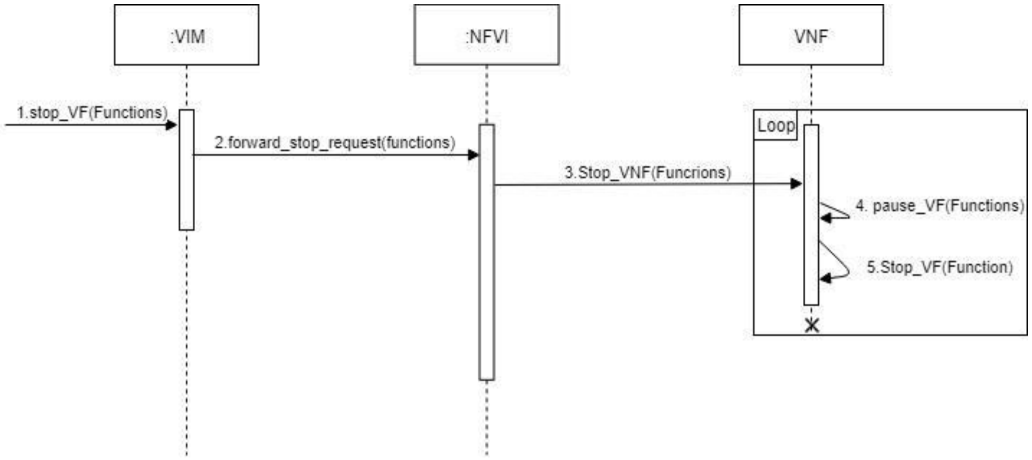


Figure 4.4 Sequence diagram for Use Case 2: “Stop Service”

4.2.7 Implementation

NFV providers must have different hardware equipment ready based on the services they plan to provide to their users base. They need a virtual machine environment as a base for this software in the form of virtual machines created by a hypervisor. Each created service must be registered in a services repository where the Management and Orchestration unit (MANO) can assign it to specific users; this process is explained in detail in the MANO pattern [30]. The VMs in the VDC is used only to deploy NFV services; the VM should have only one VMI installed on it

4.2.8 Known uses

i. Ericsson NFVI solution enables operators to deploy virtual function, OSS, BSS, IT [32].

ii. Ericsson uses the standard provided by ETSI to implement NFVI. Moreover, their solution includes an automated MANO to have better overall performance.

iii. Cisco Network Function Virtualization Infrastructure (NFVI) provides the virtual layer and hardware environment in which virtual network functions (VNFs) can operate [33].

iv. Cloudland by Nokia uses an OpenStack Platform to implement an NFVI [34].

v. Huawei has released its 100 Gigabit Ethernet-capable NFVI solution that provides high bandwidth and low latency performance [35].

4.2.9 Consequences

The NFVI pattern presents the following advantages:

I. Heterogeneity—Since the networking functions are all virtual, they can all easily be made compatible.

II. Availability— NFV services can be running at all times if the TSP includes some redundant implementation of VNFs.

III. Scalability— Since the service will be provided in a virtualized manner the TSP just needs to allocate more cloud resources when the number of customers increases.

IV. Location— The NFVI is part of a TSP and it can use any of the locations of their cloud service.

V. Upgrades— New functions can be added to the existing VNFs without affecting existing functions

VI. Security— the NVF can build secure networks by including appropriate controls in them; for example, authorization to use specifics VNFs.

VII. Performance— The TSP can allocate more VMs to a network to increase its performance.

VIII. Cost— The NFVI provides a cost-effective network to its customers because creating new functions does not require actual hardware costs.

4.2.10 Liabilities

Problems may occur when trying to combine old hardware functions with VNFs due to integration and compatibility issues.

Security is a concern in NFV systems. Every network function is now provided as software which increases its attack surface.

4.2.11 Example Resolved

Dave decided to use an NFV system rented from a public cloud, where physical components are emulated by virtual functions with the help of an NFVI. This allows Dave's company to deliver network functions to the user without the complexity of deploying and installing new equipment.

4.2.12 Related patterns

- A pattern for Network Function Virtualization [36] presented the NFV architecture that shows how to create network services using Software-as-a-Service (SaaS) of the cloud. This is a general pattern than contains the NFVI as a class. This pattern refines the details of that pattern.
- Cloud ecosystem [4] shows how the NFV pattern interacts with the different parts of the cloud ecosystem as well as with other external systems.

- A pattern for an NFV Virtual Machine Environment [28] shows this environment and how it is related to NFV, as well as describing how the NFV architecture interacts with the virtual environment.
- A pattern for NFV Management and Orchestration (MANO) is given in [30]. This pattern shows how the NFVI is managed and controlled by the MANO unit.
- A Reference Architecture model for NFV is provided in [37], which shows all the components of the architecture.
- A Pattern for VNF that shows the virtual network functions in NFV is given in [29].

4.3 A PATTERN FOR NFV VIRTUAL MACHINE ENVIRONMENT (VME)

4.3.1 Intent

The VME of NFV creates virtual machines that can be used to create and host VNFs for Consumers.

4.3.2 Context

NFVP, who are telecommunications company (telco), need to create virtual networks for their Consumers. The VME of NFV allows Consumers to use virtualized network services/functions (VNFs) that support different types of software systems. VNFs may be composed of different components (VNFCs), where each one can be scaled differently.

4.3.3 Problem

NFVP needs to be able to create virtual networks in a fast and flexible way to adapt to market changes or they will not be able to have a successful business. Therefore, the VMs created for NFV must be isolated from each other and be flexible in terms of scalability. Scaling virtualized entities (VNFs) within a traditional VME could propagate to unnecessarily scale other entities. Moreover, compatibility is an issue due to the integration of different hardware components from different manufacturers in the infrastructure.

4.3.4 Forces

The possible solution to this problem is constrained by the following:

I. Flexibility: Consumers may require a variety of VNFs running on different types of software systems; therefore, VMs should be able to run different types of software systems.

I. Elasticity: VNF resources should be dynamically and easily scale-up or down.

II. Extensibility: It should be possible to dynamically add more network services within the VNFs.

III. Isolation: VMs running on the same hypervisor should have strong isolation; in case a VM is compromised the threat should not propagate to the other VMs or to the hosted hypervisor.

IV. Integration: The virtualization environment comprises multiple virtual entities, which should be able to communicate and integrate with each other.

V. Relocation: VMs should be easily migrated to anywhere in the network.

VI. Complexity/Overhead: The virtualization environment should have as little complexity and overhead as possible.

VII. Reliability: The VMs should be reliable to assure continued use of the VNF.

4.3.5 Solution

The VME of NFV includes a hypervisor that creates and manages virtual machines that can be used to create networks for Consumers. The hypervisor ensures isolation among the virtualized entities, as well as flexibility in which Consumers would be able to transparently scale up/down the resources of a VNFC.

4.3.6 Structure

Figure 4.5 shows a class diagram for the Virtual Machine Environment (VME) of NFV (in dotted line). The Hypervisor controls hardware (storage, compute, network) to produce virtual resources. It contains several internal parts including an emulator, which is responsible for emulating the CPU, storage and network devices in VMs; the scheduler which uses a virtual CPU (vCPU) to schedule VMs for execution; a hypercalls table where the virtualization requests by VMs defined such as a call for managing disk partitions and memory management unit (MMU); drivers for all hardware units, i.e., storage, network and input devices, that are necessary to support the VMs when accessing hardware, and a management API that allows the hypervisor to receive commands to perform managerial tasks such as create, terminate, migrate, and replicate a VM. Also, the hypervisor controls the Virtual Machine Image (VMI) repository that stores VM Images (VMIs). Each VM can be built only from one image. The VM class is duplicated into two subclasses, Primary VM and Backup VM, to provide reliability; backing-up the VM is optional upon Consumer request. The VNF is a software functionally equivalent to a network element in the hardware, while the VNFC is an internal component of a VNF. Further, the VIM which is a MANO component manages the hypervisor, which is required to collect and send all the relevant metrics from the compute, storage, and network domains to the VIM.

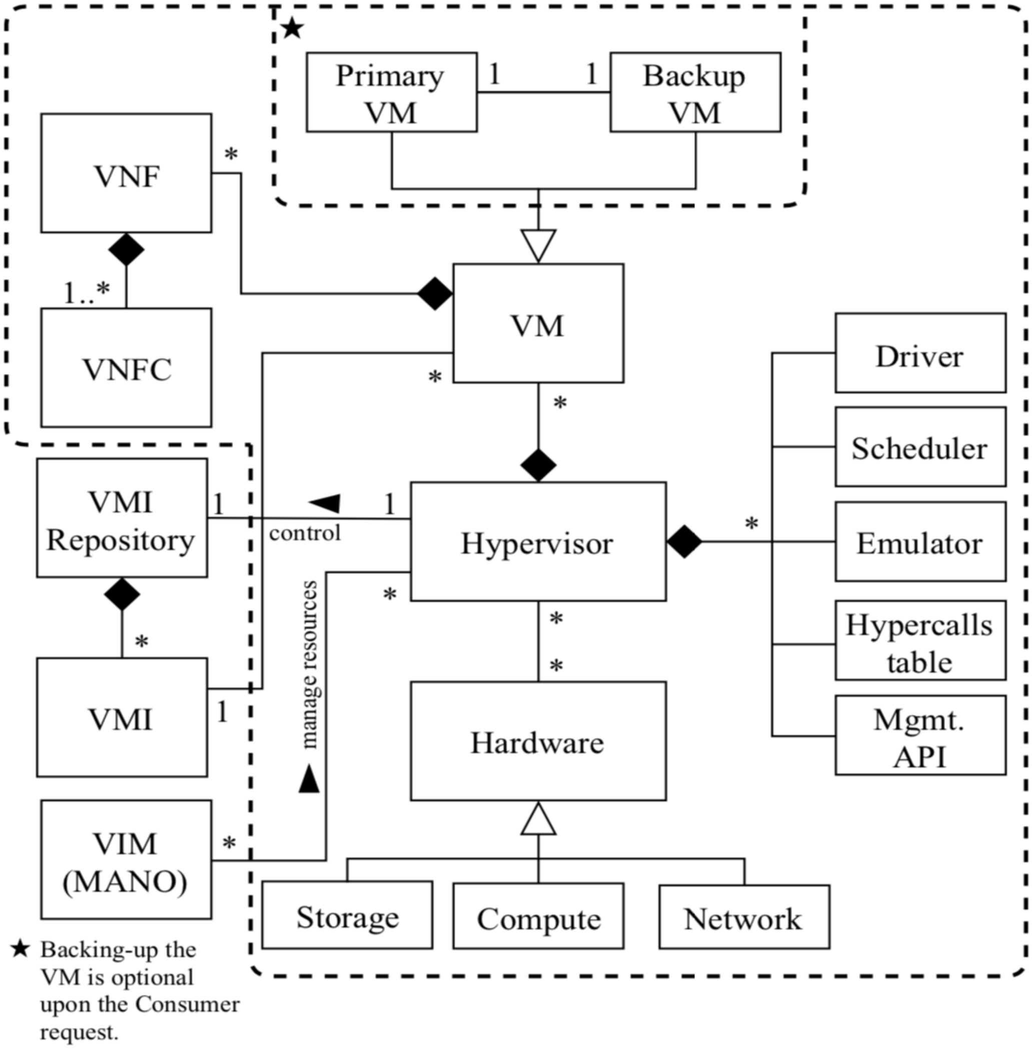


Figure 4.5 Class Diagram of the VME of NFV

4.3.7 Dynamics

UC11: Create a VM upon requesting a VNF (figure 4.6)

Summary: the Con-Op requests a VNF, such as a vSwitch, that in turn requires to create a VM; the request is fulfilled by the VIM, which is a MANO unit responsible for managing resource allocation requests.

Actor: VIM on a request from a Con-Op.

Precondition: the Con-Op should have a valid account to request a VNF.

Description:

1. A recent request from a Con-Op who requests a VNF.
2. The VIM sends a request to the hypervisor to create a VM.
3. The hypervisor receives the request and creates the VM.
4. The hypervisor sends an acknowledgment to the VIM that the VM is created.
5. The VIM will report that the VM is created to host the requested VNF.

Postcondition: A VM is created and assigned to the indirect request of VNF by the Con-Op.

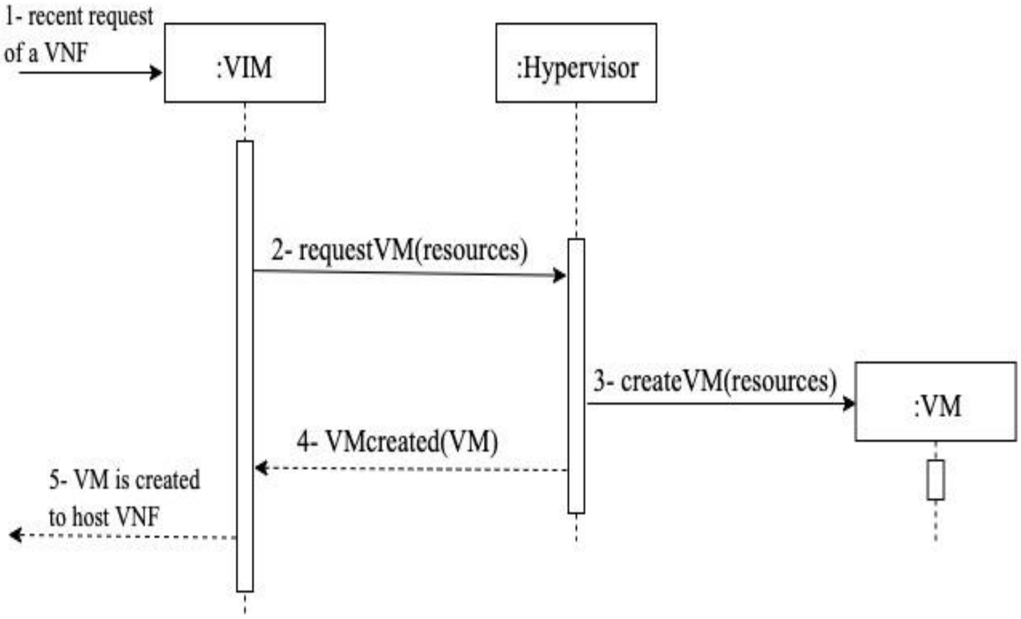


Figure 4.6 Sequence Diagram for the UC Create a VM upon Requesting a VNF

UC21: Modify Resources: this scenario shows the use case of modifying the resources of a VNFC, which is an internal part of a VNF.

Summary: the VIM receives an indirect request from a Con-Op for modifying the resources of a VNFC, for instance increasing the ports of a vSwitch. This allows modifying resources of one component without the need to modify the other components within a VNF.

Actor: VIM on a request from a Con-Op.

Precondition: the Con-Op should have a valid account to request modifying the resources of a VNFC.

Description:

1. A recent request from a Con-Op who requests to modify a VNFC.
2. The VIM sends a request to the hypervisor to modify the resources of a VNFC.
3. The hypervisor checks for available resources.
4. The hypervisor modifies the resources through a controlled VM.
5. The hypervisor sends an acknowledgment to the VIM that the resources of the VNFC are modified.

Postcondition: the resources of the VNFC is modified.

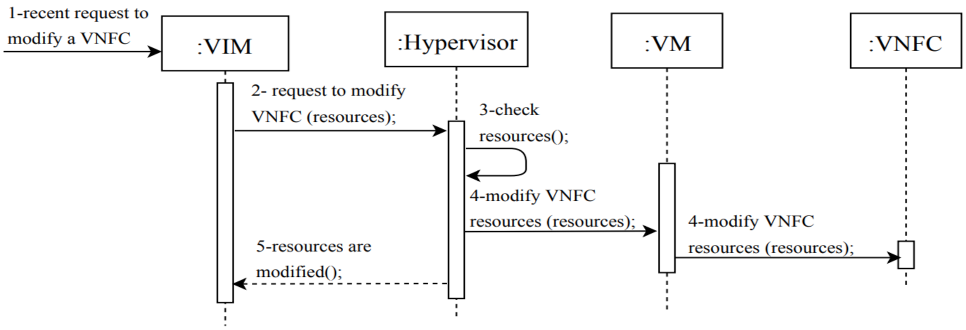

1-recent request to modify a VNFC
:VIM
:Hypervisor
:VM
:VNFC
2- request to modify VNFC (resources);
3-check resources();
4-modify VNFC resources (resources);
4-modify VNFC resources (resources);
5-resources are modified();

Figure 4.7 Sequence Diagram for the UC 21 Modify Resources of a VNFC

4.3.8 Implementation

Figure 4.8 shows a class diagram for the NFVI pattern [27]. The VME of NFV is part of the virtualization layer of the NFVI; it includes the hypervisor and interconnected with several virtualized and hardware components.

As we described earlier, the hypervisors can be classified based on the way the hypervisor installed (type 1 or type 2) and the type of hardware emulation (full or para-virtualization). Our approach here is a monolithic bare-metal hypervisor where the hypervisor runs directly on the hardware, and the device drivers are hosted on the hypervisor itself. This approach ensures a better performance to the VMs as it always have the correct driver installed. Also, it has less attack surface as long as it has less hypercall [38] libraries and the management interface is not in a partitioned VM as it is in microkernel hypervisors . However, there should be various types of drivers such as network interface cards, hard drives, and graphics drives, etc., to make sure they support as much VM as possible. An example product of this approach is VMWare ESXi hypervisor [39]. A drawback of this approach is the hypervisor will run only on a selective number of systems [40]. Also, because they reside on the hypervisor, if one driver is compromised, the threat may propagate to the entire virtual architecture along with the hypervisor.

Xen and Microsoft's Hyper-V instead are bare-metal microkernelized hypervisors [41]. The device drivers are installed in a partitioned VM called parent VM in Hyper-V and Domain 0 in Xen, and then these drivers are inherited to the child VMs. This means VNFs running in child VMs have more flexibility in terms of compatibility with physical hardware. However, this flexibility makes the parent VM to have a larger attack surface [28]. Indeed, if the operating system running on the parent VM is compromised, then all VMs associated with it will crash [42].

Lastly, our pattern shows how the VNFs/VNFCs are created using virtual machines, but software containers can also be used to build VNF [43]. Further, Software Defined Network (SDN) is a network technology that ensures network agility and flexibility by decoupling the data plane from the control plane; SDN and NFV are a complement to each other [44], and can be used to better implement VNFs. [42] leverages OpenFlow, which is a protocol that enables SDN architecture, to implement an NFV router.

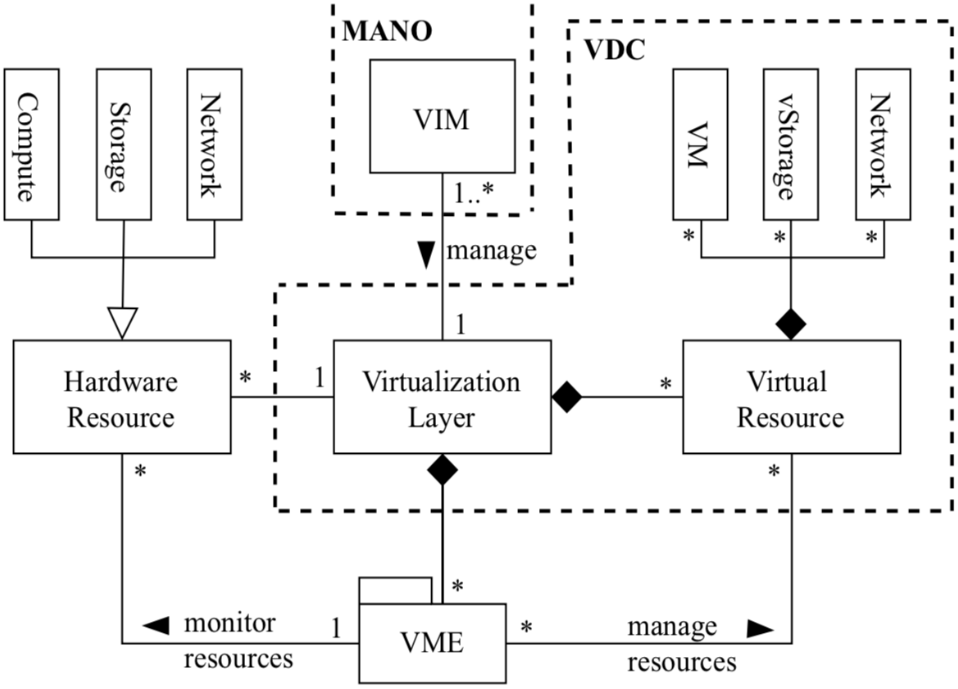


Figure 4.8 Class Diagram of the NFVI Pattern

4.3.9 Known Uses

I. VSphere is a virtualization platform by VMware that provides a virtualization environment for VMs [39].

II. Hyper-V by Microsoft provides a virtual machine environment that enables users to create virtual devices such as virtual hard drives and vSwitches that can be added to VMs [45].

III. The Alcatel-Lucent has Cloudband, an NFV platform that allows service providers to orchestrate infrastructure, applications and networks in a single virtualized infrastructure [46].

IV. Nokia has several NFV products for hosting and orchestrating VNFs within its virtualization environment; for instance, CloudBand infrastructure software, which is an NFVI and VIM software product that optimizes the infrastructure virtualization [47].

V. Cisco and Intel contributed to developing a virtualization environment that runs optimal and high-performance VMs [48].

4.3.10 Consequences

I. Flexibility: We are able to run VNFs on different types of software systems, and we can create a variety of network functions, even some which don’t have an equivalent hardware unit.

II. Elasticity: The NFV hypervisor dynamically handles hypercalls regarding scaling-up or down the VNF resources as well as access to hardware resources.

III. Extensibility: It is possible to dynamically add new network services and functions.

IV. Isolation: The NFV hypervisor provides isolation among the virtualized entities, and mediates access to hardware; a rogue VM would not be able to access resources belonging to the hypervisor or to other VMs.

V. Integration: The NFV hypervisor allows multiple virtual entities to communicate and integrate with each other taking care of managing the virtualization layer of NFV system.

VI. Relocation: It would be possible to migrate or relocate VMs to different locations in the network.

VII. Complexity/Overhead: The NFV hypervisor promises less complexity and overhead than paravirtualization as it supports full virtualization where VMs run as physical machines, and in turn, there will be a low number of hypercalls.

VIII. Reliability: With the VME of NFV, error propagation would be easy to maintain as it supports redundant sites; there will backups for VMs in case a VM is crashed.

4.3.11 Related Patterns

- A pattern for NFV [36]: presented the NFV architecture that shows how to create network services using cloud Software-as-a-Service (SaaS), in which VNFs can be created using VMs.
- A pattern of SaaS in clouds [49]: shows the architecture of cloud SaaS in applications are hosted by cloud providers and offered for consumers as on-demand services; these applications can be in form of VNFs.
- Virtual machine operating system architecture [19]: provides an environment that a hypervisor creates VMs, in which the virtualized resources can be used for NFV.

- A pattern for virtual machine environment [50]: provides an environment in which VMs can be created and managed according to user requests, and enables VMs to execute multiple types of applications and operating systems.
- The software container patter [51]: provides a solution to host applications, binaries, and libraries sharing the same operating system and hardware. Containers are lightweight and secure but have less flexibility than VMs.
- Cloud ecosystem [9]: shows how the NFV pattern interacts with the different parts of the ecosystem patterns.
- Infrastructure-as-a-Service (IaaS) pattern [51]: allows the sharing of infrastructure level resources such as servers, storage, and network.

4.4 NFV MANAGEMENT AND ORCHESTRATION (MANO) PATTERN

4.4.1 Intent

MANO unit is an architectural block for NFV that addresses management and orchestration of the entire NFV service. This includes computing, networking, and virtual resource management, and network life cycle monitoring.

4.4.2 Example

Dave is subscribed to an NFVs service that provides him with twenty ports, four load balancers, one QoE measurement tool and finally a firewall to ensure the security of the network. However, Dave’s business grows, and he has decided to extend his office so he needs to scale up his network and add another twenty ports which will also lead to adding four more load balancers. He requested them from the TSP. How can the TSP provide them and scale up the current service?

4.4.3 Context

Different components in the NFV architecture such as physical resources and virtual services (VNFs) have to be managed and assigned to network users in a proper way. Together they provide Network Service (NS) to the user. We can define Network Service (NS) as a package of network functions virtualized to the user to provide network functionality similar to the one the user consumes using legacy network systems . In NFV these network packages are called network graphs. Moreover, different services must be visible to users in a network services catalog that gets updated regularly according to the available resources in the pool in order for the user to choose from it.

4.4.4 Problem

NFV contains different functions from different vendors that must be managed. This includes resource allocation in the NFVI which could be very complex since different requirements and constraints have to be met at the same time. Also, the allocation and release of resources is a dynamic process that needs to be fast and effective in response to their consumption.

4.4.5 Forces

The solution to the management of NFV functions is affected by

I. Flexibility: we would like to run different types of network services because the consumer has different needs.

II. Isolation: users shouldn’t be able to have access to system functions or have access to functions provided to other users whether by error or intentionally.

III. Security: The created network graph should be secured from unauthorized access.

IV. Scalability: increasing the number of units in NFV may introduce integration problems due to the variety of interfaces and protocols. We need to assure their smooth integration.

V. Modularity: NSs should have a well-defined interface in order for the user or another subsystem to interact with it and to improve interoperability.

VI. Extensibility: We would like to be able to add new functions and features because of the changing needs of consumers.

VII. Management: We need to be able to handle the lifetime of a large number of networks with different requirements.

VIII. Performance: We need to provide the consumer with a satisfactory level of performance.

4.4.6 Solution

MANO contains components for the management of the system allowing the NFV components to interact with each other in the NFV system. The MANO manages virtualized resources as well as hardware resources including updating, scaling and operating the resources. Moreover, it takes care of managing the virtual functions and the life cycle of VNFs. The MANO also handles the emulation of hardware resources and their presentation as a software entity. Also, The MANO contains different catalogs for the resources that together create NSs.

4.4.7 Structure

Figure 4.9 shows the class diagram for the MANO pattern in dotted lines as well as its interaction with other components in the system.

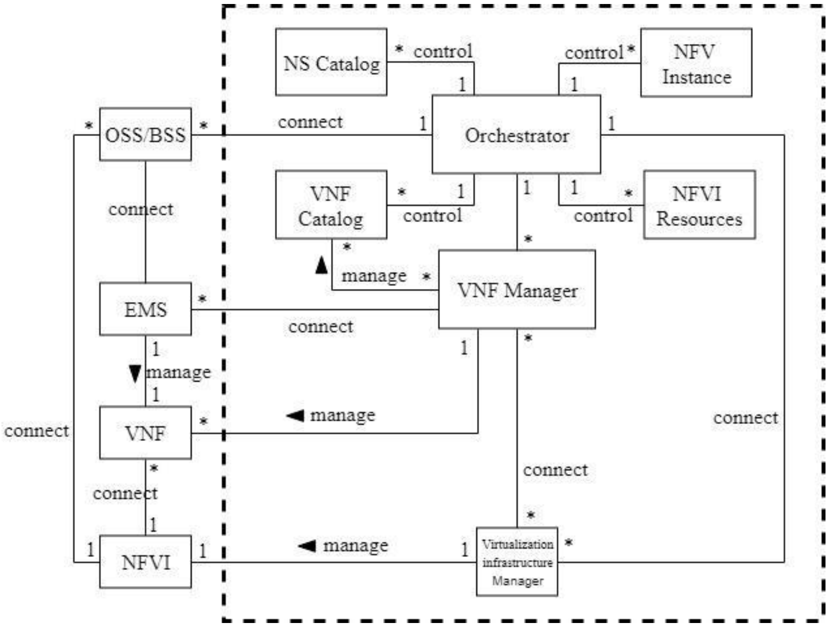


Figure 4.9 Class diagram for the structure of MANO unit

Orchestrator is considered the heart of the MANO pattern since almost all the components of the system connected to it. Including NFV instance, NS catalog, VNF catalog, and NFVI resources. VNF Manager takes control of Element Management Systems, which in turn controls the actual VNF. VNF Manager communicated with the Virtualization infrastructure manager to update the Orchestrator with the status of the infrastructure of the system. The NFVI is also responsible for four more tasks which are:

- Fault and performance management of hardware, software, and virtual resources.
- Manages the life cycle of virtual resources in an NFVI domain.
- Monitor the inventory of virtual machines and which physical resources related to it.
- Manage the emulation of physical resources into virtual resources.

The NS catalog, NFV instance, VNF catalog, and NFVI resources are all repositories that contain metadata about different components that could be used to build an NFV service [52]. NS catalog contains a list of usable network services as well as a template for possible services to be provided to user, NFV instance holds all details about network services instances and how its related to VNF instances catalog which contain description of VNF deployment and operational processes ,and NFVI resources which contain all the resources available to the system to establish NFV service. Finally, the NFV orchestrator class takes care of the coordination and management of the four repositories earlier mentioned.

4.4.8 Dynamics

MANO use cases include “Request NFV network”, “Modify NFV network”, “Migrate NFV service”, as well as other use cases. In this subsection, we cover three use cases which are “Request NFV network”, “Terminate an NFV network”, and “Modify an NFV service”.

UC1: Request NFV network

Summary: the NFV consumer requests a network, and in turn, the NFV orchestrator handles the request.

Actor: NFV orchestrator.

Precondition: the NFV consumer has a valid account with the NFV provider.

Description

1. The NFV orchestrator receives an NFV network request from a consumer.
2. NFV orchestrator checks available resources in the NFVI resources repositories
3. NFV Orchestrator forwards requests to the VIM.
4. VIM forwards the request to NFVI.
5. The NFVI sets up the service and create a VMs to host the service.
6. NFVI confirms network creation to the VIM.
7. VIM confirms network creation to NFV orchestrator.
8. NFV Orchestrator assigns service ID to the user ID which requested it.

Postcondition: a new NS will be created and assigned to the user who requested it.

Exception: the requested network cannot be provided due to a lack of resources. After the NFVI receives the request and before creating the network graph it must check for the resources and inform the customer with the result.

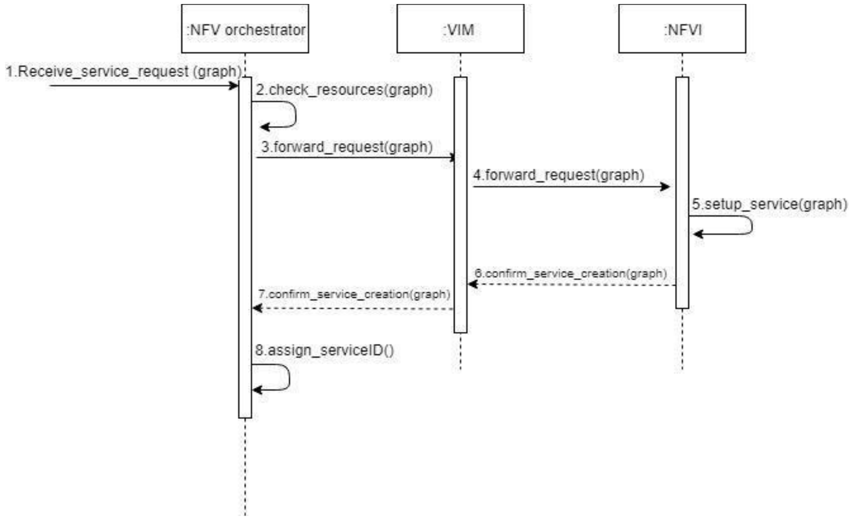


Figure 4.10 Sequence diagram for requesting a service

UC2: Terminate an NFV service

Summary: the NFV consumer requests termination of service, and in turn the NFV orchestrator handles the request.

Actor: NFV orchestrator.

Precondition: the NFV consumer has a working network.

Description:

1. NFV orchestrator receives a service termination request from a consumer.
2. NFV orchestrator forwards the request to VNF manager.
3. VNF Manager terminates VNFs related to the network; this step is repeated until all VNFs services are terminated.
4. VNF manager notifies VIM of NFV service termination.
5. VIM notifies NFVI of NFV service termination.
6. NFVI releases resources of NFV service.
7. NFVI notifies the VIM of the release of resources.
8. VIM notifies the NFV orchestrator of the release of resources.

9. NFV orchestrator updates repositories.

Postcondition: The network will be terminated, and the resources of the network will be returned to the resource pool.

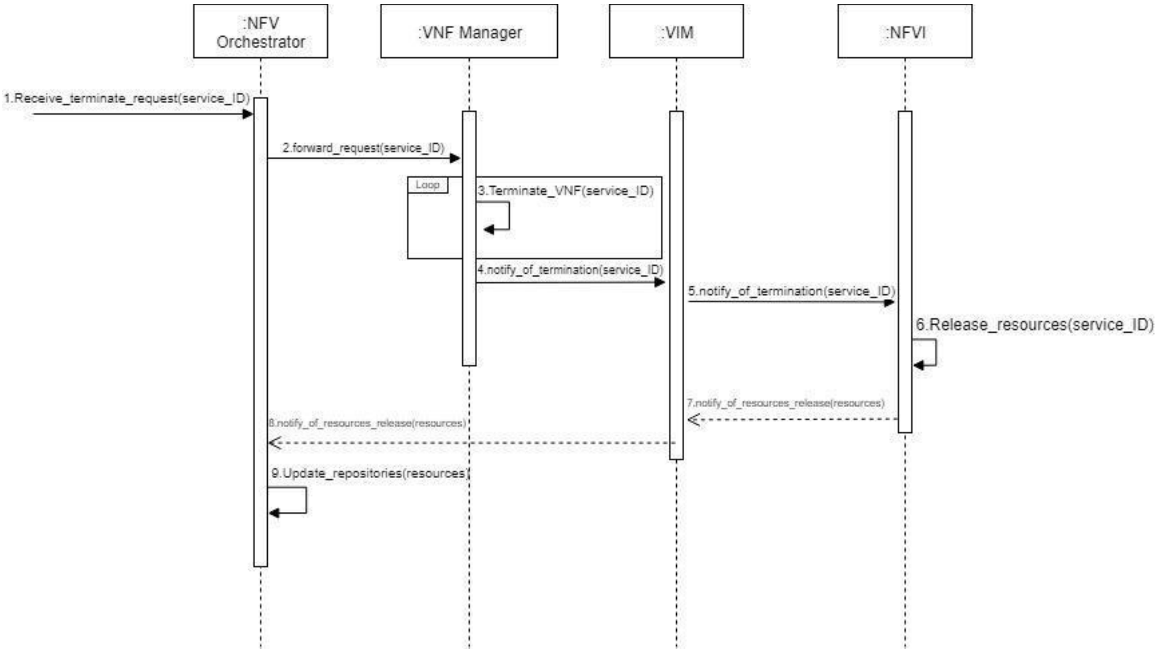

:NFV Orchestrator
:VNF Manager
:VIM
:NFVI
1.Receive_terminate_request(service_ID)
2.forward_request(service_ID)
Loop
3.Terminate_VNF(service_ID)
4.notify_of_termination(service_ID)
5.notify_of_termination(service_ID)
6.Release_resources(service_ID)
7.notify_of_resources_release(resources)
8.notify_of_resources_release(resources)
9.Update_repositories(resources)

Figure 4.11 Sequence diagram for service termination.

UC3: Modify an NFV service

Summary: the NFV consumer requests a modification of network service that he/she already has, and in turn, the VNF manager handles the request.

Actor: VNF Manager.

Precondition: the NFV consumer has a working network service.

Description:

1. VNF manager receives a network service modification request from the user.
2. VNF manager forwards user requests to the NFV orchestrator.

3. NFV Orchestrator checks available resources in the resource's repository.
4. This use case has two alternative scenarios:

A-[Resources Available]

1. NFV Orchestrator forwards the request to VIM.
2. VIM forwards the request to NFVI.
3. NFVI modifies the network service according to the user needs.
4. NFVI confirms network service modification to VIM.
5. VIM confirms network service modification to NFV orchestrator.
6. NFV Orchestrator confirms network service modifications to NFV manager.

B-[Resources not available]

1. NFV Orchestrator Notifies VNF manager of unavailability of resources

Postcondition:

1. The user network service will be modified to meet his needs.
2. The user will be notified that there aren't available resources to fulfill his request.

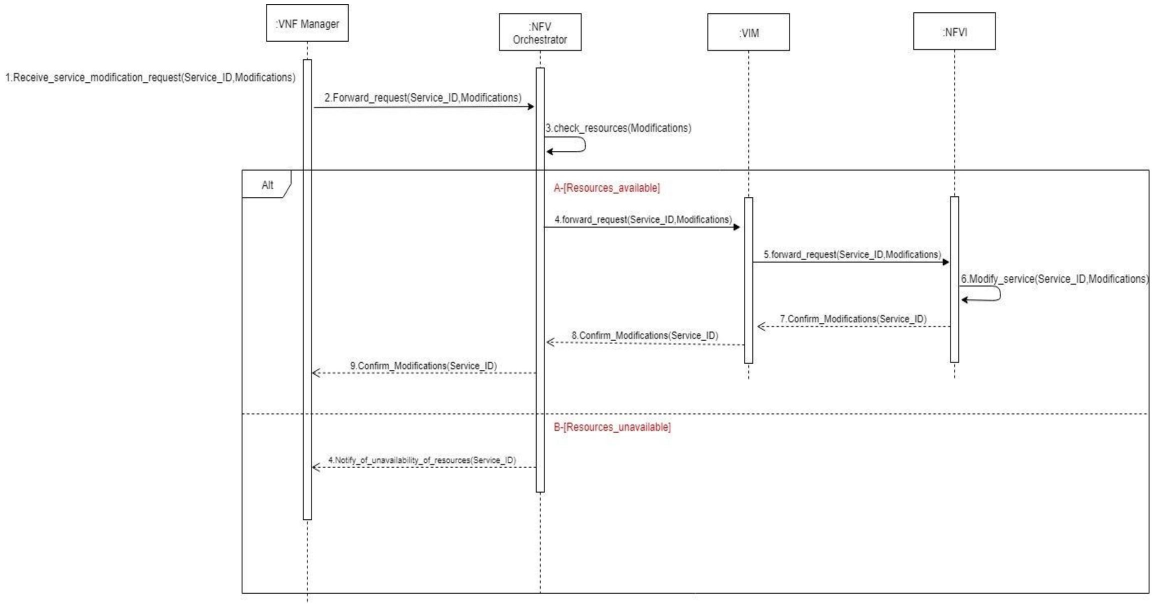

:VNF Manager
:NFV Orchestrator
:VIM
:NFVI
1.Receive_service_modification_request(Service_ID,Modifications)
2.Forward_request(Service_ID,Modifications)
3.check_resources(Modifications)
Alt
A-[Resources_available]
4.forward_request(Service_ID,Modifications)
5.forward_request(Service_ID,Modifications)
6.Modify_service(Service_ID,Modifications)
7.Confirm_Modifications(Service_ID)
8.Confirm_Modifications(Service_ID)
9.Confirm_Modifications(Service_ID)
B-[Resources_unavailable]
4.Notify_of_unavailability_of_resources(Service_ID)

Figure 4.12 Sequence diagram for Modify an NFV service

4.4.9 Implementation

A TSP that implements VNFs can either create their own MANO unit or it can be rented from other providers to be composed with the NFV architecture that TSP provides. The MANO should provide management for the different entities and components of NFV [53]. VNF management unit is always a part of MANO controls and access all the VNFs in the system to provide management tasks including fault and performance management. A Virtualized Infrastructure Manager is also created to have access to the hardware resources and takes care of decoupling and transforming these hardware resources into virtual resources in order to be used to build VNFs. The internal components of MANO are interconnected with each other through different reference points, reference points also connect external NFV components to the MANO [54].

A good implementation of MANO is Open source MANO (OSM) which is created by ETSI with the help of different contributors in the industry. Open MANO allows the entire infrastructure of NFV to be deployed and configured in minutes to avoids the need for expensive and time-consuming customization [55]. Open MANO have the following advantages [56]: Automated end to end service orchestration, a plugin model that allows integration of multiple NFVIs with different hardware's, fault and performance management of NS, support for network service scaling, support simplifying VNF package generation with user friendly interface, support for NS instances with VNFs running in multiple different datacenters, and catalog and repositories search functions.

Several main leaders in the networking industry use OSM such as Atos, Cable labs and Verizon. Figure 4.13 shows a simplified view of OSM architecture [55].

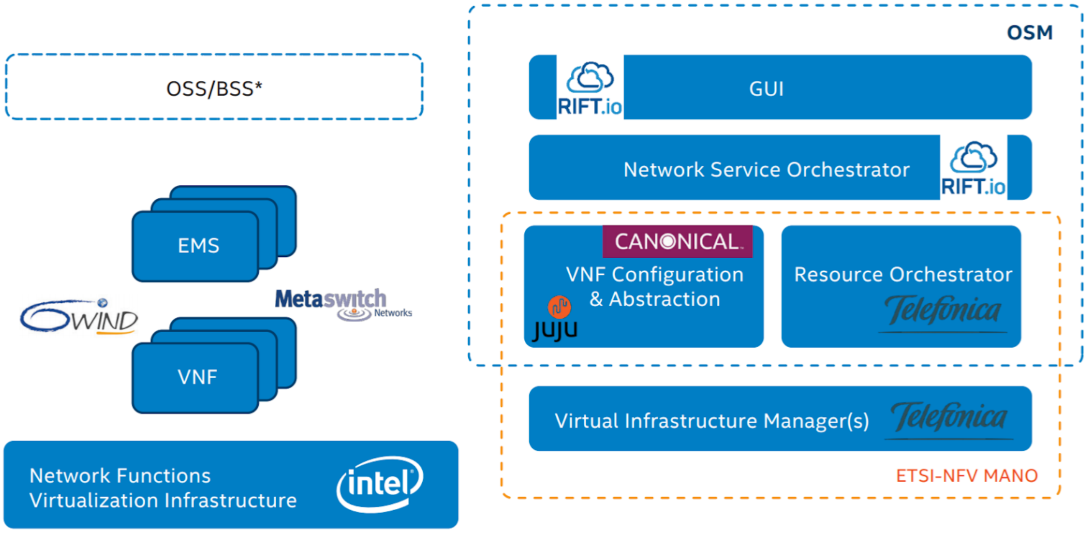

OSS/BSS*
OSM
GUI
RIFT.io
Network Service Orchestrator
RIFT.io
EMS
Metaswitch Networks
WIND
VNF
CANONICAL
VNF Configuration & Abstraction
juju
Resource Orchestrator
Telefonica
Virtual Infrastructure Manager(s)
Telefonica
ETSI-NFV MANO
Network Functions Virtualization Infrastructure
intel

Figure 4.13 OSM architecture simplified view

4.4.10 Known uses

I. Ericsson uses Anuta Networks Solution in order to provide Management and orchestration for their NFV architecture [57].

II. The OPEN-Orchestrator Project has an open-source orchestration that provide orchestration and management for software-defined networking (SDN) and network function virtualization (NFV) operations provided by the Linux Foundation [58].

III. Hewlett Packard Enterprise (HPE) uses a MANO provided by Ciena called Blue Planet NFV orchestrator [59].

IV. Huawei created their own MANO which is called CloudOpera MANO [60].

V. Open source MANO (OSM) is an open source project provided by ETSI, which implements the ETSI MANO framework [55].

4.4.11 Consequences

The MANO pattern presents the following advantages:

I. Flexibility: we can run different types of network services.

II. Isolation: The VIM ensures that the underlying infrastructure is protected.

III. Security: VNF manager should ensure that network functions and network graphs including virtual links, virtual networks, sub-nets etc are only accessed by authorized users.

IV. Scalability: MANO can handle the increase of NFV services

V. Modularity: Orchestration part of the MANO provides NS execution with a

VI. standard interface that is easy to adopt to other subsystems.

VII. Extensibility: It is possible to dynamically provide additional services by adding more resources to the repositories.

VIII. Management: using MANO we can handle and manage the life cycle of any number of networks with different requirements in the same time

IX. Performance: We can provide the consumer with a satisfactory level of performance.

4.4.12 Example Resolved

When Dave wants to expand his network service, he will log into the portal of his NFV provider and request network service modification. The request will be sent directly to the MANO in order to be handled. The MANO will check in the repositories for available resources and then forward the request to the VNF manager to perform the necessary changes to the service. In Dave’s case the VNF manager will add twenty more ports, four new load balancers, and make sure that they are connected to the network and expand the coverage of the firewall to include the new ports. finally, the orchestrator will notify Dave with the result of the request.

4.4.13 Related Patterns

- A pattern for Network Function Virtualization [36]: presented the NFV architecture that shows how to create network services using cloud Software-as-a-Service (SaaS).
- Virtual Machine Operating System architecture [19]: describes the structure of the hypervisor and its VMs.
- Cloud ecosystem [9]: shows how the NFV pattern interacts with the different parts of the ecosystem.
- A pattern for an NFV Virtual Machine Environment [28]: shows this environment and how it is related to NFV, as well as describing how the NFV architecture interacts with the virtual environment.
- A pattern for SaaS is given in [49].
- A Pattern for VNF is given in [29].

4.5 VIRTUAL NETWORK FUNCTION (VNF) PATTERN

4.5.1 Intent

A Virtual Network Function (VNF) is a software module which realizes network functions as software rather than having them represented via hardware. A set of VNFs can be combined into a network graph that represents a complete network.

4.5.2 Example

Dave wants to start a company where he will offer a variety of network configurations that can subsequently be modified. Therefore, he will need a variety of network solutions that provide the required flexibility to achieve that objective.

4.5.3 Context

Telecommunication companies that intend to present network functions to their consumers in a flexible and cost-efficient way.

4.5.4 Problem

Different network functions may be provided from different vendors and require different network devices which may make integrating these components with each other troublesome. If we need to scale the network up or down, we will need to spend time and money. Moreover, the installation of the network equipment will need professional technicians. Also, updating the equipment will also require a lot of effort to keep up with new technologies; hardware devices are not flexible and it is not easy to modify their functions. How can we provide a more flexible and responsive services to our customers?

4.5.5 Solution

The solution to this problem is affected by the following forces:

I. Heterogeneity: we want to be able to combine VNFs coming from different providers which may have different standards.

II. Life cycle: VNF life cycles have different stages starting from emulation hardware to software until termination; therefore, we need proper management as well as the ability to configure them without disturbing the service.

III. Availability: VNFs have to be up and running at most times.

IV. Scalability: VNFs should be able to scale up or down and to include a single VNFC (VNF software component) or several VNFCs without a problem.

V. Location: VNFs should be dynamically portable and accessible from any location with no problems.

VI. Upgrades: VNF software should be upgradeable without affecting the continuity of provided services.

VII. Security: The created network graph should be able to be secured from unauthorized reading or modification.

VIII. Modularity: VNFs should have a well-defined interface in order for the user or another subsystem to interact with them and to improve interoperability.

IX. Performance: VNFs should perform all the tasks that the legacy networks provide to users' expectations.

4.5.6 Structure

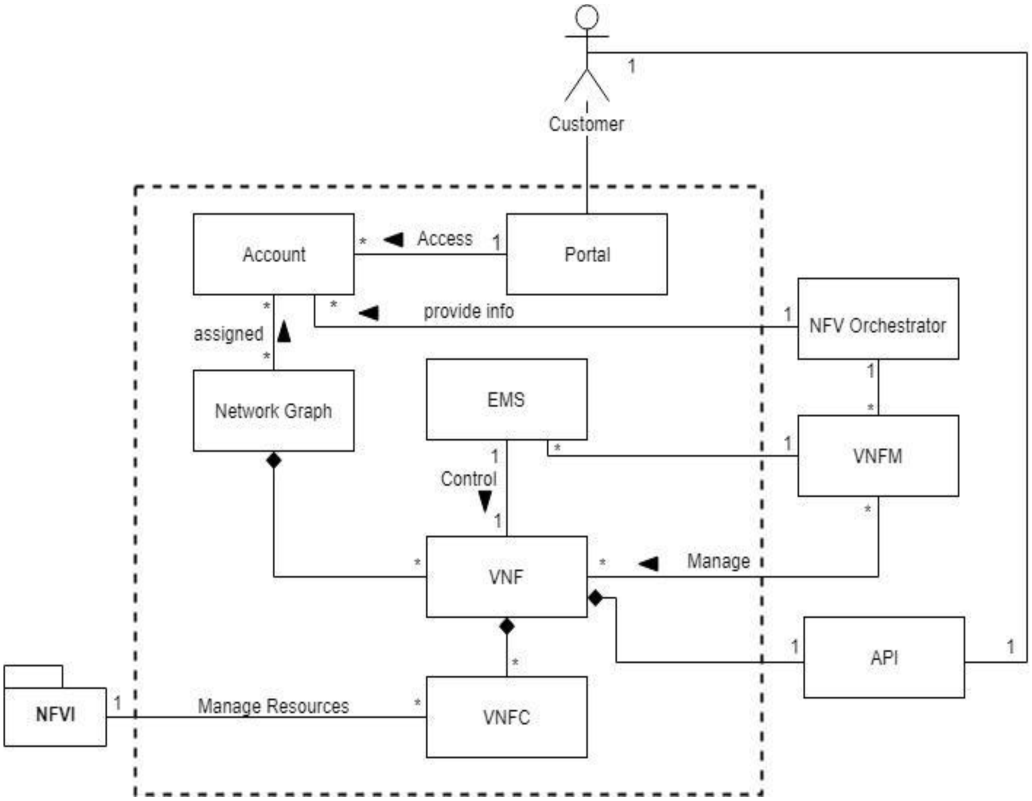


Figure 4.14 Class diagram for the VNF pattern

Figure 4.15 shows the class diagram of the VNF pattern (within dotted lines). A fundamental class in this pattern is the VNF Manager which controls each VNF in the system, including scaling up and down the VNF and monitoring it. Each customer that uses the system can request networks through a Portal; customers can also modify and terminate the network. Each Network Graph represent a full network service assigned to an Account in the system. The EMS class takes control of operating the VNF throughout the VNF life cycle. NFVI manages the resources of the VNFCs and do several other tasks such as handling the emulation process of hardware components into virtual network components. Each VNF in the system is created from the combination of one or more VNFCs.

4.5.7 Dynamics

VNF use cases include “Create VNF” (create a new VNF), “Consume VNF” (use an existing VNF), “Request VNF” (request a VNF from a telco catalog), “Terminate VNF”, “Modify VNF” (add or modify operations), “Delete VNF” (from the telco catalog), and “Build a Network Graph using VNFs”. In this section we describe three use cases: “Consume VNF”, “Request VNF” and “Terminate VNF.

UC1: Consume VNF (Figure 4.16).

Summary: the NFV consumer wants to use an existing VNF.

Actor: NFV consumer.

Precondition: the NFV consumer has a valid account with the NFV provider.

Description:

1. NFV consumer logs in to the portal with his account details.
2. The system verifies the user credential.
3. The consumer gets access to the account.
4. The consumer gets access to the network graph
5. The consumer consumes the VNF service.

Postcondition: The user has access to the VNF service and may consume it.

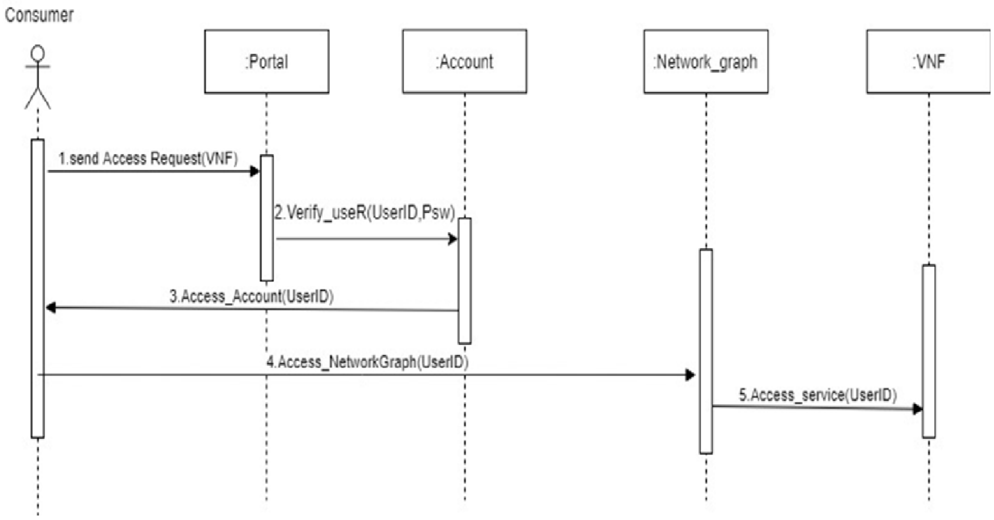


Figure 4.15 Sequence diagram for consuming VNF service

UC2: Request VNF (Figure 4.17)

Summary: the NFV consumer requests a VNF service from the catalog to be added to the existing network graph that the user got from the NFV provider.

Actor: NFV consumer.

Precondition: the NFV consumer has a valid account with the NFV provider.

Description:

1. The consumer sends a VNF request with its needed requirements.
2. Portal forwards user request.
3. Consumer account gets verified.
4. Account validates the requirements and forwards request to NFV orchestrator to handle the request.
5. NFV orchestrator forwards the request to VNF Manager.
6. VNF manager assigns VNF to the user ID.
7. New VNF is assigned to Network Graph.
8. Network Graph assigned to user ID and account.

9. Account confirms request success.

10. Portal confirms to the user the success of his request.

Postcondition: The system will create a new VNF and assigns it to the user account ID.

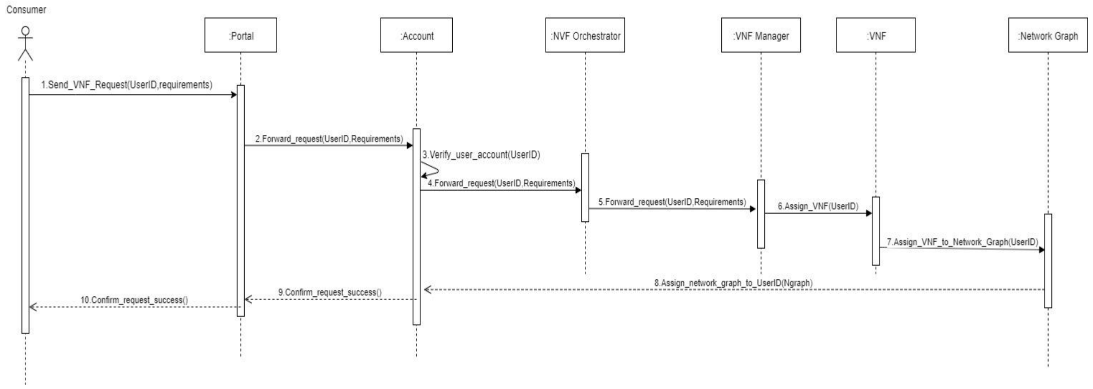

Consumer
:Portal
:Account
:NVF Orchestrator
:VNF Manager
:VNF
:Network Graph
1.Send_VNF_Request(UserID,requirements)
2.Forward_request(UserID,Requirements)
3.Verify_user_account(UserID)
4.Forward_request(UserID,Requirements)
5.Forward_request(UserID,Requirements)
6.Assign_VNF(UserID)
7.Assign_VNF_to_Network_Graph(UserID)
8.Assign_network_graph_to_UserID(Ngraph)
9.Confirm_request_success()
10.Confirm_request_success()

Figure 4.16 Sequence diagram for requesting a VNF service

UC3: Terminate VNF (Figure 4.18)

Summary: the NFV consumer has a network graph that contains several VNFs and he wants to terminate one of the VNFs that he doesn't need anymore.

Actor: NFV consumer.

Precondition: the NFV consumer has a valid account with the NFV provider and an active network graph.

Description:

1. The Customer sends an VNF termination request to the portal with the ID of the VNF he wishes to terminate.
2. The portal forwards the customer request to the account class.
3. Consumers account gets verified.
4. The request is forwarded to the NFV Orchestrator.
5. NFV orchestrator forwards the request to VNF Manager.

6. VNF manager sends pause request with the VNF ID to the target VNF.
7. VNF sends a confirmation to VNF manager that the specific VNF is paused.
8. VNF manager sends terminate request to the target VNF.
9. The target VNF gets terminated.
10. Confirmation of termination is sent to the VNFM.
11. VNFM confirms Termination to NFV orchestrator
12. NFV orchestrator confirms Termination to Account unit.
13. Account unit confirms Termination to the portal.
14. The portal notifies the user of termination of the selected VNF.

Postcondition: A specific VNF has been terminated.

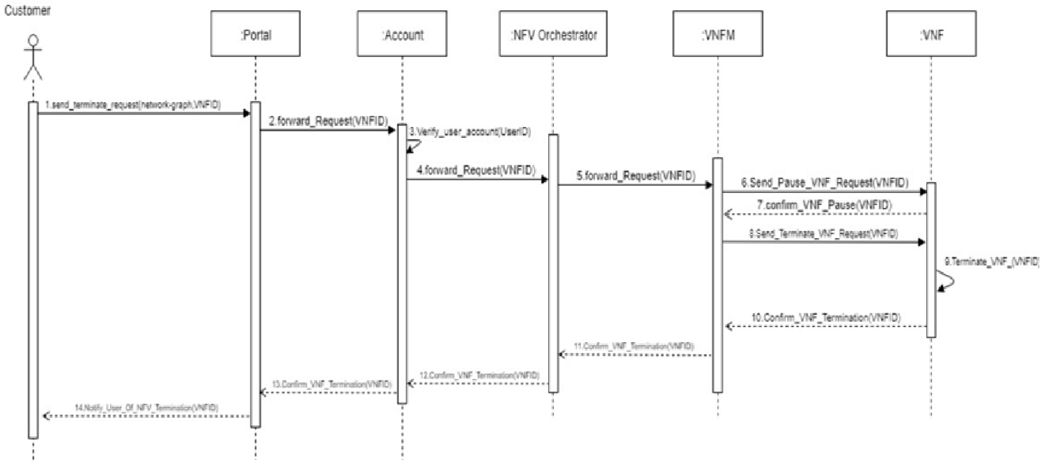

Customer
:Portal
:Account
:NFV Orchestrator
:VNFM
:VNF
1.send_terminate_request(network-graph,VNFID)
2.forward_Request(VNFID)
3.Verify_user_account(UserID)
4.forward_Request(VNFID)
5.forward_Request(VNFID)
6.Send_Pause_VNF_Request(VNFID)
7.confirm_VNF_Pause(VNFID)
8.Send_Terminate_VNF_Request(VNFID)
9.Terminate_VNF_(VNFID)
10.Confirm_VNF_Termination(VNFID)
11.Confirm_VNF_Termination(VNFID)
12.Confirm_VNF_Termination(VNFID)
13.Confirm_VNF_Termination(VNFID)
14.Notify_User_Of_NFV_Termination(VNFID)

Figure 4.17 Sequence diagram for terminating a VNF service

4.5.8 Implementation

NFV providers can implement VNFs through composing one or multiple VNFCs; each VNFC defines a network function as a software entity deployed into a virtualized container. Multiple VNFs have to be chained with each other in order to build complex services in a process called Service Function Chaining (SFC) [61].

Each function is emulated by a virtual function performed through a virtualization layer in the NFVI and managed by the VNFM. Figure 4.19 illustrates the VNF life cycle. The VNF manager by default creates different network graphs that may meet users’ requirements. The process starts with the development of the network graph by combining several VNFCs together [62]. Next, the VNF manager has to validate that the different components work with each other in sync without any component affecting the performance of other components. The VNFs will be deployed to the user upon request. Once the user requests a VNF the management process starts from setting up the service and assign it to the user account until termination of the service. Other management tasks include scaling up and down the network and stop the VNF.

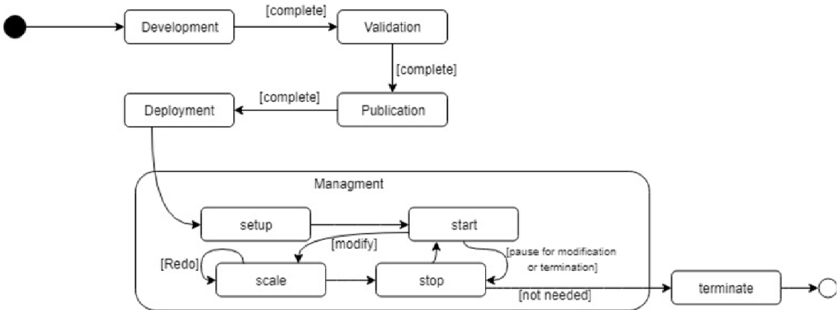


Figure 4.18 VNF life cycle

4.5.9 Known uses

I. Cisco uses VNFs in its CSR 1000V cloud services routers [63]. This router includes a zone-based firewall with Access Control Lists for authorization. It runs on a Google Cloud Platform (GCP).

II. Blue Planet is a division of Ciena networking solutions that provides VNF services [64]. They combine NFV with SDN for potential greater user control and increased efficiency.

III. F5 solutions provide Virtual Network Functions including Vrouting and load balancing as well as complete life cycle management of VNFs [65]. The packages can be purchased in capacity-based throughput options of 5, 10, and 50 Gbps increments, which simplifies network planning, sizing, and purchasing. This solution includes the F5 VNF Manager [66].

IV. Alcatel-Lucent has the Cloudband [46], with which users can select and instantiate individual VMs and storage volumes as well as complex carrier applications. Using Bell Labs algorithms, CloudBand finds an optimal location for these virtual elements based on resource availability, latency and high availability requirements, compliance, regulatory requirements and other business policies. These optimizations can be tailored around resource cost, cost-to-operate, customer requirements, compliance, and regulatory concerns. Cloudband can operate with different hardware platforms.

V. Ericsson has implemented VoLTE (Voice over LTE) services using VNFs [52].

VI. Qosmos is a networking solution company that provide NFV services as well as standalone VNF components and tools to the customers [67] [61]. They implement the Orchestrator using OpenStack functions. Figure 4.20 shows the Qosmos architecture for VNF service.

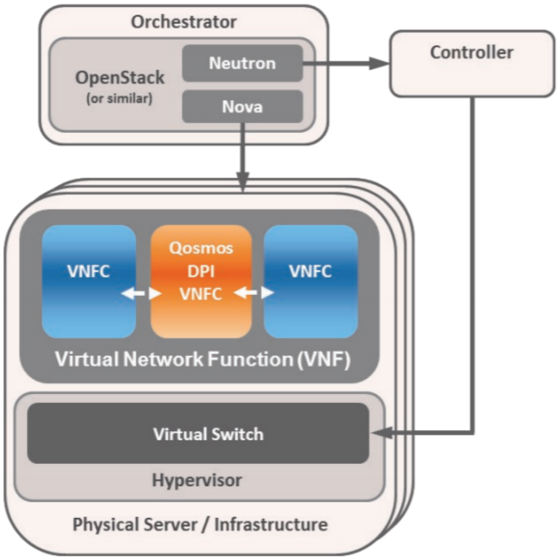


Figure 4.19 Implementation of Qosmos VNFC in an NFV architecture

4.5.10 Consequences

The VNF pattern presents the following advantages:

I. Heterogeneity: different network functions can be integrated with each other without worrying about compatibility between hardware components or different standards since they are now implemented in virtualized manners.

II. Lifecycle: The VNF Manager can create and control the VNF life cycle.

III. Availability: The VNF can always be available to users using different cloud solutions to achieve availability.

IV. Scalability: VNFs can be scaled up or down and can include one or several VNFCs.

V. Location: Having the service provided as software allows its distribution and access from many locations.

VI. Upgrades: We can upgrade a single or several VNFs without affecting the services to be provided to the user.

VII. Security: different levels of security could be applied to different users and application in order to protect the network graph.

VIII. Modularity: VNFs can have a well-defined interface to allow users to interact with the system.

IX. Performance: We can provide to the consumer a satisfactory level of performance by allocating the necessary resources.

4.5.11 Liabilities

This pattern has the following liabilities:

I. Since some users may require custom-built networks this pattern can’t fulfill their needs and human interaction is required.

II. Securing user data from internal attackers or imposters is another issue that may face VNF.

III. Availability could suffer if the cloud service goes down because the NFV system relies directly on the cloud service provider.

4.5.12 Example Resolved

When Dave started planning the equipment that he will need in order to have networking solutions in his company, instead of using traditional networks he decided to use Virtual Network Functions and subscribed to a virtual network function provider. Doing that allowed Dave to scale up and down his network whenever he needed and to have a large variety of functions to provide his users.

In addition, having a virtual network allowed the network to be deployed instantly in the customer location.

4.5.13 Related patterns

- A pattern for SaaS is given in [50] that shows how different components of SaaS interact with each other as well as external components.
- A pattern for Network Functions Virtualization [36] presented the NFV architecture that shows how to create network services using Software-as-a-Service (SaaS) of the cloud this pattern represents the base for NFV architecture other pattern related to NFV can be added to it to enhance it and illustrate the system more.
- Cloud ecosystem [9] shows how the NFV pattern interacts with the different parts of the ecosystem.
- A pattern for an NFV Virtual Machine Environment [28]: shows this environment and how it is related to NFV, as well as describing how the NFV architecture interacts with the virtual environment.
- A pattern for NFV Management and Orchestration (MANO) is given in [30] which represent MANO unit and how it takes control of different components of NFV system.
- A Reference Architecture for NFV is provided in [37]: shows the architectural components of NFV.
- A Pattern for NFV Infrastructure is given in [27] that shows the infrastructure where NFV services are deployed.

4.6 THE PROPOSED REFERENCE ARCHITECTURE FOR NFV

In this chapter, we present the RA that illustrates the architecture of NFV systems. Figure 4.21 shows the patterns that contributed in building this RA. Since we explained all these patterns in every subsection, we won't go into details here. This RA can be extended into SRA by adding to it misuse and security patterns [24].

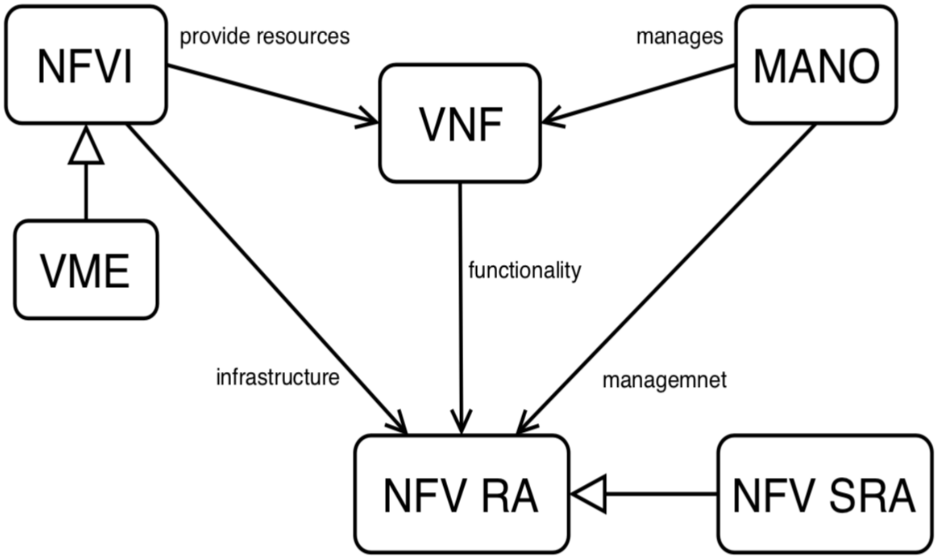
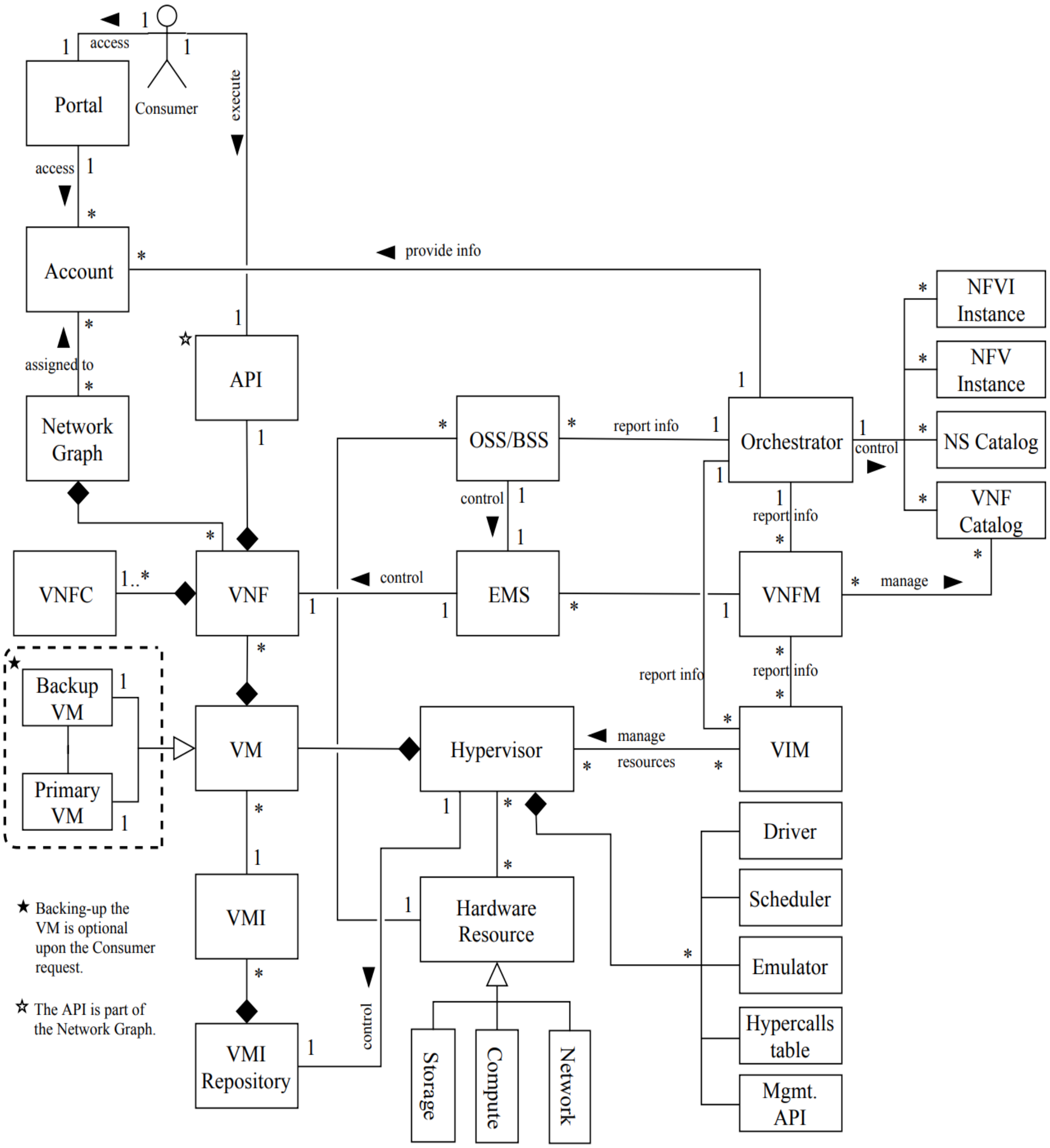


Figure 4.20 Pattern Diagram of NFV RA



★ Backing-up the VM is optional upon the Consumer request.

☆ The API is part of the Network Graph.

Figure 4.21 The Reference Architecture for NFV

Now that we have a good understanding of how the fundamental components of NFV work we are ready to present the RA of NFV in Figure 4.22. This RA shows the global view of the NFV environment. Each architectural component is shown in addition to the interactions between its components. The portal is the gate to access the NFV services by the system consumers, where they will get connected to the accounts they own. Consumer accounts are connected to network graphs which represent the networks the consumer can use. This network graph is created from one or several VNFs, as it was explained in detail in section 4.2.4. The hypervisor was presented in the VME pattern, and it plays an important role in the NFV system since it controls the VMs of the entire VME (see subsection 4.2.2.) The API is an interface to VNFs from the user has access to the virtual. Orchestrator, VNFM, and VIM are all parts of MANO pattern that has been explained in subsection 4.2.3.

5 VALIDATION OF THE ARCHITECTURAL MODELS

Reference architecture modeling has been proven to be a powerful mechanism to present and understand complex systems. However, RA is considered an abstract model, and such models cannot be evaluated in terms of performance, reliability, or security using traditional experimentation or testing methods [11]. On the other hand, other criteria could be used to validate our RA and patterns. We can achieve validation of abstract models by comparing our RA model to models of commercial NFV systems, explicitly comparing the fundamental components of our NFV RA with the elements of these systems; keeping in mind that some of these systems use additional components as add-on to support their functions that are not essential. First, our patterns include “Known Uses” subsections, which indicate examples of commercial NFV models that have the same critical components necessary to ensure network service operability. For instance, we have found that our NFVI model can be mapped to five commercial models which are Ericsson, Nokia, Cisco, Alcatel, and Huawei. Second, we can show the usefulness of our models by showing what can be done with them where they can be implemented. All our patterns also contain an “Implementation subection” that indicates software-related aspects and how the pattern features can be realized. For example, in the VNF pattern, we showed how a system could apply VNFs as well as the life cycle of VNF from creation to termination. Moreover, we have presented our work in several architectural platform meetings, such as Pattern Languages of Programming (PLoP) conferences, where we received significant feedback on the usefulness and benefits of this work.

In terms of precision, we created our RA through UML modeling and a well-defined pattern template, which is a method widely used and approved in software engineering as more precise as compared to other representations such as textual or block diagrams.

We decided to compare our work to three different platforms that offer NFV services, noting that all these platforms are still under evaluation, and their architecture may be changed in the future. Further, we will also refer to the Global System for Mobile Communications Association (GSMA) best practices and requirements for NFV [68] [69]. GSMA unites more than 750 operators with almost 400 companies in the broader mobile ecosystem, including handset and device makers, software companies, equipment providers, and internet companies, as well as organizations in adjacent industry sectors. In their NFV requirements and best practice, they suggested that any NFV system should at least include: NFVI layer, MANO unit, VNF packages, and OSS/BSS. All the requirements that GSMA suggested to have in an NFV system are included in our RA, which proves the completeness of our RA.

5.1 RED HAT NFV PLATFORM

Red Hat is considered one of the world's leading providers of open-source software solutions. One of their products is Red Hat Network Functions Virtualization platform, which is a platform built on Red Hat Enterprise Linux

OpenStack Platform. It enables convenient deployment of NFV and VNFs services from multiple suppliers [70]. Although their NFV solutions include some of their own add-ons that are not included in our RA, such as Red Hat OpenStack as a virtualization platform. Other fundamental components that are necessary for NFV system to be up and running are included, such as VNFs, NFVI layer, and MANO unit, which are all represented by standalone patterns and are part of our final RA. Figure 5.1 shows the Red Hat NFV platform.

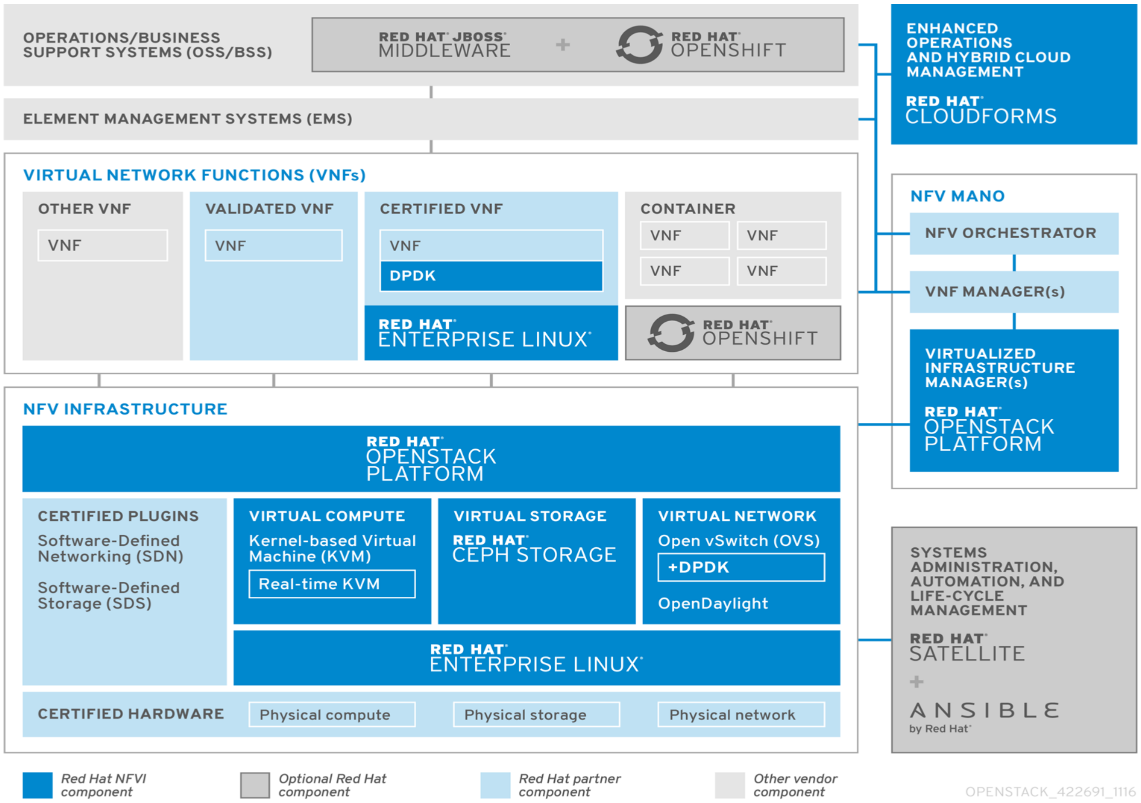


Figure 5.1 Red Hat NFV Platform

5.2 OPEN PLATFORM FOR NFV (OPNFV)

OPNFV is a project of the Linux Foundation. OPNFV aims to provide an open-source platform to apply NFV services with the help of other open-source components such as OpenStack and open vSwitchs [71]. OPNFV works closely with ETSI, so their architecture reflects on the architecture proposed by ETSI. The support of their platform is highly active; they also have their own technical forum to address performance, interoperability, security, and scalability concerns and issues among users/developers of the service. They promise to have a twice-yearly release cycle for their product. However, OPNFV does not seek to provide a distribution for a full NFV system; instead, they provide an open-source platform to build NFV systems. Figure 5.2 shows its NFV architecture.

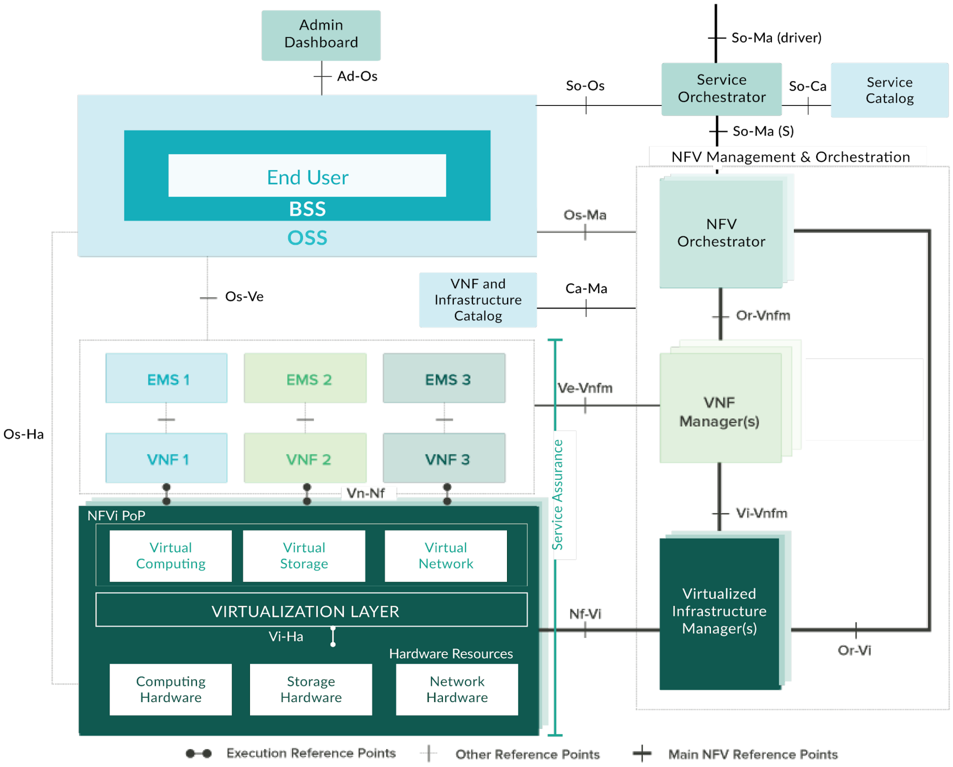


Figure 5.2 OPNFV Architecture

5.3 VCLOUD NFV

Vcloud is VMWare's solution for network operators to deliver NFV system to their customers by providing complete carrier-grade cloud infrastructure with capabilities to host NFV services. Vcloud NFV delivers a platform to support Communication Service Providers (CSPs)[72]. Vcloud uses its own VMware OpenStack virtual infrastructure manager to control the NFV system [73]. Figure 5.3 shows its architecture for the NFV system. Although they use their add-ons in building their architecture, they still have the fundamental components to create an NFV system, which makes it an excellent candidate to be compared to our NFV.

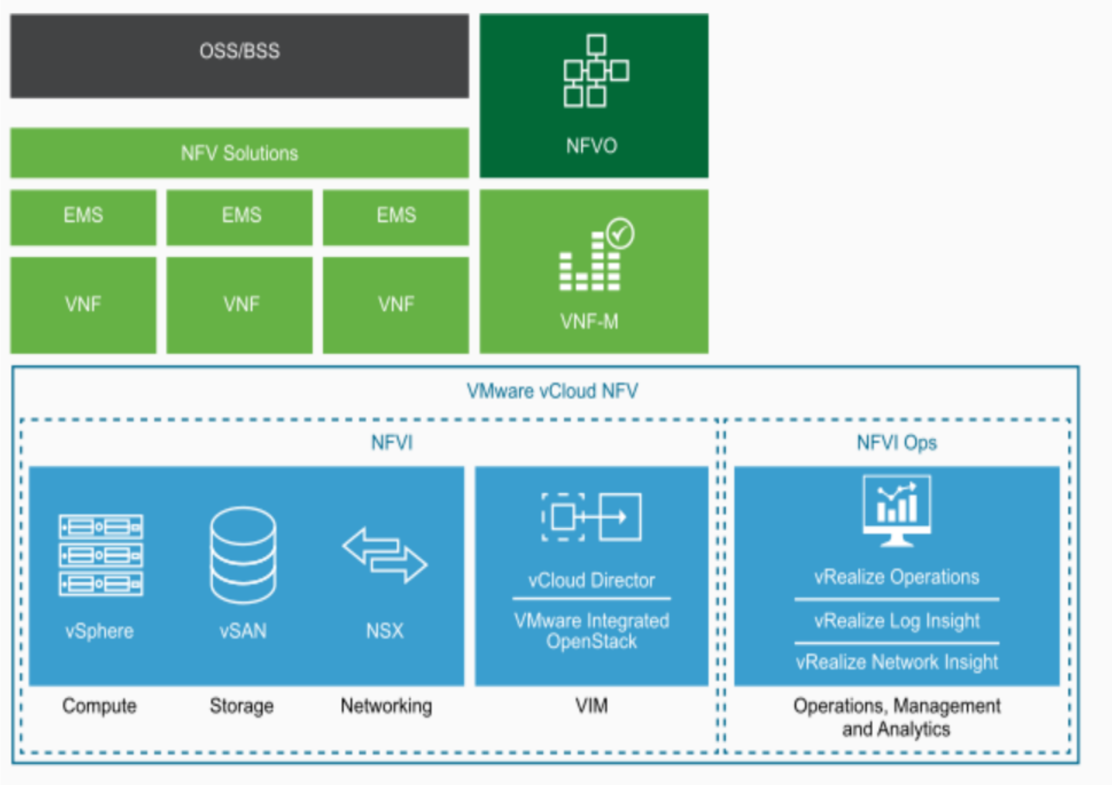


Figure 5.3 Vcloud Architecture

Now that we listed some of the actual products that provide NFV services, we will show the validation of our RA through considering it against the systems we listed. We will list the fundamental components of our RA as a column where the system we will compare to it will be presented as rows. If an element is also found in our RA, a “✓” will be placed. Table 5.1 shows the components of the NFV systems as well as our RA blocks. We noted that some of the models include all the elements of NFV but are arranged in a different way than in the ETSI standards [16]. For example, in Vcloud architecture the virtual infrastructure manager is part of the NFVI layer instead of being part of the MANO unit [72]. The reasons behind that are that they provide users with their own NFVI to build NFV systems from it. Therefore, to ensure that the system works according to their specification, they let the VIM be part of their system to control the infrastructure.

Table 5.1 Validation of RA for NFV

components system	Orchestr ator	VNFM	VIM	OSS/BSS	EMS	VNF	NFVI	NFVI Instance	NFV Instance	VNF Catalog	NS Catalog
RA for NFV	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Other NFV framework compared with our RA											
Red Hat NFV platform	✓[1]	✓[1]	✓	✓	✓	✓	✓				
OPNFV	✓	✓	✓	✓	✓	✓	✓			✓	✓
VCloud NFV	✓	✓	✓	✓	✓	✓	✓				
GSMA suggestions	✓	✓	✓	✓	✓	✓	✓				✓

Since our RA is based on the ETSI standardized architecture, it contains all the fundamental components needed to set up an NFV service. When we compare our RA to Red Hat NFV platform, we notice that all the components, even the components provided by third parties to support their system, are also included in our RA. This also applies to OPNFV and vCloud NFV components, as well. In fact,

[1] Provided by third party partner

our RA consists of some detailed internal components such as NFVI and NFV instances, which are included in the ETSI standard [16].

6 ANALYSIS OF USE CASES ACTIVITY MISUSE

In this chapter, we consider some of the listed use cases for NFV in subsection 3.2 and we analyze their activity flow. Having a good understanding of the system activities could help us in detecting and securing the system against potential threats. First, we systematically analyze the use cases activities and consider each activity in every use case [74]. From this analysis, we enumerate threats and select appropriate security policies to mitigate these threats, as well as relate the threats to their source, whether it's from an insider or an outsider attacker [75]. The use case activities are represented using UML activity diagrams extended with dashed rounded rectangles to represent threats and dashed lines to represent misuse related control flows. Note that we combined two use cases in one example.

6.1 USE CASES “OPEN AN ACCOUNT” AND “REQUEST NETWORK SERVICE”

Using a UML activity diagram, we analyze two fundamental use cases for the NFV life cycle which are "Opening an account" and "requesting a network service". Opening an account is a primary activity in NFV because without a valid account a user can't consume a service. On the other hand, creating a network service is the main activity performed by the user registered for service. Figure

6.1 illustrates the activity diagram for these two use cases showing possible misuses activities. Note that these two use cases must be done in this order but not necessarily one comes immediately after the other.

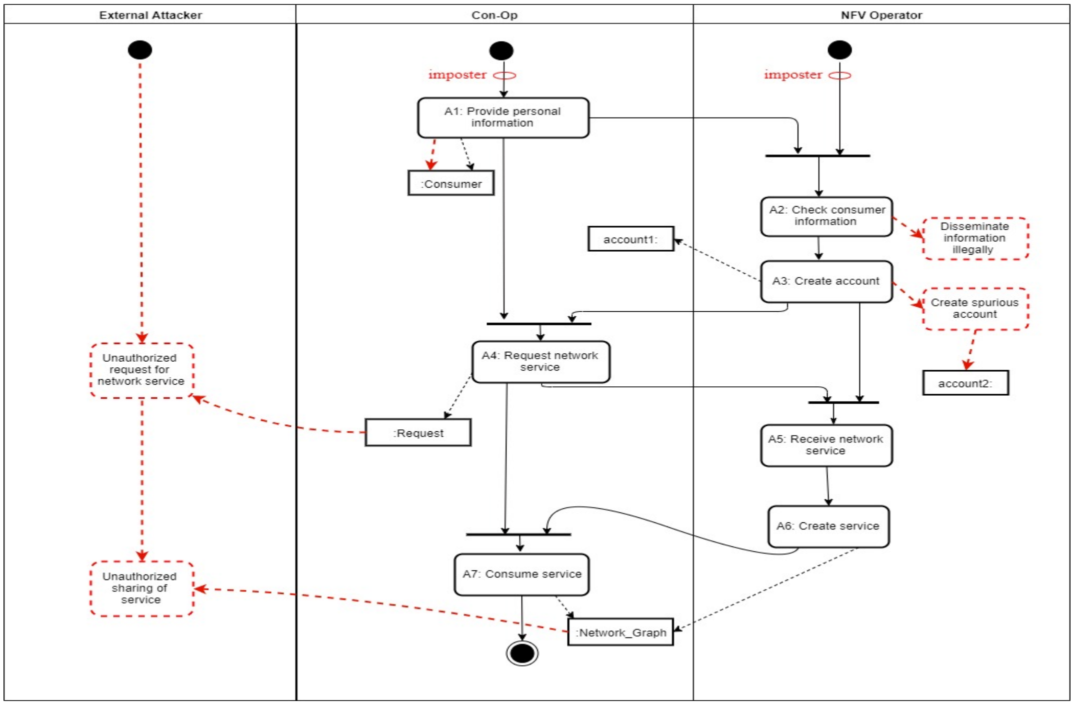

External Attacker
Con-Op
NFV Operator
imposter
imposter
A1: Provide personal information
:Consumer
A2: Check consumer information
Disseminate information illegally
account1:
A3: Create account
Create spurious account
account2:
Unauthorized request for network service
A4: Request network service
:Request
A5: Receive network service
A6: Create service
Unauthorized sharing of service
A7: Consume service
:Network_Graph

Figure 6.1 Activity diagram for open account and request a network service use cases

Now that we analyzed the flow of events for the two use cases and illustrated the activities to be performed by the actors of the system and how an attacker of the system could misuse it we list in Table 6.1 some of the possible threats for these use cases that we identified through the activity diagram of the two use cases whether it's performed by an insider attacker or an outsider attacker and relate each threat to the activity it belongs to. Later in this subsection we will list some general security policies which can stop and/or mitigate the identified threats in the two use cases. Moreover, we will relate the identified threats to a security attribute to be able to find a suitable solution for that type of threat.

Table 6.1 Threats for open account and request network service use cases

Threats		
A2	T2.1	NFV operator is an imposter who disseminates consumers’ information illegally.
A3	T3.1	NFV operator is an imposter who creates a spurious account using a consumer’s information.
A4	T4.1	An attacker sends an unauthorized requests for network service.
A4	T4.2	An attacker tries to prevent consumers from using the service (DoS)
A7	T7.1	An attacker sharing the resources of consumers in an unauthorized way.
A7	T7.2	An attacker tries to prevent consumers from using the service (DoS)

6.2 USE CASE “MODIFY NETWORK RESOURCES”

One of the main features of NFV is its flexibility compared to the legacy network. In NFV, users can request at any time a modification of the resources they are using according to their needs, whether it's scaling the network up or down, which makes this use case one of the fundamental use cases of the NFV life cycle. The process of modifying resources for NFV is done merely by request from the user to the NFV operator, which in turn alters the network to match the user’s need and sends a modification acknowledgment back to the user. However, during this process, several threats and misuses could happen. Figure 6.2 illustrates the activity diagram for modifying network resources showing some possible misuses by an attacker.

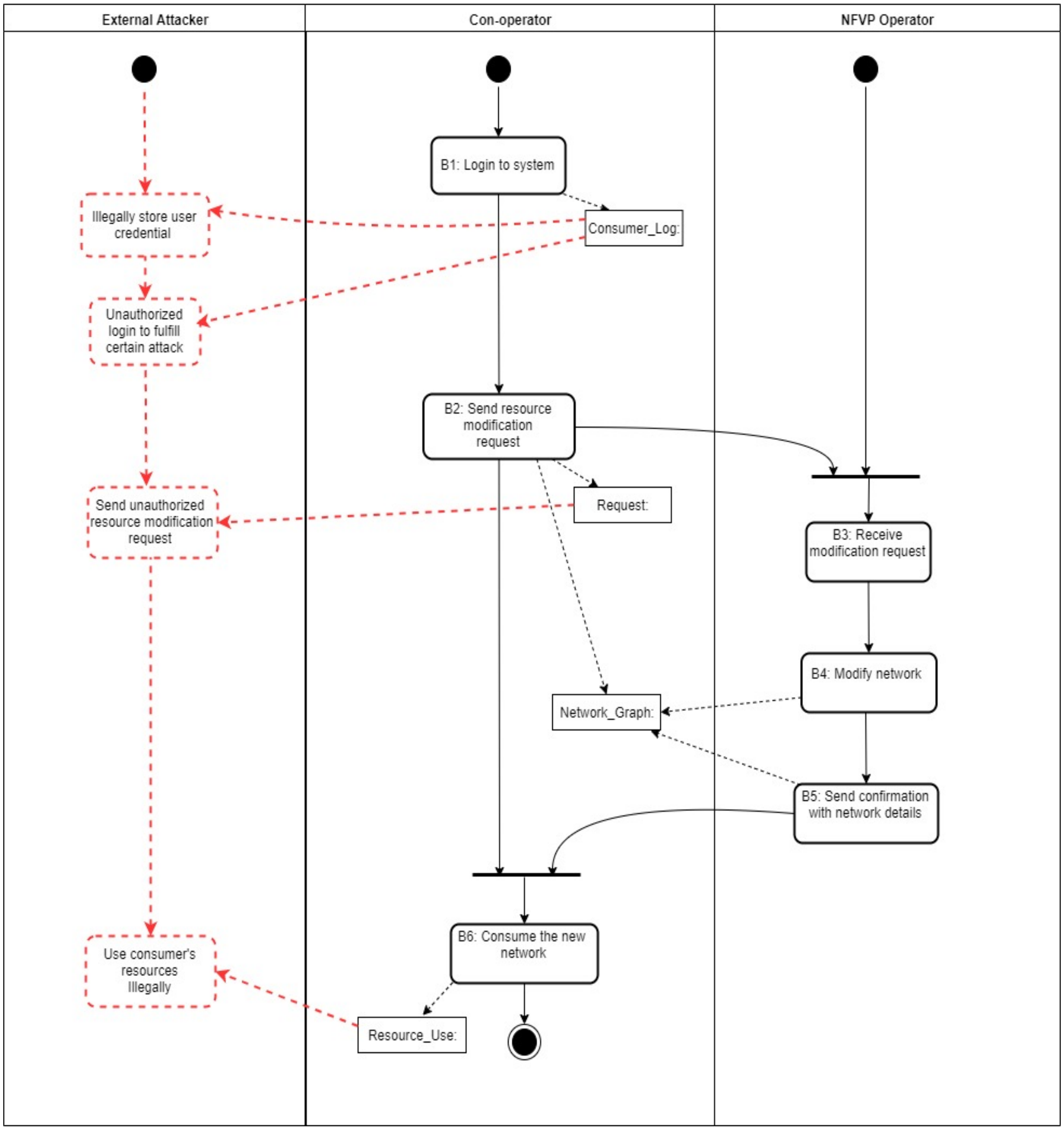

External Attacker
Con-operator
NFVP Operator
B1: Login to system
Illegally store user credential
Consumer_Log:
Unauthorized login to fulfill certain attack
B2: Send resource modification request
Request:
Send unauthorized resource modification request
B3: Receive modification request
B4: Modify network
Network_Graph:
B5: Send confirmation with network details
B6: Consume the new network
Use consumer's resources Illegally
Resource_Use:

Figure 6.2 Activity diagram for modify network resource use case

Through a systematic analysis of the use case and its activity, we identified several threats that could affect the flow of the system denoted by T1 to T6 in Table 6.2, which shows some additional threats.

Table 6.2 Threats for open account and request network service use cases

Threats		
B1	T1.1	An attacker saves user information and credential in unauthorized manners
B1	T1.2	An attacker disseminates consumers' data.
B1	T1.3	An attacker illegally logs-in to consumer account to fulfill certain attack.
B2	T2.1	An attacker uses the consumer data to send unauthorized resource modification request.
B2	T2.2	An attacker performs DoS preventing consumers to create account.
B6	T6.1	An attacker illegally uses consumers network resources.
B6	T6.2	An attacker performs DoS preventing consumers to create account.

We won't discuss countermeasures for the attacks in detail in this dissertation since it's out of the main scope of it. However, we could consider some security policies that could be applied to any complex system including NFV systems. Although security policies don't provide a definitive solution to stop these threats, they still provide guidelines for any organization or service provider to prevent or mitigate these threats. In the previous section, we listed some of the primary security threats. To further enhance the analysis of the system and provide a better judgment of misuse activities, we should consider different security attributes and connect these attributes to each activity. The standard security attributes are:

- Confidentiality: which refers to the restriction of access to sensitive information related to both the system and the user.
- Confidentiality includes traffic analysis, inference and information disclosure.

- Integrity: This ensures that the data provided are valid and not corrupted as well as only accessible or modified by those authorized to do so. Integrity protection includes unauthorized data modification or destruction.
- Availability: which refers to the ability to access the system and using the service at all times, which is a critical aspect of cloud-based systems. Availability can be compromised by the denial of service and disruption.
- Accountability: which refers to tracing the activity performed on the system as well as the source of that activities. Accountability attacks include track erasing and repudiation.

7 RELATED WORK

The concept of NFV has been dealt frequently in the literature. Most work describes how and where the NFV could be deployed, and what are the architectural requirements needed to virtualize the network functions, conforming to the NFV framework of the ETSI. In [76], the authors defined primary NFV application cases using UML and explained the requirements needed to design and deploy Network Functions-as-a-Service (NFaaS) over a virtualized infrastructure. In their approach, they adopted the high-level virtualization architecture of T-NOVA, an integrated research project focusing on NFV [77], to define the business scenarios, roles, and stakeholders. However, their focus was from a business and marketplace perspective.

Furthermore, several white papers have been written by ETSI [78], [79], [80] to explain the fundamental architecture of the NFV system and how each component created; yet these papers lack semantics and don’t cover the topic of having a complete RA for NFV. Also, SDXCenteral has several reports that discuss the architecture and standard of NFV architecture [13], [81], [82], [83].

In [36], the authors use architectural modeling to describe NFV architecture. To our knowledge, this is the only attempt to present the NFV system using architectural patterns. Their pattern describes the general architecture of NFV, in which the service providers can build network functions using IaaS and PaaS then provide it as SaaS. However, their pattern didn’t include all the NFV components; instead, it just describes the concept of NFV.

Further, different technical organizations presented architectural presentations for the NFV system. In [72], the authors provided an RA for vCloud NFV platform. Their reference architecture covers the leading platform and how to provide a solution for NFV providers by providing them with an infrastructure ready to host NFV service. Yet, their work doesn’t show the details of the integration between the different components of NFV and external systems components [84].

In [85], the authors presented an open-source NFV project (Open-NFV) started by Linux foundation and include different leading companies in networking and telecommunication industry such as AT&T, Orange SA, Telecom Italia, etc [12]. The goal of this project was t6o bring companies interested in this innovation together to speed up the deployment and development of NFV for enterprise and service providers.

RAs are proven to be a successful way to present complex systems. Several authors used reference architecture as a powerful tool to represent and understand complex systems and their architecture [86]. Further, in [11], the authors used architectural modeling to develop a reference architecture for cloud system; their architecture defined the three different cloud models, which are Infrastructure as a Service (IaaS), Software as a Service (SaaS), and Platform as a Service (PaaS). Also, in [87], the authors created a reference architecture for container ecosystems that combine several systems that rely on the cloud, such as the Internet of Things (IoT) and fog computing.

In [10], the authors used patterns to create an RA for the mobile shared workspace (MSW). Their design aims to present a reusable design solution for mobile shared workspaces.

In [88] the authors presented an architectural design pattern towards open cloud-based 5G communications. Cloud computing and NFV are considered part of the 5G ecosystem and are essential to achieve design concepts for 5G, such as Network slicing.

Moreover, architectural modeling has also been used to study NFV threats. In [89], the authors designed a misuse pattern for privilege escalation in the NFV system. Privilege escalation is a serious threat that could jeopardize the privacy of users of the system or even cause the system to be unavailable altogether. In [90], the authors discussed another threat, which is VM escape, that could potentially harm NFV systems. Finally, in [91], the authors designed a misuse pattern for DDoS attacks in NFV systems DDoS attacks are dangerous toward such systems because they may make the system unavailable to users, which will affect the services to be delivered to them.

8 CONCLUSIONS AND FUTURE WORK

Network Function Virtualization is a new network paradigm that aims to deliver network functions as software in a virtualized manner. This technology promises several benefits for network providers and users. Some of NFV benefits are reducing costs of purchasing, installing and maintaining network equipment, faster deployment time, flexibility in terms of scaling up or down the network, and having access to different independent networking software, including open-source software by the community [92]. However, in order to benefit from the full potential of NFV, we need to fully understand how the internal components of the system interact with each other as well as with other external systems, which create even more challenges because, in NFV, the network functions could be delivered by several vendors. Moreover, integrating an NFV system with legacy network systems increases the complexity of the system architecture. Through a literature review, we found that the existing NFV architectures lack semantics. To address the heterogeneity and complexity of the integration of NFV, we proposed the use of abstract architectural models. We found that abstract architectural models such as patterns and RAs are powerful tools to address these issues as they describe the architectural components of systems in the highest level of abstraction. More precisely, the model-based approach has been proven to be very valuable when it comes to complex environments and systems such as NFV systems [93] [11]. To our knowledge, this is the first time where patterns and RAs are used to represent and analyze NFV and its components.

Fully understanding the potential of NFV services will not only help us to benefit from its capabilities, but it will also help us build and integrate other complex systems; for example, mobile 5G network architectures are very complex, and NFV technology could help in building and achieving their requirements. 5G standards promise a high-speed and reliable bandwidth with stable coverage; according to Nokia Bell Labs, the number of IoT devices connected to mobile networks in 2014 was 1.6 billion devices, and it is expected to exceed 20 billion devices by 2020 [94]. For mobile operators to be able to handle the ever-increasing consumer and business connectivity demands, it is natural to turn to advanced networking technological solutions such as NFV and SDN to be able to deliver effective service. In the core of the 5G network, a technique called network slicing will be used to support multiple virtual networks over one physical network infrastructure [95]. Virtualizing various appliances of the 5G network and applying network slicing techniques make NFV a key component in building 5G mobile networks. Moreover, NFV enables a distributed cloud to provide flexible and programmable network support for 5G networks. 5G NFV will allow a physical network to be divided into various virtual networks capable of supporting multiple radio access networks (RANs). NFV can also address barriers to 5G by optimizing resource provisioning of the virtual network functions (VNFs) for price and energy, scale VNFs and ensure that VNFs consistently operate properly [96]. Many network operators believe NFV is the key technology for enabling 5G networks since NFV fulfill 5G environment requirements [97].

In this dissertation research, we have made the following contributions:

I. We surveyed the literature for reference architectures for NFV systems and their related patterns.

II. We created a pattern for Virtual Machine Environment (VME) of NFV that shows hypervisor of NFV that is responsible for creating and managing VMs in which VNFs are running on [28].

III. We created a pattern for Network Function Virtualization Infrastructure (NFVI) that shows the infrastructure layer of NFV system [27].

IV. We created a pattern for Virtual Network Function (VNF) that shows how network functions are created and managed in virtualized manners [29].

V. We created a pattern for Management and Orchestration (MANO) unit that shows how NFV systems are managed and controlled [30].

VI. We listed the primary use cases for NFV, as well as the actors that trigger them [37].

VII. We proposed a Reference Architecture for NFV based on the patterns that we created as building blocks [37] .

The RA of the cloud ecosystem could be extended further by combining the RAs of IoT, fog computing, and NFV systems [4].

APPENDIX

LIST OF PUBLICATIONS

1. A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, "A Survey of Network Function Virtualization Security," in *IEEE SoutheastCon 2018*, 2018, doi: 10.1109/SECON.2018.8479121.
2. A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, "Analysis of threats and countermeasures in NFV use cases," in *SysCon 2019 - 13th Annual IEEE International Systems Conference, Proceedings*, 2019, pp. 1–6.
3. A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, "Toward a Reference Architecture for NFV," in *2nd International Conference on Computer Applications and Information Security, ICCAIS 2019*, 2019.
4. A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, "A Pattern for Network Function Virtualization Infrastructure (NFVI)," in *In Proceedings of the 26th PLoP'19*, 2019.
5. A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, "A Pattern for a Virtual Network Function (VNF)," in *The 14th International Conference on Availability, Reliability and Security (ARES 2019)*, 2019.
6. A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, "A Pattern for NFV Management and Orchestration (MANO)," in *Proceedings of the 8th Asian Conference on Pattern Languages of Programs*, 2019.

7. A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “A Pattern for an NFV Virtual Machine Environment,” in *Proceedings of the 13th Annual IEEE International Systems Conference 2019*, 2019.
8. A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “Threats Against the Virtual Machine Environment of NFV,” in *2nd International Conference on Computer Applications and Information Security, ICCAIS 2019*, 2019.
9. A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “A Misuse Pattern for NFV based on Privilege Escalation,” in *Proceedings of the 8th Asian Conference on Pattern Languages of Programs*, 2019.
10. A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “A Misuse Pattern for Compromising VMs via Virtual Machine Escape in NFV,” in *Proceedings of the 14th International Conference on Availability, Reliability and Security (ARES 2019)*, 2019.
11. A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “A Misuse Pattern for Distributed Denial-of-Service Attack in Network Function Virtualization,” in *In Proceedings of the 26th PLoP’19*, 2019.
12. A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “Towards a Security Reference Architecture for Network Function Virtualization,” 2019. (Submitted).

REFERENCES

[1] M. Aiash, G. Mapp, and O. Gemikonakli, “Secure live virtual machines migration: Issues and solutions,” in *Proceedings - 2014 IEEE 28th International Conference on Advanced Information Networking and Applications Workshops, IEEE WAINA 2014*, 2014, pp. 160–165.

[2] ETSI, “Network Functions Virtualisation (NFV); Architectural Framework,” 2014. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.02.01_60/gs_NFV002v010201p.pdf.

[3] N. Sharma, “Hyper-V and VMware vSphere Architectures: Pros and Cons,” 2013. [Online]. Available: https://www.serverwatch.com/server-tutorials/microsoft-hyper-v-and-vmware-vsphere-architectures-advantages-and-disadvantages.html.

[4] Applied Science and Technology Research Institute (ASTRI), “Network Functions Virtualization (NFV) for Next Generation Networks (NGN),” 2016.

[5] L. Radware, “Network Functions Virtualization Role of ADCs in Network Functions Virtualization (NFV)-Whitepaper,” 2014. [Online]. Available: https://networkbuilders.intel.com/docs/Radware_NFV_WP.pdf.

[6] International Telecommunication Union, “Measurement method for energy efficiency of network functions virtualization,” 2018. [Online]. Available: https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-L.1361-201811-I!!PDF-E&type=items.

[7] Z. Xu, F. Liu, T. Wang, and H. Xu, “Demystifying the energy efficiency of Network Function Virtualization,” *2016 IEEE/ACM 24th Int. Symp. Qual. Serv. IWQoS 2016*, 2016.

[8] M. Chiosi *et al.*, “Network Functions Virtualisation: An Introduction, Benefits, Enablers, Challenges & Call for Action,” in *SDN & OpenFlow World Congress*, 2012.

[9] E. Fernandez, N. Yoshioka, H. Washizaki, and M. Syed, “Modeling and Security in Cloud Ecosystems,” *Futur. Internet*, vol. 8, no. 4, p. 13, Apr. 2016.

[10] J. Rodríguez-Covili *et al.*, “Towards a reference architecture for the design of mobile shared workspaces,” *Futur. Gener. Comput. Syst.*, vol. 27, no. 1, pp. 109–118, Jan. 2011.

[11] E. B. Fernandez, R. Monge, and K. Hashizume, “Building a security reference architecture for cloud systems,” *Requir. Eng.*, vol. 21, no. 2, pp. 225–249, Jun. 2016.

[12] F. A. Braz, E. B. Fernandez, and M. VanHilst, “Eliciting security requirements through misuse activities,” *Proc. - Int. Work. Database Expert Syst. Appl. DEXA*, pp. 328–333, 2008.

[13] SdxCentral, “2017 NFV Report Series Part I Foundations of NFV : NFV Infrastructure and VIM,” 2017. [Online]. Available: https://en.resources.lenovo.com/analyst-reports/sdx-central-2018-nfv-report-series-nfv-infrastructure-nfvi-and-vim.

[14] ETSI, “Network Functions Virtualisation (NFV); Infrastructure; Hypervisor Domain,” 2015. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/004/01.01.01_60/gs_nfv-inf004v010101p.pdf.

[15] ETSI, “Network Functions Virtualisation (NFV); Infrastructure Overview,” 2015. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/001/01.01.01_60/gs_NFV-INF001v010101p.pdf.

[16] ETSI, “Network Functions Virtualisation (NFV); Management and Orchestration; Or-Vnfm Reference Point - Interface and Information Model Specification,” 2016. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-IFA/001_099/007/02.01.01_60/gs_NFV-IFA007v020101p.pdf.

[17] F. Buschmann, R. Meunier, P. Sommerlad, and M. Stal, *Pattern-Oriented Software Architecture Volume 1: A System of Patterns*. Wiley, 1996.

[18] E. Fernandez, J. Pelaez, and M. Larrondo-Petrie, “Attack Patterns: A New Forensic and Design Tool,” in *Advances in Digital Forensics III*, New York, NY: Springer New York, 2007, pp. 345–357.

[19] E. B. Fernandez, *Security patterns in practice: designing secure architectures using software patterns*. J. Wiley & Sons, 2013.

[20] S. Angelov, P. Grefen, and D. Greefhorst, “A framework for analysis and design of software reference architectures,” *Inf. Softw. Technol.*, vol. 54, no. 4, pp. 417–431, Apr. 2012.

[21] P. Avgeriou, “Describing, Instantiating and Evaluating a Reference Architecture: A Case Study,” *Default J.*, 2003.

[22] R. N. Robert Cloutier, ,* Gerrit Muller, Dinesh Verma, E. Hole, and and M. Bone, “The Concept of Reference Architectures,” *Wiley Intersci.*, 2009.

[23] M. Pankowska, “Stakeholder Oriented Enterprise Architecture Modelling,” in *Proceedings of the 12th International Conference on e-Business*, 2015.

[24] A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “Towards a Security Reference Architecture for Network Function Virtualization,” 2019.

[25] NIST, “NIST Cloud Computing Standards Roadmap,” *NIST Spec. Publ. 500-291*, 2013.

[26] T. C. Lethbridge and R. (Robert) Laganière, *Object-oriented software engineering : practical software development using UML and Java*. McGraw-Hill, 2001.

[27] A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, “A Pattern for Network Function Virtualization Infrastructure (NFVI),” in *In Proceedings of the 26th PLoP’19*, 2019.

[28] A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “A Pattern for an NFV Virtual Machine Environment,” in *Proceedings of the 13th Annual IEEE International Systems Conference 2019*, 2019.

[29] A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, “A Pattern for a Virtual Network Function (VNF),” in *The 14th International Conference on Availability, Reliability and Security (ARES 2019)*, 2019.

[30] A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, “A Pattern for NFV Management and Orchestration (MANO),” in *Proceedings of the 8th Asian Conference on Pattern Languages of Programs*, 2019.

[31] M. Veeraraghavan, T. Sato, M. Buchanan, R. Rahimi, S. Okamoto, and N. Yamanaka, “Network Function Virtualization: A Survey,” *IEICE Trans. Commun.*, vol. E100, no. 11, 2017.

[32] Ericsson, “Industrializing Network Functions Virtualization with Software-Defined Infrastructure,” 2017. [Online]. Available: https://builders.intel.com/docs/cloudbuilders/industrializing-network-functions-virtualization-with-software-defined-infrastructure.pdf.

[33] Cisco, “Overview to Cisco NFVI,” 2016. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/net_mgmt/network_function_virtualization_

Infrastructure/2_4_1/Cisco_VIM_Install_Guide_2_4_1/Cisco_VIM_Install_Guide_2_4_1_chapter_00.pdf.

[34] Alcatel-Lucent, “Cloudband With Openstack As NFV Platform,” 2014. [Online]. Available: https://www.tmcnet.com/tmc/whitepapers/documents/whitepapers/2014/10694-cloudband-with-openstack-as-nfv-platform.pdf.

[35] Huawei, “Huawei Releases NFVI 100GE Solution Setting a New Standard for NFVI,” 2017. [Online]. Available: https://www.huawei.com/en/press-events/news/2017/2/Huawei-Releases-NFVI-100GE-Solution.

[36] E. B. Fernandez and B. Hamid, “A pattern for network functions virtualization,” in *Proceedings of the 20th European Conference on Pattern Languages of Programs*, 2015, p. 47.

[37] A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, “Toward a Reference Architecture for NFV,” in *2nd International Conference on Computer Applications and Information Security, ICCAIS 2019*, 2019.

[38] J. Shropshire, “Analysis of monolithic and microkernel architectures: Towards secure hypervisor design,” in *Proceedings of the Annual Hawaii International Conference on System Sciences*, 2014.

[39] VMware, “Introduction to VMware vSphere ESX 4.0 ESXi 4.0 vCenter Server 4.0,” 2009. [Online]. Available: http://www.vmware.com/go/patents.

[40] A. Finn and P. Lownds, *Mastering Hyper-V deployment*. Wiley, 2011.

[41] A. Iqbal, N. Sadeque, and R. I. Mutia, “An Overview of Microkernel, Hypervisor and Microvisor Virtualization Approaches for Embedded Systems,” *Lund University*. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.460.4188&rep=rep1&type=pdf.

[42] J. Batall and C. G. Capit, “On the implementation of NFV over an OpenFlow infrastructure : Routing Function Virtualization,” in *SDN4FNS 2013 - 2013 Workshop on Software Defined Networks for Future Networks and Services*.

[43] M. H. Syed and E. B. Fernandez, “The Software Container pattern,” in *Proceedings of the 22nd Conference on Pattern Languages of Programs.*, 2015.

[44] E. Denny, “HP News - Telefónica Selects HP OpenNFV Platform to Build its UNICA Infrastructure,” 2015. [Online]. Available: http://www8.hp.com/us/en/hp-news/press-release.html?id=1923363#.W-mmsnpKh-V.

[45] S. Cooley, “Introduction to Hyper-V on Windows 10 | Microsoft Docs,” 2018. [Online]. Available: https://docs.microsoft.com/en-us/virtualization/hyper-v-on-windows/about/.

[46] Alcatel-Lucent, “Alcatel-Lucent CloudBand: The Platform for NFV,” 2014. [Online]. Available: https://www.tmcnet.com/tmc/whitepapers/documents/whitepapers/2014/9901-alcatel-lucent-cloudband-platform-nfv-release-20.pdf.

[47] Nokia, “CloudBand Infrastructure Software,” 2016. [Online]. Available: https://onestore.nokia.com/asset/200841.

[48] Cisco and Intel, “High-Performance VNFs on Cisco NFV Infrastructure,” 2016. [Online]. Available: https://builders.intel.com/docs/networkbuilders/high-performance_VNFs_on_NFV_infrastructure.pdf.

[49] K. Hashizume, E. B. Fernandez, and M. M. Larrondo-Petrie, “A pattern for software-as-a-service in clouds,” *Work. Redefining Integr. Secur. Eng.*, 2012.

[50] M. H. Syed and E. B. Fernandez, “A Pattern for a Virtual Machine Environment,” *Proc. 23rd Conf. Pattern Lang. Programs*, pp. 1–8, 2016.

[51] K. Hashizume, E. B. Fernandez, and M. M. Larrondo-petrie, “Cloud Service Model Patterns,” in *Proceedings of the 19th Conference on Pattern Languages of Programs*, 2012.

[52] E. Miucci, G. Monteleone, G. F. Andreotti, and P. S. Crosta, “Implementation of VNF Descriptor Extensions for the Lifecycle Management of VNFs,” vol. 10, no. 3, pp. 107–116, 2017.

[53] K. Katsalis, N. Nikaein, and A. Edmonds, “Multi-Domain Orchestration for NFV: Challenges and Research Directions,” in *2016 15th International Conference on Ubiquitous Computing and Communications and 2016 International Symposium on Cyberspace and Security (IUCC-CSS)*, 2016.

[54] Broadband Forum, “A Framework for Virtualization,” 2016. [Online]. Available: https://www.broadband-forum.org/download/TR-359.pdf.

[55] Intel, “End-to-End Service Instantiation Using Open-Source Management and Orchestration Components Introduction and Problem Statement,” 2016. [Online]. Available: http://osm.etsi.organdcontinuetheevolutionoftheproject.

[56] A. Hoban, A. Israel, C. Boyer, and M. Harper, “Open Source MANO,” *OSM Release Four Technical Overview*, 2018. [Online]. Available: https://osm.etsi.org/images/OSM-Whitepaper-TechContent-ReleaseFOUR-FINAL.pdf.

[57] Anuta Networks, “Accelerate Service Delivery with Network Service Orchestration,” 2016. [Online]. Available: https://anutanetworks.com/wp-content/uploads/2017/05/NCX-Platform-Overview.pdf.

[58] F. Slim, F. Guillemin, A. Gravey, and Y. Hadjadj-Aoul, “Towards a dynamic adaptive placement of virtual network functions under ONAP,” in *2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2017, pp. 210–215.

[59] Blue Planet, “Blue Planet SDN and NFV are Changing the Game.” [Online]. Available: https://www.blueplanet.com/products/nfv-orchestration.html.

[60] R. Mijumbi, J. Serrat, J.-L. Gorricho, S. Latré, M. Charalambides, and D. Lopez, “Management and Orchestration Challenges in Network Function Virtualization,” *IEEE Commun. Mag.*, vol. 54, no. 1, 2016.

[61] P. Quinn and T. Nadeau, “Problem Statement for Service Function Chaining,” *Ietf Rfc*, vol. 53, no. 9, pp. 1689–1699, 2015.

[62] G. M. Yilma, Z. Yousaf, V. Sciancalepore, and X. Costa-Perez, “On the Challenges and KPIs for Benchmarking Open-Source NFV MANO Systems: OSM vs ONAP,” *NEC Laboratories Europe GmbH*, 2019. [Online]. Available: https://arxiv.org/pdf/1904.10697.pdf.

[63] Cisco, “Cisco Cloud Services Router and Google Cloud Platform Solutions,” 2018. [Online]. Available: https://www.cisco.com/c/dam/en/us/products/collateral/routers/cloud-services-router-1000v-series/white-paper-c11-741200.pdf.

[64] BluePlanet, “Blue Planet SDN and NFV are Changing the Game,” 2016. [Online]. Available: https://media.ciena.com/documents/SDN-NFV-Are-Changing-The-Game-WP.pdf.

[65] F. Yue, “Network Functions Virtualization-Everything Old is New Again,” *F5 Networks*, 2013. [Online]. Available: https://worldtechit.com/wp-content/uploads/2015/07/f5-white-paper-network-functions-virtualization-everything-old-is-new-again.pdf.

[66] F5 Networks, “Virtual Solutions for Your NFV Environment,” 2018. [Online]. Available: https://www.f5.com/pdf/solution-center/network-functions-virtualization-nfv-solution-overview.pdf.

[67] Qosmos, “Qosmos White Paper: Classification as a Virtual Networking Function Component (VNFC),” 2015.

[68] GSMA, “Considerations, Best Practices and Requirements for a Virtualised Mobile Network,” 2019. [Online]. Available: https://www.gsma.com/futurenetworks/wp-content/uploads/2017/05/Virtualisation.pdf.

[69] Intel, “Networking and Communications Realising the Benefits of Network Functions Virtualisation in Telecoms Networks.” [Online]. Available: https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/benefits-network-functions-virtualization-telecoms-paper.pdf. [Accessed: 12-Jan-2020].

[70] RedHat, “Network Function Virtualization with RedHat,” 2017. .

[71] OpenStack, “Accelerating NFV Delivery with OpenStack - Global Telecoms Align Around Open Source Networking Future,” 2016. [Online]. Available: https://object-storage-ca-ymq-1.vexxhost.net/swift/v1/6e4619c416ff4bd19e1c087f27a43eea/www-assets-prod/telecoms-and-nfv/OpenStack-Foundation-NFV-Report.pdf.

[72] VMware, “vCloud NFV OpenStack Edition Reference Architecture,” 2017. .

[73] VMware, “vCloud NFV Reference Architecture,” 2017. .

[74] E. B. Fernandez, “Threat Modeling in Cyber-Physical Systems,” *Proc. - 2016 IEEE 14th Int. Conf. Dependable, Auton. Secur. Comput. DASC 2016, 2016 IEEE 14th Int. Conf. Pervasive Intell. Comput. PICom 2016, 2016 IEEE 2nd Int. Conf. Big Data*, pp. 448–453, 2016.

[75] F. Buschmann, K. Henney, and D. C. Schmidt, *Pattern-oriented software architecture. v. 4, A pattern language for distributed computing*. John Wiley, 2007.

[76] A. Cimmino and J. Carapinha, “Requirements and Use Cases System for Virtualized Network Functions Platforms,” *J. Telecommun. Syst. Manag.*, vol. 03, no. 02, pp. 1–11, Aug. 2014.

[77] T-NOVA, “T-NOVA Project Website.” [Online]. Available: http://www.t-nova.eu/.

[78] ETSI, “Network Functions Virtualisation (NFV) Release 3; NFV Evolution and Ecosystem; Hardware Interoperability Requirements Specification,” 2017. [Online]. Available: https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx.

[79] ETSI, “Network Function Virtualisation (NFV); Reliability; Report on the resilience of NFV-MANO critical capabilities,” 2017. [Online]. Available: https://www.etsi.org/deliver/etsi_gr/NFV-REL/001_099/007/01.01.01_60/gr_nfv-rel007v010101p.pdf.

[80] ETSI, “Network Functions Virtualisation (NFV); NFV Security; Security and Trust Guidance,” 2014. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-SEC/001_099/003/01.01.01_60/gs_NFV-SEC003v010101p.pdf.

[81] SdxCentral, “eBrief: How NFV Unlocks Multi-Cloud Opportunities,” 2019. [Online]. Available: https://www.sdxcentral.com/resources/sponsored/ebriefs/equinix-reconciling-nfv-multi-cloud/.

[82] SdxCentral, “2016 Mega NFV Report Pt. 1: MANO and NFVI,” 2016. [Online]. Available: https://events.windriver.com/wrcd01/wrcm/2016/08/WP-Mega-NFV-Report-Part-I.pdf.

[83] SdxCentral, “2016 Mega NFV Report Pt. 2: NFV and VNFs,” 2016. [Online]. Available: https://www.telco.com/analyst-coverage/311.

[84] VMware, “VMware vCloud® Director^{TM} Infrastructure Resiliency Case Study,” 2013. [Online]. Available: https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/techpaper/vmware-vcloud-directore-infrastructure-resiliency-white-paper.pdf.

[85] Linux, “Paving the Way to Open Source NFV,” 2016. [Online]. Available: file:///Users/MilanKnight/Downloads/OPNFV_WhitePaper_Paving_Way_OpenSource_NFV_101016 (1).pdf.

[86] V. Hubka and W. Ernst Eder, “A scientific approach to engineering design,” *Des. Stud.*, vol. 8, no. 3, pp. 123–137, 1987.

[87] M. H. Syed and E. B. Fernandez, “A reference architecture for the container ecosystem,” in *ACM International Conference Proceeding Series*, 2018.

[88] K. Katsalis, N. Nikaein, E. Schiller, R. Favraud, and T. I. Braun, “5G Architectural Design Patterns,” in *2016 IEEE International Conference on Communications Workshops, ICC 2016*, 2016, pp. 32–37.

[89] A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “A Misuse Pattern for NFV based on Privilege Escalation,” in *Proceedings of the 8th Asian Conference on Pattern Languages of Programs*, 2019.

[90] A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “A Misuse Pattern for Compromising VMs via Virtual Machine Escape in NFV,” in *The 14th International Conference on Availability, Reliability and Security (ARES 2019)*, 2019.

[91] A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “A Misuse Pattern for Distributed Denial-of-Service Attack in Network Function Virtualization,” in *In Proceedings of the 26th PLoP’19*, 2019.

[92] M. Ruffini and F. Slyne, “Moving the network to the cloud: The cloud central office revolution and its implications for the optical layer,” *J. Light. Technol.*, vol. 37, no. 7, pp. 1706–1716, Apr. 2019.

[93] E. B. Fernandez, N. Yoshioka, H. Washizaki, and M. H. Syed, “Modeling and security in cloud ecosystems,” *Futur. Internet*, 2016.

[94] WeldonMarcus, *The future X network: a Bell Labs perspective*, 1st Editio. CRC Press, 2016.

[95] K. Katsalis, N. Nikaein, E. Schiller, A. Ksentini, and T. Braun, “Network Slices toward 5G Communications: Slicing the LTE Network,” *IEEE Commun. Mag.*, vol. 55, no. 8, pp. 146–154, 2017.

[96] B. Chatras, U. S. Tsang Kwong, and N. Bihannic, “NFV enabling network slicing for 5G,” in *Proceedings of the 2017 20th Conference on Innovations in Clouds, Internet and Networks, ICIN 2017*, 2017, pp. 219–225.

[97] B. Canada *et al.*, “Network Functions Virtualisation – White Paper on NFV priorities for 5G,” *ETSI White Pap.*, no. 1, pp. 1–15, 2017.