C O M M O N S D E E D

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

A Thesis
for the Degree of Master of Science

Dynamic Resource Management On Top of SDN and NFV platforms by applying Machine Learning

Khan Talha Ahmed

Department of Computer Engineering

**GRADUATE SCHOOL
JEJU NATIONAL UNIVERSITY**

June 2019

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Dynamic Resource Management On Top of SDN and NFV platforms by applying Machine Learning.

Khan Talha Ahmed
(Supervised by Professor Khi – Jung Ahn)

Submitted to the Department of Computer Engineering and the Faculty of Graduate School of Jeju National University in partial fulfillment of the requirements for the degree of Master of Computer Engineering

2019.06.27

This thesis has been examined and approved.

송 왕 철

Thesis Director, Wang - Cheol Song, Professor, Jeju National University

변 영 철

Thesis Director, Yung - Cheol Byun, Professor, Jeju National University

안 기 중

Thesis Supervisor, Khi – Jung Ahn, Professor, Jeju National University

Department of Computer Engineering

GRADUATE SCHOOL

JEJU NATIONAL UNIVERSITY

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Dedicated to
My dearest parents, loving brothers and sisters, friends and colleagues.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

ACKNOWLEDGMENTS

First and foremost, my humble praise and gratitude to Allah Almighty; the most gracious, the most merciful, for showering His endless blessings on me throughout my life. Many blessings and salutations on Prophet Muhammad (PBUH) who taught and emphasized the importance of learning and seeking knowledge.

I am enormously grateful to my supervisor Prof. Wang-Cheol Song for his continuous guidance, suggestions, and constant interaction throughout my Master's studies at Jeju National University. I would like to extend my sincere gratitude towards my Korean language teacher, Madam Kim Yun Mi.

I offer my humble gratitude to Prof. Khi- Jung Ahn, Prof. Yung-Cheol Byun for their valuable suggestions and extremely important comments during the process of my thesis evaluation. I would like to thank all the professors and teachers for their contributions throughout the program.

I would like to acknowledge the company, cooperation, and help of my friends and colleagues during the tenure of the study. I offer special thanks to Dr. Afaq Muhammad, Dr. MuqeetRehman, Dr. Israr Ahmed, Dr. Jahanzeb Gul, Adeel Rafique, Muhammad Ali Jibran, Khizar Abbas, Jahanzeb Ahmed, Muhammad Tahir Abbas, Hafiz Mutee Ur Rehman, Mudassar Liaq, Saim Satti, and Mr. ViqarAhmed Kiyani for their enormous help and support.

I will be in debt to Shahbaz Raza who has been very kind and supportive to me in all circumstances since the beginning of my stay in South Korea. He was always a supportive hand in many hard times that I faced during my stay. I am greatly thankful for your enormous support.

I owe a debt of special thanks to Javier Diaz and Asif Mehmood for their cordial cooperation for overcoming research problem and providing technical support to overcome dark moments. Their support, encouragement, and continuous help with kindness, and care allowed me to finish this walk of life. We joined Network Convergence Lab on the same day and shared many ups and downs together throughout the journey. I would also like to offer special thanks to my elder brotherFasih-Ullah Khan who has always been my mentor and well-wisher. His deterministic guidance never let my moral down.

It is worth it to mention my dearest parents and sisters, and all the family members. I am most indebted to my father and mother for their guidance love and prayers. I cannot describe their support and love in words. Their faith in me made it possible for me to never stop.

Last but not least my special thanks to Wei-Yu Chen for his open-source code of OAI for cord services and a special thanks for GWAT-13 Materna dataset providers.

Talha Ahmed Khan
June 2019

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Dynamic Resource Management of SDN and NFV platforms by applying Machine Learning

Khan Talha Ahmed

Supervisor: Prof. Wang-Cheol Song

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

ABSTRACT

The upcoming 5th generation network is not only restricted to mobile devices and high bandwidth provision but also will support multiple services with their dynamic resource demands while enabling ultra-reliability and low latency. As a matter of fact, this increased in the domain made it very complex to design, configure and control the next generation platforms. Considering the fact, SDN (Software Defined Networking) and NFV (Network Function Virtualization) enabled programmability and elasticity in network infrastructure and also provided freedom from the dedicated hardware was a major achievement which broke the monopoly of the major networking industry. They are capable to accomplish the dynamic demands of upcoming applications through the programmable capabilities and network slicing i.e. provisioning of multiple virtual resources over a single physical infrastructure. The SDN and NFV based platforms enable slicing and optimized resource provisioning but they still require complex configurations for setting up each slice and it requires several proficient experts for any additional updates. Also, the 5G support of many services resulted in a lot of dynamicity in terms of user demands e.g. number of cars in C-V2X and also in events like Olympic games the traffic stream through the network can raise exponentially and requiring the network to provide tons of resources. Currently, the resource management is done by experts which are not very efficient and is prone to errors requiring a replacement. Considering the complexity of 5G and its requirements manuscript aims to automate configuration management using IBN (Intent-based Networking) application and apply ML(Machine Learning) to manage the next generation platforms for the automatic scaling of resources in accordance with dynamic user demands. The IBN application will simplify the configuration procedure by enabling what to achieve (high-level generic instructions) as input and how to achieve (Policies and Configuration) as its output.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

On the other hand, the ML with a monitoring integration proficiently perceive the current state of the system resources and predicts future utilization to suggest updates in network configuration.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

CONTENTS

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

List of Figures and Tables

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

List of Acronyms

5G	5th Generation
SDN	Software Defined Networks
NFV	Network Function Virtualization
NFV-MANO	Network Function Virtualization management and Orchestration
ETSI	European Telecommunication Standard Institute
NGMN	Next Generation Mobile Networks
IBN	Intent-Based Networking
ONF	Open Networking Foundation
CO	Central Office
CORD	Central Office Re-architected as Datacenter
M-CORD	Mobile-CORD
XaaS	Everything as a Service
XOS	XaaS Operating System
ONOS	Open Networking Operating System
NGPaaS	Next-Generation Platform as a Service
IaaS	Infrastructure as a Service
ML	Machine Learning
MLM	Machine Learning Model
OAI	OpenAir-Interface
OAI-SIM	OAI-Simulator
OAI-EPC	OAI-Evolved Packet Core
NSSF	Network Slice selection Function
vSPGWC	virtual Serving and Provider Gateway Control Plane
vSPGWU	virtual Serving and Provider Gateway Data Plane
vMME	virtual Mobility Management Entity
vHSS	virtual Home Subscriber server

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

eNB	evolved NodeB
RAN	Radio Access Network
OSS	Operation Support System
BSS	Business Support System
TOSCA	Topology and Orchestration Specification for Cloud
UE	User Equipment
SLA	Service Level Agreement
VTN	Virtual Tenant Application
AMF	Access and Mobility Management Function
SMF	Session Management Function
UPF	User Plane Function
ZSM	Zero Touch Network and Service Management

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Chapter 1
Introduction
1.1. Background

The future 5G network has the aim to support not only the mobile devices but also many other services including connected cars, augmented and virtual reality, smart homes, e-health, and many others. Additionally, 5G guarantee not only the bandwidth but it also ensures ultra-low latency and ultra-reliable connectivity depending on the type of service. Furthermore, it also assures to fulfill the dynamic resource demands of users depending on time, event, type of service and position [1] [2]. In order to accomplish the goals of 5G, many platforms have been designed based on NFV-MANO (Network Function Virtualization- Management and Orchestration) architecture. MANO is developed by ETSI (European Telecommunications Standards Institute) both serve as key players in achieving 5G goals. In the first place, it is based on SDN (Software Defined Networks) and NFV (Network Function Virtualization) and hence it enables network slicing which empowers the implementation of multiple virtual slices over a single physical infrastructure [1]. Also, such platforms enable programmability flexibility and scalability i.e. firstly, they enable users to program the network on runtime, secondly they enable customized service model for providers and finally they enable scalable resource allocation dynamically. M-CORD is one of the example platforms that provide the above three properties and is an open reference solution provided by ONF (Open Networking Foundation) also serves as the test-bed for this work [1].

In general, SDN/NFV platforms are based on MANO architecture where from the top it has the NFV orchestrator which consist of multiple network function, slice, instances and resource catalogs which are instantiated using policy and configuration provided by operators. Such

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

policy and configuration define the network properties and change in policy and configuration also changes the network behavior on runtime. Orchestrator replicates the policy/configurations and instantiates the network resources on the physical infrastructure using VNF (virtual network function) managers. It has the VIM (Virtual Infrastructure Manager) that manages the physical resources. The goal of this work is to simplify policy and orchestration of the platforms and automate the updates in resources with the dynamic requirements.

On the other hand, M-CORD enabled the virtualization of CO (Central Office) by moving it to the cloud-datacenter, as a result, it provides freedom from the dedicated hardware [3]. Hence, M-CORD is a single platform that enables optimized QoS and resource allocation in accordance with each service demands. CORD consists of XOS (everything-as-a-service operating system) orchestrator. XOS lies on top of the ONOS (Open Networking Operating system) SDN controller and OpenStack IaaS (Infrastructure as a Service) provider. Basically, XOS enables the user to integrate two types of services first, VNF as a service and second ONOS control application as a service for controlling the network [4]. We will use specific 3GPP (3rd Generation Partnership Project) based LTE modules as VNF's under CORD and specified ONOS applications are provided with the M-CORD package for controlling the network. M-CORD combined with a network slicing scenario combines to form the test-bed for our platform.

Even though M-CORD and other SDN and NFV platforms fulfill the requirement of the next-generation platform and it enables the user to configure the network to fulfill the goals of 5G. But, it requires multiple experts to generate the configuration for the management and deployments; also it requires the involvement of many decision makers per update. The manual procedure for generating policy and configuration is tedious and error-prone. Hence, we developed a simplified configuration management engine in the form of IBN application which

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

provides a high-level interface to insert generic information for the configuration of a network. Also, it simplifies and automates the configuration and update procedures for underground platforms. Technically, IBN-application automates the policy generation for the underground platforms and it determines proper resources depending on the requirement and resource availability.

1.2. Research Problems and Objectives

The broad scope of 5G made it complex to provide and allocate resources for each of the services under the umbrella. However, technology provided the solutions that enable multiple virtually allocate-able resources and they can be scaled on runtime. As a matter of fact, the support for multiple services forced the platforms to optimized allocation of the resources, and use the scalable property at runtime to change resources. The technical requirement of 3GPP as referenced under [7] for orchestration and management of next-generation networks requires us to enable a user to instantiate, monitor and update network resources as per dynamic demands. Also, it suggests providing a platform that can accept user SLA (Service Level Agreement) and can properly allocate resources using the underground platform. Firstly, for such process each network service providing platform requires many tedious configurations, the simplification and automation of those configurations is an esteem requirement. Not only the configuration management but also the time for orchestration of dynamic resources is a complex issue and requires some novel solutions. Hence, this research is focused on applying Machine learning to predict beforehand to dynamically update the system resources so that to improve the smooth user experience and keep the SLA in operation mode. The main objective of the system is to provide a platform that can automatically manage, configure and control the network platforms

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

and is capable of managing dynamic updates using the prediction through Machine Learning. More precisely, the following research problems are investigated:

- **How to automatically configure a large scale network?** This part we implemented an abstraction layer on top of the 5G network as an Intent-Base application. The purpose of the application is to hide the internal details of the underground platforms. It also provides a High-level interface for the users to interact and provide what they require from the underground system. Hence, by the use of intended high-level information, it configures the underground platforms.
- **How to determine resources?** A resource manager is part of the IBN-Tool which determines the amount of resource for each host based on requested and available resources.
- **How to monitor resources in a 5G platform?** A mechanism in the form of analytics application is designed to monitor resources. It fetches a virtual machine resource statistic from VIM/ Cloud providers.
- **When to initiate scaling up of a VM.** The decision is made based on predictions of future resource usage by Machine Learning models and then decide to scale the system for avoiding performance degradations keeping the optimized resource allocation in accordance with SLA.

As a solution for the above outline challenges, the following objectives are considered:

- Investigate the research area of dynamic resource allocation and introducing automation in next-generation networks using ML.
- Develop an IBN-tool which can be used for simplifying configurations that will result in the automation of configuration update procedure.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

- Define new and enhance network approaches that can support resource allocation on the physical plane in form M-CORD based network slicing scenario.
- Develop an ML approach for predicting future system states and a decision engine for suggesting the proper updates.

1.3. Thesis Organization

The core chapters of this thesis are structured as shown in Figure 1.1. They are organized as follows:

- Chapter 2 considers the overview of existing literature on the topics of next-generation network requirements, SDN/NFV platforms, M-CORD and use of ML in the 5G network for dynamic resource control. Also, it explains the automation of 5G configurations.
- Chapter 3 discusses the details of Overall system, IBN-application and M-CORD, and the specially enhanced network slicing test-bed.
- Chapter 4 explains the Monitoring of the test-bed, Machine Learning approaches and Experimental Results of the overall system.
- Chapter 5 Concludes the thesis.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Chapter 2
Related Work

The 5th generation network goal for providing the internet of everything compelled many new applications in its domain. The upcoming applications of 5G require specific changes in the existing networks in terms of technology, architectures, and platforms. Hence, many novel approaches have been proposed by research groups and standardization bodies for achieving the aims of 5G. 3GPP is the most prominent standard body for defining the standard architectures for 5G, it introduced the network slicing and it introduced association of S-NSSAI (Single-Network Slice Selection Instance Information) to describe the requirement of a certain device from the network. The total number of S-NSSAI considered here is 8 however 3GPP only support 3 S-NSSAI for its current version and the 8 different slices are based onQoSrequirementsand specifically it considers latency, bandwidth and reliability parameters [1][2][7][19]. Figure 2.1 shows the details of how services are divided into 8 different groups and is referenced from NGMN (Next Generation Mobile Networks) [2].

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

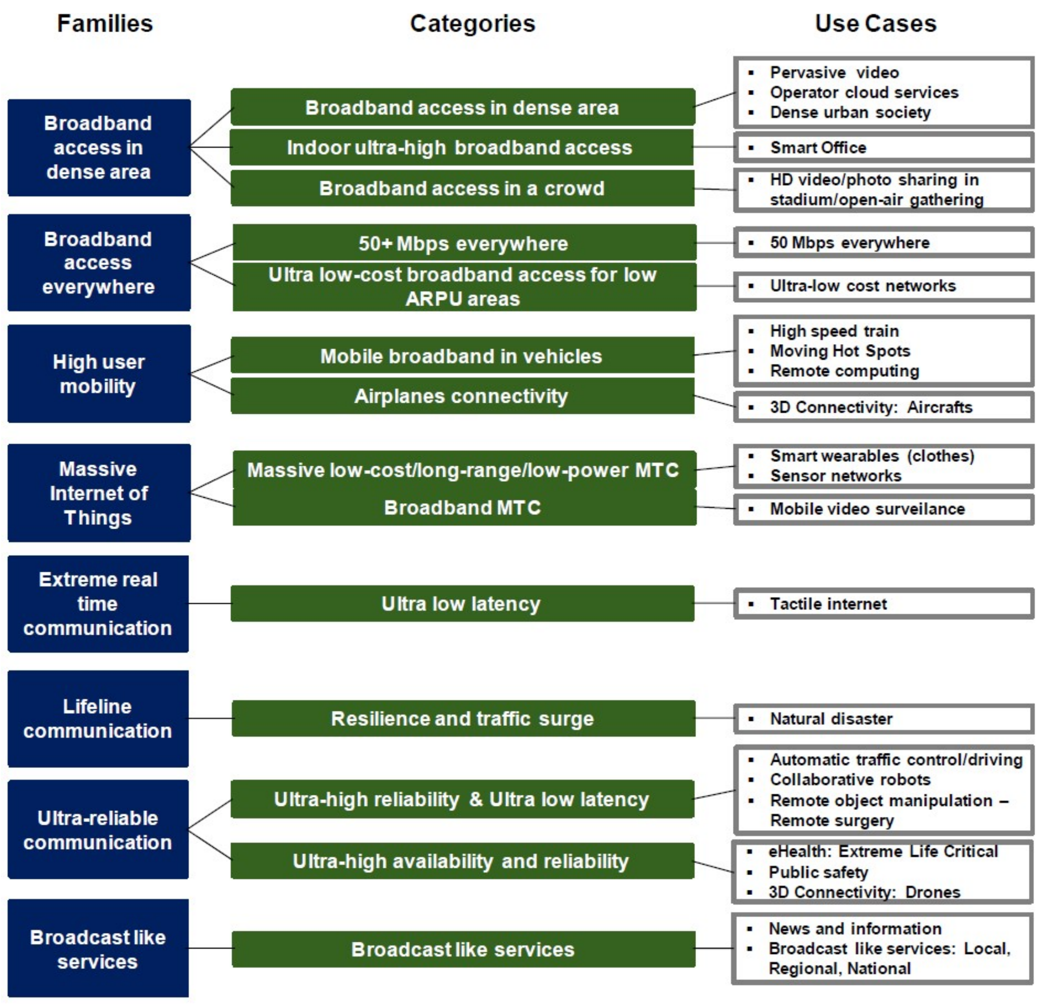
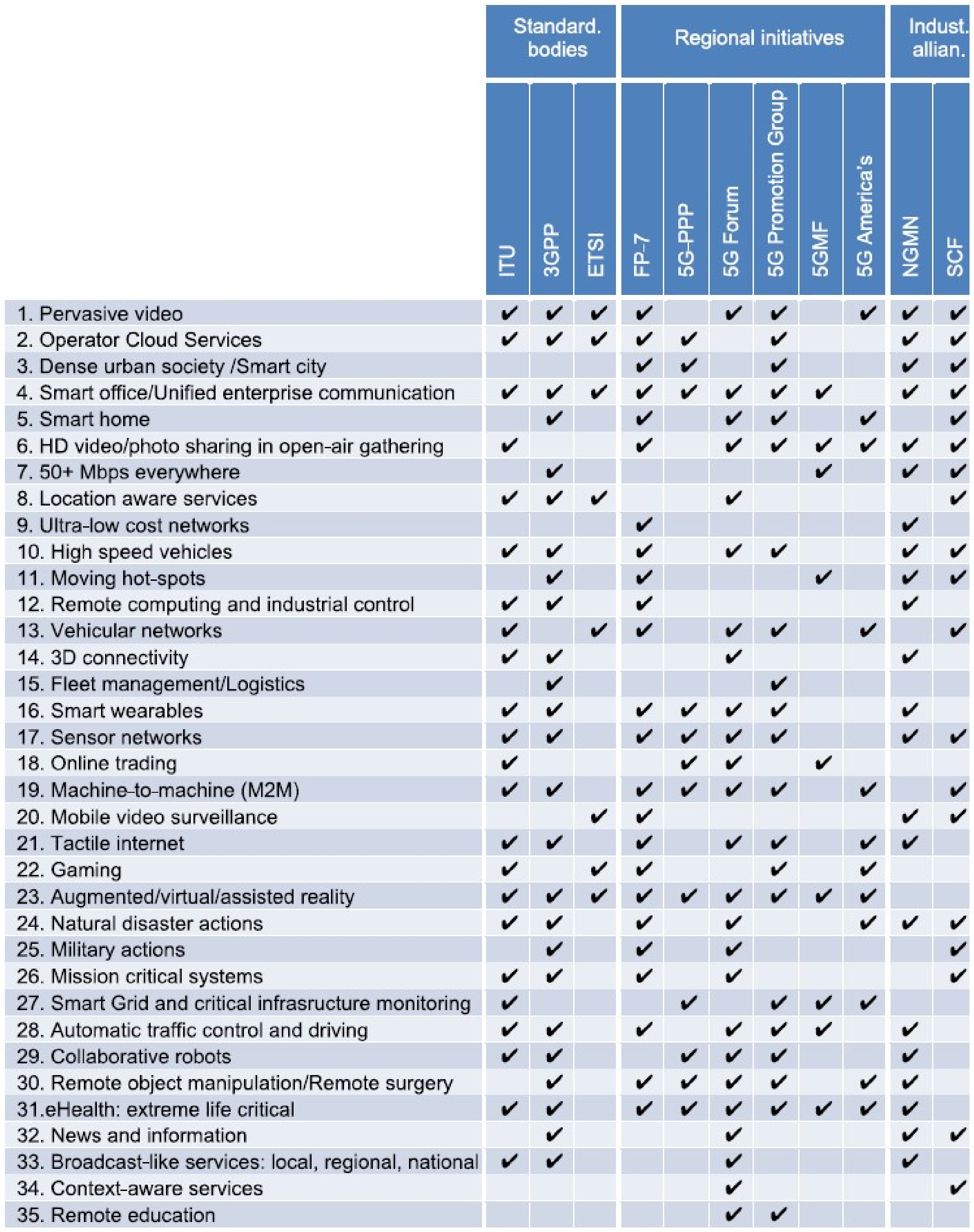

Families
Categories
Use Cases
Broadband access in dense area
Broadband access in dense area
Indoor ultra-high broadband access
Broadband access in a crowd
Pervasive video
Operator cloud services
Dense urban society
Smart Office
HD video/photo sharing in stadium/open-air gathering
Broadband access everywhere
50+ Mbps everywhere
Ultra low-cost broadband access for low ARPU areas
50 Mbps everywhere
Ultra-low cost networks
High user mobility
Mobile broadband in vehicles
Airplanes connectivity
High speed train
Moving Hot Spots
Remote computing
3D Connectivity: Aircrafts
Massive Internet of Things
Massive low-cost/long-range/low-power MTC
Broadband MTC
Smart wearables (clothes)
Sensor networks
Mobile video surveilance
Extreme real time communication
Ultra low latency
Tactile internet
Lifeline communication
Resilience and traffic surge
Natural disaster
Ultra-reliable communication
Ultra-high reliability & Ultra low latency
Ultra-high availability and reliability
Automatic traffic control/driving
Collaborative robots
Remote object manipulation – Remote surgery
eHealth: Extreme Life Critical
Public safety
3D Connectivity: Drones
Broadcast like services
Broadcast like services
News and information
Broadcast like services: Local, Regional, National

Figure 2. 1: Shows 8 slice classes that are that can further be extended to requirement and services [2].

The standardization bodies that include the development and defining of standards for fulfilling the aims are as follows ITU, ETSI, FP-7, 5G-PPP, 5G Forum, 5G Promotion Group, 5GMF, 5G America’s, NGMN and SCF. Figure 2.2 shows applications that lie under the umbrella of each standard body. Standardization Bodies are not just focusing on providing architectures of platforms that will serve all the application but also aims at providing easy management and ultimately wants to include features like plug and play, self-configuration with optimization and healing capabilities. Also, the technical requirements documents of 3GPP already focused on explaining the requirement of next-generation networks. One of the main requirement is

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

orchestration and management where it requires that the platforms should be able to provide the users with full control and can automatically manage the user requirements on the go [7].



	Standard. bodies			Regional initiatives						Indust. allian.	
	ITU	3GPP	ETSI	FP-7	5G-PPP	5G Forum	5G Promotion Group	5GMF	5G America's	NGMN	SCF
1. Pervasive video	✔	✔	✔	✔		✔	✔		✔	✔	✔
2. Operator Cloud Services	✔	✔	✔	✔	✔		✔			✔	✔
3. Dense urban society /Smart city				✔	✔		✔			✔	✔
4. Smart office/Unified enterprise communication	✔	✔	✔	✔	✔	✔	✔	✔		✔	✔
5. Smart home		✔		✔		✔	✔		✔		✔
6. HD video/photo sharing in open-air gathering	✔			✔		✔	✔	✔	✔	✔	✔
7. 50+ Mbps everywhere		✔						✔		✔	✔
8. Location aware services	✔	✔	✔			✔					✔
9. Ultra-low cost networks				✔						✔	
10. High speed vehicles	✔	✔		✔		✔	✔			✔	✔
11. Moving hot-spots		✔		✔				✔		✔	✔
12. Remote computing and industrial control	✔	✔		✔						✔	
13. Vehicular networks	✔		✔	✔		✔	✔		✔		✔
14. 3D connectivity	✔	✔				✔				✔	
15. Fleet management/Logistics		✔					✔				
16. Smart wearables	✔	✔		✔	✔	✔	✔			✔	
17. Sensor networks	✔	✔		✔	✔	✔	✔			✔	✔
18. Online trading	✔				✔	✔		✔			
19. Machine-to-machine (M2M)	✔	✔		✔	✔	✔	✔		✔		✔
20. Mobile video surveillance			✔	✔						✔	✔
21. Tactile internet	✔	✔		✔		✔	✔		✔	✔	
22. Gaming	✔		✔	✔			✔		✔		
23. Augmented/virtual/assisted reality	✔	✔	✔	✔	✔	✔	✔	✔	✔		
24. Natural disaster actions	✔	✔		✔		✔			✔	✔	✔
25. Military actions		✔		✔		✔					✔
26. Mission critical systems	✔	✔		✔		✔					✔
27. Smart Grid and critical infrasructure monitoring	✔				✔		✔	✔	✔		
28. Automatic traffic control and driving	✔	✔		✔		✔	✔	✔		✔	
29. Collaborative robots	✔	✔			✔	✔	✔			✔	
30. Remote object manipulation/Remote surgery		✔		✔	✔	✔	✔		✔	✔	
31.eHealth: extreme life critical	✔	✔		✔	✔	✔	✔	✔	✔	✔	
32. News and information		✔				✔				✔	✔
33. Broadcast-like services: local, regional, national	✔	✔				✔				✔	
34. Context-aware services						✔					✔
35. Remote education						✔	✔				

Figure 2. 2: The Standardization Bodies and the application that will be served by them as referenced under [1].

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

ONF (Open Networking Foundation) is one other key player in designing platforms for the 5G, it introduced CORD as a novel platform for SDN and NFV based network application. It simplifies the network architecture deployment and enables flexibility and scalability through network slicing [3]-[5]. In a very short time CORD becomes a favorite platform for many network operators and companies including Argella, Qualcomm, China Unicom, Skt and many others around the world. The flexibility, scalability and its capabilities allow the operator with freedom of deploying and controlling network. In the past, many operators have shown many demonstrations of its capabilities in terms of slicing scaling on different platforms and one of the major can be referenced from Radisys CORD videos on YouTube channel. Radisys is one of the key partners and collaborators with CORD [34] [35].

ETSI-NFV is one other platform that enables programmability both in terms of network data plane function and network control. It enables a platform that introduces OSS/BSS (Operation Support System/ Business Support System) firstly BSS deploys business requirement using Orchestrator and secondly, OSS fetches the monitor data to operate the system through the orchestrator [8].

OAI (OpenAirInterface) is compliance with ETSI and it provides open source network components including OAI-SIM and OAI-EPC [9] [10]. We can simulate and emulate different Network Functions using OAI components.

2.2.Intent-based platforms for simplifying Configuration

NGpaaS (Next-Generation Platform as a Service) is a platform that focuses on providing a generic platform that can easily configure and manage multiple 5G platforms, e-g M-CORD and NFV-MANO can coexist under one platform. It allows the user to configure underlying systems no matter from which vendor they belong to, it does it by defining SLA (Service Level

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Agreements) from the top and converts them to policies for fulfilling them [18]. Another platform that should not be neglected is provided by Nokia Bell Lab. Firstly, it is a demo provided by Nokia bells lab in which the simplify the resources allocation for energy slices and monitored slices for updating resources [11] [12].

2.3. The ML Approaches for Dynamic Resource Management

If look closely M-CORD is a CO on a cloud data center, so it allows us to develop and deploy VNF as VM (virtual Machines) on the cloud. But, at the configuration time, we cannot decide how much traffic will be streamed through the network. So it is an esteem requirement to follow a smart approach to dynamically manage the VM. Recently many researchers have worked on predicting the load on cloud i.e. they followed different approaches to find out the future load on the cloud [13]. Machine learning can be applied to the network in different many ways and a comprehensive survey can be referenced under [14]. Furthermore, researchers detected the VM error detection a SOM (Self Organizing Model) has been proposed and they decide VM performance based on CPU, RAM, Storage, and Network and used Machine Learning to detect the anomaly [15]. Conclusively, a very novel approach has been followed using apache spark and used big data to classify the VM/ HOST machine state in Fault and Normal on the basis of CPU, RAM, and Storage [16]. Finally, a very extensive comparison between different ML-approaches applied to the datacenter and VM resources usage and updates have been studied in the thesis and compared many approaches and results and found regression fits the best on such problems [17]. ETSI zero touch network and service management (ZSM) is an initiative which aims to automate the network control so that network management does not require any human assistance [32].

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Chapter 3
The overall system, IBN-Application, and M-CORD platform

The two major parts of the work consist of an automatic configuration and policy management application and automatic control over network operation for dynamic management of resources using Machine Learning. The detailed overall architecture design and IBN application with CORD are described as follows:

3.1 Overall system in view point of SDN and NFV platforms:

The overall system is aimed at providing a platform for network operators to simply configure network and it automates the dynamic resource management by applying machine learning. The IBN application provides an abstraction for network operators to configure underlying SDN/NFV platforms as shown in figure 3.1. It also accepts monitoring information to update system resources as is capable of predicting future resource utilization depending upon current resource usage. The IBN-application from top requires high-level information (what) and convert it to system policies and configurations (how) and eliminates the requirement of proficient experts required for different platforms. The system is designed to provide a top-level layer which can provide detailed configuration for network orchestrators. Currently, one of the major issue for operators is to monitor the network and keep the network state stable as resource requirement for fulfilling user demand may increase at any timestamp. The operator follows the manual updates by increasing number of resources by changing the system configurations which is tedious and error-prone. Hence, we follow a smart approach in which we will train Machine Learning models which perceives the current state of the system and predicts the system state at future time stamps which enable the IBN-application to decide what should be changed to keep

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

the system at a stable state. Figure 3.1 shows IBN application integrated with NFV-MANO based SDN/NFV platforms where IBN application receives high-level configuration that is network administrator intents in the form of contracts and is converted to system level-policies and configuration. Upon receiving the configurations, the NFV orchestrator instantiates the network function and slices on the physical network. On the other hand, a Monitoring/ Feedback agent provides monitoring information to the ML model which predicts future resource utilization of network on the basis of the predicted state decision engine decides the required updates in the system. So the system is a complete loopback system where network operator defines high-level intents and SLA and system automatically updates its state using ML.

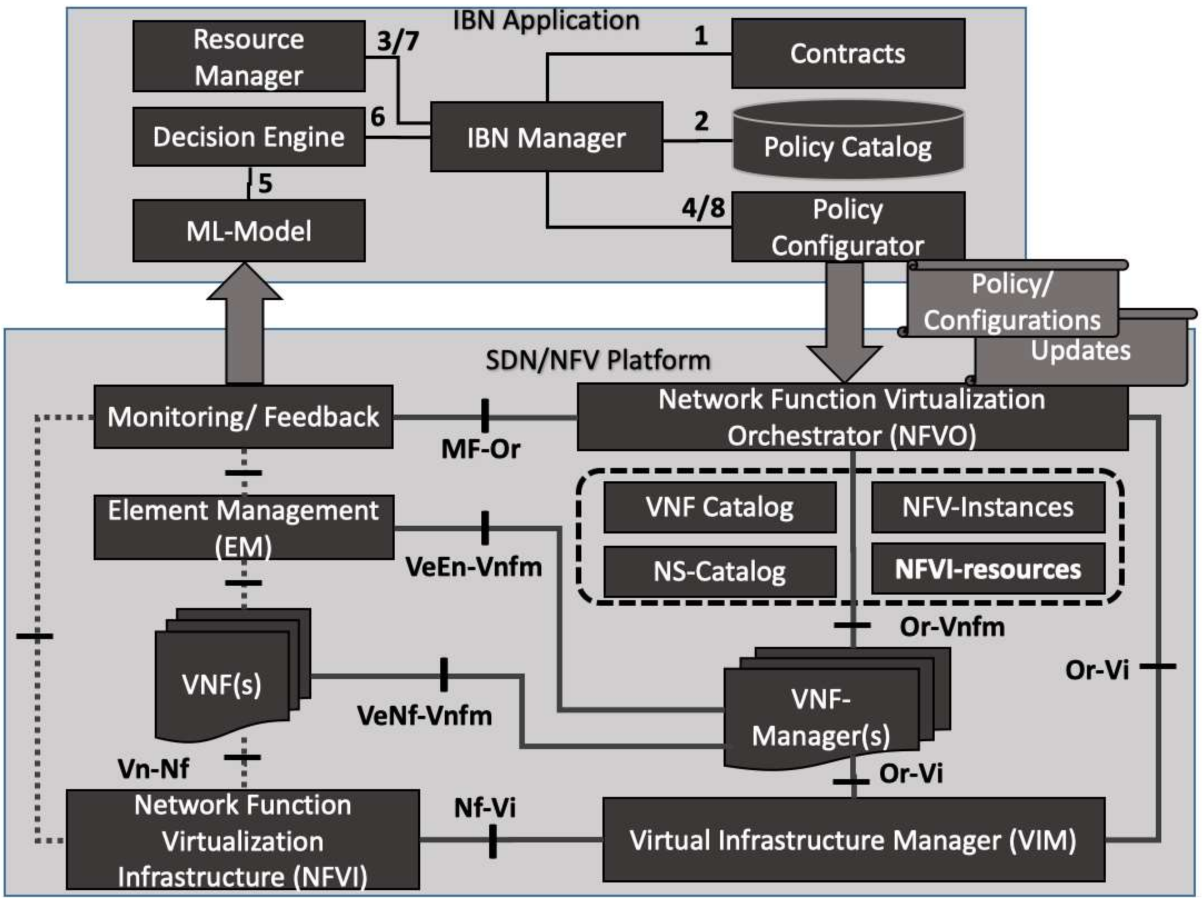


Figure 3. 1: Overall system in view point of NFV-MANO architecture.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

3.2IBN-Application

The need for simplification in the configuration of next-generation networks cannot be neglected. The purpose of IBN-Application is to simplify, generalize and automate the policy and configuration of 5G networks. The application is designed while considering the complexity of 5G and solutions available in the market [2][10][11][12][18]. The application is developed not only to automate and simplify the configuration and policy management but also to optimize the resource management. Furthermore, the application consists of five major parts Contract, IBN-Manager, Policy Catalog, Resource Manager and Policy Configurator. The overall working of the system is as follows. System Administrator or User inserts the SLA (Service Level Agreements) in the form of contracts from top and Intent-Manager as central part receive the SLA and its goal is to generate the system configuration in accordance with the agreement. For doing so it requires to consider many aspects of the networks. For example, slice type, network platform type, infrastructure constraints and Policy that should be applied. So, the Intent-Manager is assisted by the three other components that are Policy catalog for deciding the policy, network catalog, decision of dependencies and the VNF and architecture that will be required for configuring the SLA. The second is resource manager which determines the amount of resource required or can be allocated for each SLA while considering the stable state of the physical infrastructure. Thirdly the Policy Configurator Modules responsibility is to generate policies for the underground platform depending on the SLA or user request. IBN detailed working model of IBN application considering M-CORD as underground platform is depicted in figure 3.1. Furthermore, detailed design and working of each module is explained below.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

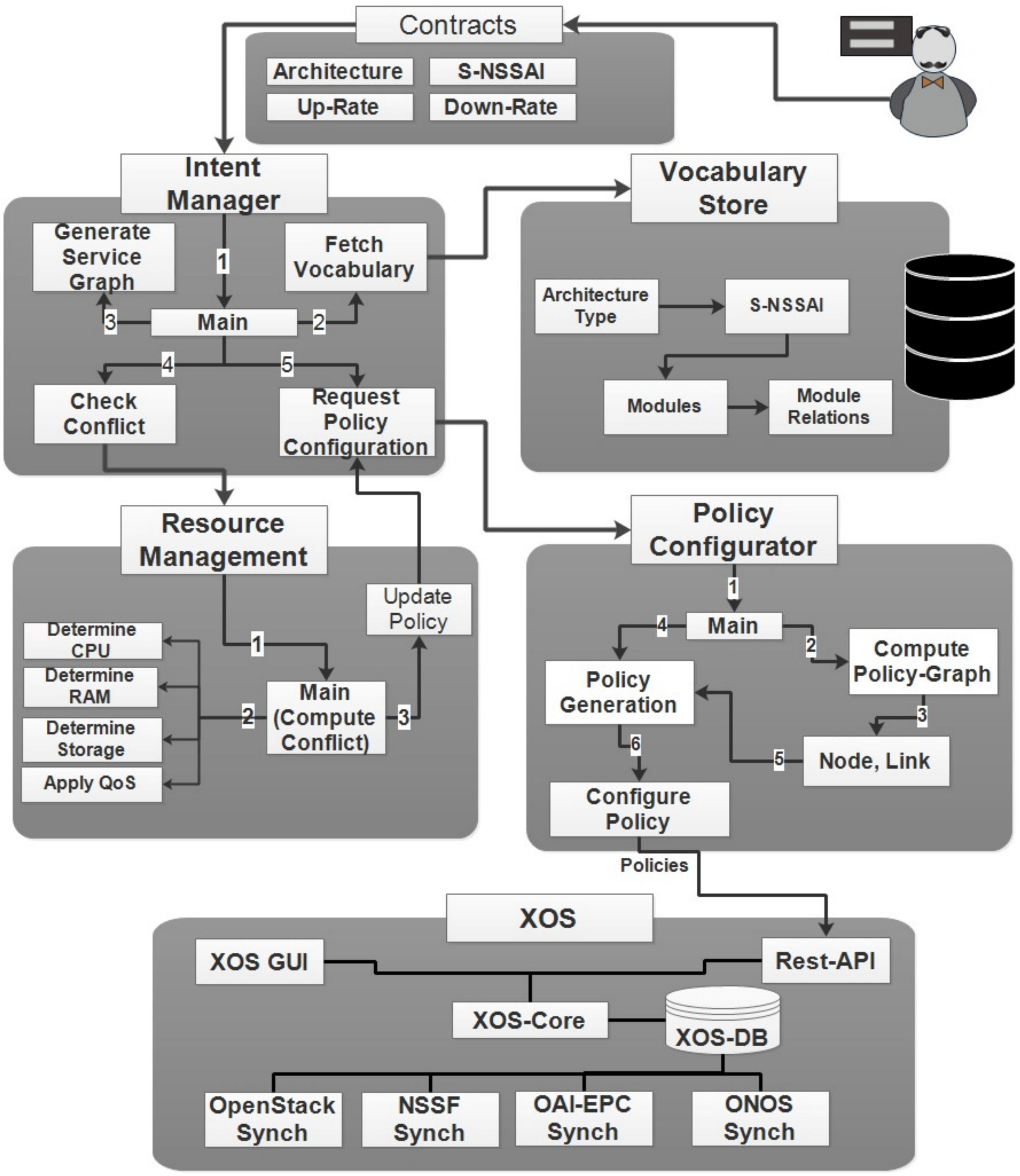


Figure 3. 2: The detailed architecture of IBN- Application.

I. Contracts

Contracts are the simple and high-level instruction that is required by the system to generate the configurations and policy for the underground system. The reason to introduce contracts for

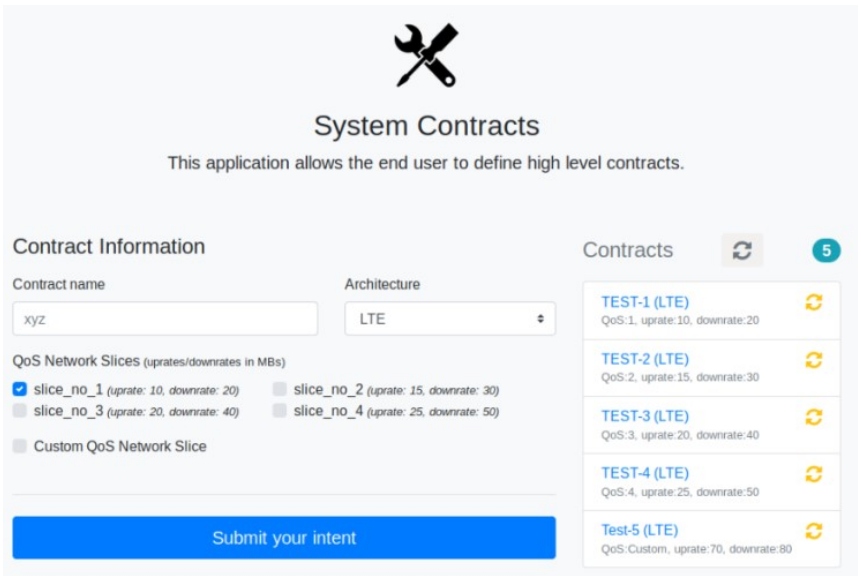
제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

the IBN-application is that they must be understandable by the entire technical person and they must overcome the requirement of multiple experts for network configurations. Hence, it used S-NSSAI as a major field and from the studies of different standardization bodies specifically 3GPP each S-NSSAI refers to a specified requirement that can be requested by a specified class of user equipment. Also, S-NSSAI is simple information that can explain what a user can require from the underlying system. Furthermore, each S-NSSAI can further be classified into subclasses as SST (Slice/Service Type) and SD (Slice Differentiator). In fact, SST classifies the slice and its service type, which allows us to select the Network function and capabilities accordingly. Likewise, SD further classifies the exact slice within a specific service type. Hence, it is clear that the use of S-NSSAI can allow the application to generate detailed network configurations which can be requested by user equipment [19].

Architecture-ID is the second most important input of the contract information. Since companies and organization have developed unique architectures and components that will be used as part of their network services. Also, for this test-bed, a new network architecture is designed and it consists of a mixture of OAI with NSSF. This Architecture ID simply classifies among the different architectures and components to be used as part of the configurations.

Since the beginning of networking and computing, the race was always about speed and bandwidth. Hence, the fourth most important input required by the system is QoS. In this, we enabled a user to allocate the Up-rate and Down-rates in accordance with its requirement. Furthermore, the application will tune the underlying system in accordance with the selected QoS.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY



System Contracts

This application allows the end user to define high level contracts.

Contract Information

Contract name: xyz

Architecture: LTE

QoS Network Slices (uprates/downrates in MBs)

- [✓ slice_no_1 *(uprate: 10, downrate: 20)*]
- slice_no_2 *(uprate: 15, downrate: 30)*
- slice_no_3 *(uprate: 20, downrate: 40)*
- slice_no_4 *(uprate: 25, downrate: 50)*
- Custom QoS Network Slice

Contracts 5

- TEST-1 (LTE) QoS:1, uprate:10, downrate:20
- TEST-2 (LTE) QoS:2, uprate:15, downrate:30
- TEST-3 (LTE) QoS:3, uprate:20, downrate:40
- TEST-4 (LTE) QoS:4, uprate:25, downrate:50
- Test-5 (LTE) QoS:Custom, uprate:70, downrate:80

Figure 3. 1. 1: Contracts GUI example.

II. IBN-Manager

IBN-manager implements the central logic of the IBN-Application and is responsible for collecting and formatting the data required for configuration. It follows the following steps:

- Firstly, it receives the contract from the user and using the contract information it sends a request to network catalog for fetching the details of required architecture and slice demands. The details are provided in the form of a service graph which determines the number of services, their order, and details about the dependency among them.
- Second, step is to request the resources for the architecture to be deployed and allocate proper QoS for each of the slices.
- Finally, on completion of requirement and determination of resources to be allocated it request Policy Configurator to convert the requirement into the configurations for the underlying platforms.

The following section explains the internal architecture and details of each of the above steps.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

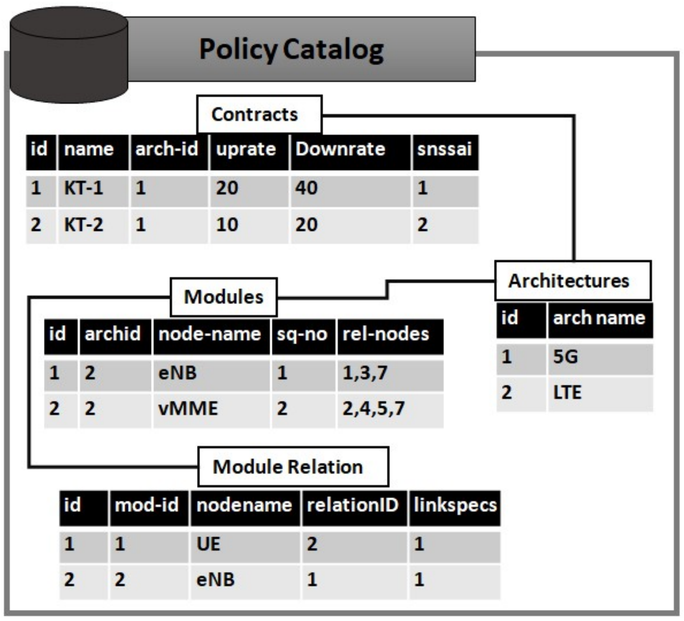
III. Network Catalogue

It is the central database it fulfills two requirements first it consists of all the details of architecture; second, the architecture details are stored in a form that they consider the dependency of each service on each other hence helps in policy formation for the underground system. In [11][12] and [21][22] they are showing some form policy blueprints, policy information which serves similar to the Network Catalogue. The uniqueness of our DB is that its architectural style is so well formed that it stores information while considering dependencies. The Database consists of four tables the details are as follows:

- Firstly, the Contract Table contains information about each of the requested contracts. That will be used by the database to fetch the architectural details to form a network catalog.
- The second table is the architecture table which contains the available architecture that can be configured. It is mapped through the contract.
- The third and most important table is Modules, it contains the information of the network functions to be allocated and it defines the location and relations of the function with other network functions.
- Finally, the module relation table is used to define the relationship between the VNF's that can be a bi/uni-directional. On the basis of the dependency, we define the Networks between the nodes also the network service graph is generated in accordance with the relation among the nodes.

It basically, provides information inform of service graph, and the graph contains the information of dependency and order. Figure 3. 1. 2 shows the internal architecture of the database.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY



Contracts

id	name	arch-id	uprate	Downrate	snssai
1	KT-1	1	20	40	1
2	KT-2	1	10	20	2

Architectures

id	arch name
1	5G
2	LTE

Modules

id	archid	node-name	sq-no	rel-nodes
1	2	eNB	1	1,3,7
2	2	vMME	2	2,4,5,7

Module Relation

id	mod-id	nodename	relationID	linkspecs
1	1	UE	2	1
2	2	eNB	1	1

Figure 3. 1. 2: Network Catalogue Database Design.

IV. Resource Manager

Resource management is the crucial requirement of the system; the purpose is to develop a smart approach to determine resources for the network modules. As we are allowing the user from top to configure the network to allocate slices and determine configurations for the underground system but each slice requires a certain amount of resources. So, we determined an approach through which we can define resources for each of the slices in accordance with the available resource amount. The details of the resource manager are as follows:

- IBN-Manager, request resources for the detailed number of instances required in accordance with the architecture details provided by Network Catalogue.
- From the list of the required number of slices, it calculates the number of VNFs i-e required no of VM, using the following formula.

$$\textbf{Required Number of Instances} = \mathbf{a} + \mathbf{b} \times \sum \textbf{Contracts}$$

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

- In network architecture e.g.vMME, vHSS and NSSF don't scale out with the increase in the number of slices and named as common functions. So **'a'** represents such function
- However, data-plane functions like SPGWU instances are specified for each slice and increase in the number of instances occurs. Also, a single slice can require more than one scalable instance, Hence, **'b'** represents the number of instances that should be instantiated per slice.
- A contract represents the requests by users and it is equal to the number of slices.

- Secondly, now for each instance, we need to decide how much of the CPU, RAM, and Storage should be allocated. Hence, we use the monitoring information which details the total available resources to determine the resources for each of the VNF. Depending Upon the total available resources we allocate the flavors for the VNF similar to standard flavor size defined by OpenStack.
  - Small: 1 vCPU and 2 GB RAM
  - Medium: 2 vCPU and 4 GB RAM
  - Large: 4 vCPU and 8GB RAM
- If, the number of slices cannot be allocated according to the above policy system decides 2 approaches
  - First is notify the user with a resource overload condition
  - Secondly, it just updates the configuration of the system to slice the sharing policy. The slice sharing policy basically forces two different slice requests to use the same resources and it is allocated using the slice information that is most similar to the requested slice.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Hence, by using the above strategy, resource management assures to determine conflict-free and stable resources for the underlying platform. Also, it tries to give a solution for resource shortage. Following, the algorithm represents the whole procedure.

Algorithm 1 Resource Manager

```
Input: Service Graph (Node, Links), S_NSSAI_list []
Output: S_NSSAI list, Image Size, number of Instances
1.  Function 1:Main (Service graph, S_NSSAI_list [])
2.  required_slices ← size of (S-NSSAI_list [])
3.  required_instances ← 4 + 2 ×required_slices
4.  While(change in Monitoring)
5.     vCPU ← Fetch(Cores) × 4
6.     RAM ← Fetch(RAM)
7.     Disk ← Fetch(Disk)
8.  InstanceFlavor ← Allocate(required_instances)
9.  Function 2: Allocate (required_instances)
10. if (required_instances>(vCPU÷4)&&(RAM÷8) &&(DISK÷80))
11. Imagesize ← Large
12. elseif (required_instances>(vCPU÷2)&& (RAM÷4) &&(DISK÷40))
13. Imagesize ← Medium
14. elseif (required_instances>(vCPU÷1)&& (RAM÷2) && (DISK÷20))
15. Imagesize ← Small
16. else
17. extraslices=(required_instances-((vCPU÷1) && (RAM÷2) && (DISK÷20)))÷2
18. itr ← 0
19.     X ← required_slices
20. while(extraslices>0)
21. S_NSSAI[itr] ← (S_NSSAI[itr]×10+S_NSSAI[X])
22. update_QoS (Max, Max, S_NSSAI[itr])
23. itr ← itr+1
24.         X ← X-1
25. templist [] size ← X
26. i ← 0
27.     while(i<X)
28. templist[itr1]=S_NSSAI_list[itr1]
29. i ← i+1
30. S_NSSAI_list[]=templist[]
31. Imagesize ← small
32. Return Imagesize
33. Return S_NSSAI_list[], InstanceFlavor
```

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

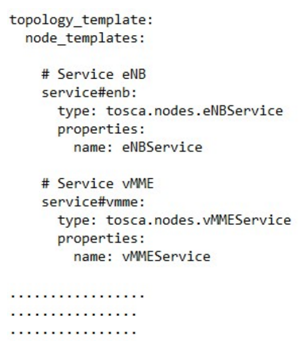
V. Policy Configurator

Policy Configurator converts the generated requirements and policies to the underlying platforms compatible pattern. In this work we developed it for configuring CORD platform so TOSCA is the most compatible option for orchestrating the configuration/policies on top of CORD [23]. The information received by the Policy Configurator can be listed as

a. Service Graph
b. Number of instances per VNF/ service/Node
c. Flavor of Instance
d. Dependency among the services
e. Order of VNFs
f. QoS in terms of up-rate and down-rate

In order to convert this information into CORD compatible configuration it generates three TOSCA files named as:

a. Service Graphs: it replicates the name of the nodes and considers them as a service in XOS and defines the dependencies among each of the services in accordance with the information listed above.



```
topology_template:
  node_templates:

    # Service eNB
    service#enb:
      type: tosca.nodes.eNBService
      properties:
        name: eNBService

    # Service vMME
    service#vmme:
      type: tosca.nodes.vMMEService
      properties:
        name: vMMEService

.................
................
................
```

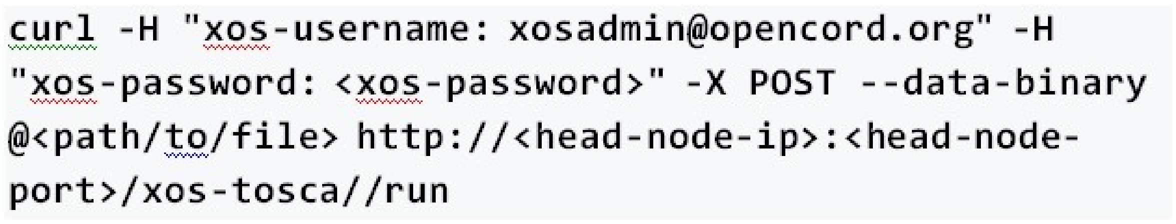
Figure 1: Service Graph Template example.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

b. Networks: it determines the private network for each service. It is responsible to associate the Ip address for each of the services and it uses an iterative approach for determining the networks e.g. for vmme_network IP is 10.0.1.0 and nssf_net will be assigned with 10.0.2.0.

c. Service Instances: it uses the number of slices and associates number instances per service according to the slicing policy provided by the above-listed information. Also, it associates QoS policies with each instance in accordance with user demand.

Once all the TOSCA files are created and information/Policies are converted the Policy Configurator using the TOSCA-RESTFUL API send it to CORD. It uses the following example command to accomplish this goal.



```
curl -H "xos-username: xosadmin@opencord.org" -H
"xos-password: <xos-password>" -X POST --data-binary
@<path/to/file> http://<head-node-ip>:<head-node-
port>/xos-tosca//run
```

After, pushing TOSCA-file to CORD controller the Policies are then processed by XOS to synchronize them to the physical layer next section will detail about CORD.

3. 3 M-CORD with it Components

M-CORD is one of the prestigious platforms for developing SDN and NFV based network architecture. It provides freedom of defining the network functions and SDN control applications. M-CORD provides the freedom from dedicated hardware and it shifted the CO to the Datacenter. M-CORD due to its novel design and applications is adopted by many telecom operators and industrial alliances have joined CORD community as partners including Radysis, Skt, China Unicom, Turk Telecom, Argella, Qualcomm and many others. The CORD consists of XOS everything as a service operating system, it allows the network functions and SDN control

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

application as services. As a comparison, it is a service operating system while normal operating systems are files based OS. XOS further uses ONOS as network controller and OpenStack as a cloud IaaS [24]. These three combined are the building blocks of CORD as shown in Figure 3. 2.1. XOS allows to create and deploy new services from the top, also ONOS is an open platform and one can develop its own network control applications for controlling the network [35]. A service can be anything from VNF to Function that can perform enhancements in the network. The network architecture, for example, whole 5G components including AMF (Access and Mobility Management Function), UPF (User Plane Function) and others can be developed as M-CORD services. Also, a specific control application can be developed as services, for example, a control application for an SMF (Session Management Function) functionality can be developed to control the UPF tunneling. Figure 3.2.1 shows the building block of the CORD including XOS, ONOS, and OpenStack. Each component is explained in the remaining section.

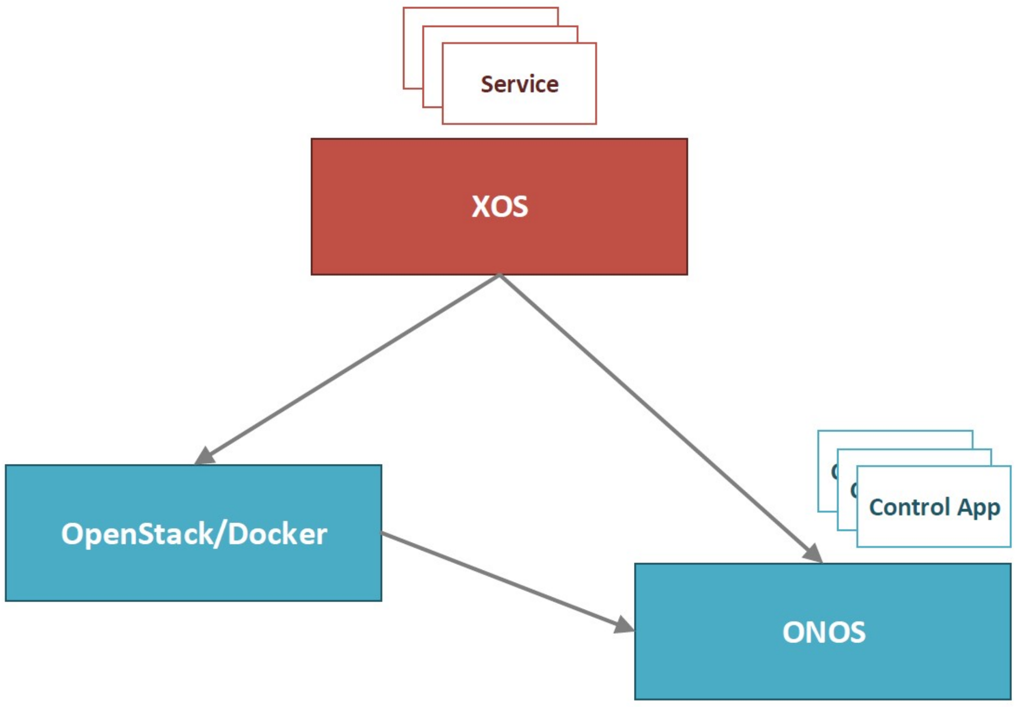


Figure 3. 2. 1: The Building Blocks of CORD

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

a. XOS (Everything as a Service Operating System)

XOS is the orchestrator used by CORD and is supported by OpenStack and ONOS in M-CORD architecture. Basically, it behaves similar to an operating system as it allows the assembling and composing of network functions and control application as services [4]. XOS can be divided into three layers as follows.

- Firstly, from the top, it accepts network configurations using two interfaces either GUI or TOSCA recipe. XOS GUI allows the user to generate configuration using an interactive interface. However, it allows third parties to configure it using the TOSCA recipe that can be pushed using REST-API.
- Secondly, XOS core constitutes of core models and each model represents a specific service. Also, models are created and instantiated using the configuration provided from the top. A service in XOS is the instantiated model using specified parameters. By an instantiated Model it means that Model instantiates the developed network Functions using the configuration provided from the top.
- Basically, a model is an abstraction of how service will behave on a real physical system. So the third and most important part of XOS is synchronizers that orchestrated these models on the physical system. Synchronizers basically use ONOS and OpenStack and other services to replicate the instruction provided from top to the physical system.

XOS consists of a set of Docker containers as illustrated in Figure 3. 2. 2. Each of the containers has a specific role in the above-mentioned steps. Each layer of XOS has specified containers so the first layer that represents the XOS northbound interface consists of the following:

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

- **xos-tosca:** It consists of a TOSCA engine, upon receiving TOSCA configuration it forwards the defined configuration in it to xos-core over gRPC.
- **xos-gui:** it is a gui interface for allowing the system administers to easily manage and change the service configuration. Each change is processed through REST using xos-chameleon.
- **xos-chameleon:** it is the REST interface translates messages received from GUI to gRPC and send s them to xos-core.

As we know the second layer constitute of data model following four containers manage the data models.

- **xos-db:** A PostgreSQL database used for the storage of model instances.
- **xos-core:** it contains the definition for each model and is used to translate the model instances that exist in xos-db into PythonDjango classes using Object Relational Mapping(ORM).
- **xos-redis:** it manages internal notifications using pub/sub channels. Watchers, a particular type of synchronizer use it to notify about the changes and system states.
- **xos-ws:** it propagates the notification generated by xos-redis to gui and notifies about the failures in synchronization.

The third and most important layer in CORD is synchronization so it is done using the synchronizers as follows.

- **synchronizers:** the purpose of the synchronizer is to replicate the models on to physical layer and any change in the model must also be replicated. So the synchronizers always accept model from xos-core and xos-db and convert them to the configuration for the specified service. The keep on watching both the changes in system and model using

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

watchers to keep the system state same as defined by models. They are the one responsible to update the underlying system for each entry made by an administrator [3] [4] [5].

XOS also contain other modules that are as follows:

- **xos-ui:** it is basically used on the deployment time and has no use on runtime. Also, is expected to be depreciated in the upcoming version of CORD.
- **registrator:** in CORD every service has its own synchronizer and synchronizers exist in CORD as a Docker container. So, on every new service discovery, it finds out the container for it and adds it to a service registry.
- **consul:** it is registry in CORD and is used for managing clusters of datacenter. But in our example, it will be sued to deploy different CORD deployment scenarios.

Figure 3. 2. 2 represents overall XOS Docker container and the blue boxes in the bottom shows the real system that will be deployed using CORD, VNFs box represent the actual virtual network functions that can be deployed using CORD.

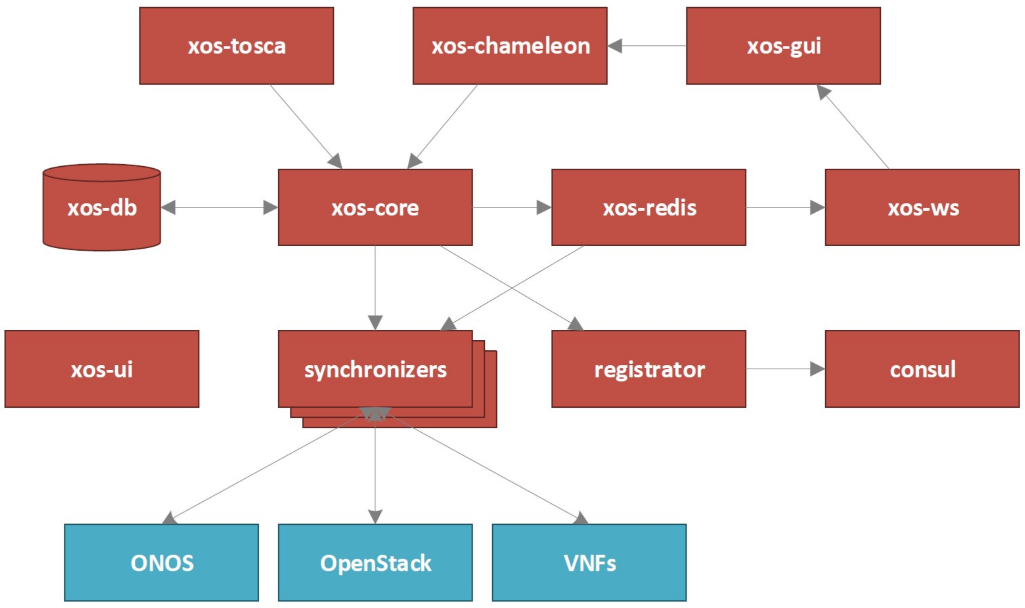


Figure 3. 2. 2: Shows interworking of Containers inside XOS and the real infrastructure that can be deployed using XOS.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

b. ONOS

ONOS is an open networking operating system it basically manages the physical and virtual connectivity and it is also responsible for hosting first-class CORD services [24]. ONOS in CORD has two separate instances which are as follows

- **onos-cord:** it consists of VTN (Virtual Tenant Networking Application) which provides us with the implementation of service composition. It helps the user to basically map instances across the services. It is basically responsible to apply the policy and dependency provided from the top onto the physical system. For example, it ensures that there is bidirectional communication required between two services or unidirectional. Either service requires connectivity to the internet or it requires the services from the management plane. So all the decisions and policy related to services and their tenants can be implemented using VTN. Implementation of working of VTN is depicted in Figure 3.2.3. It explains three services with multiple instances and it is shown that VTN provides the network between services and selects instance that will serve a certain slice. It shows that service dependency Policy is being provided from top e-g in the case of this development it will be provided using IBN-Application. VTN according to dependencies implements networking among the services using OpenStack Neutron. While Configuring Policy we also decide the network subnets and port for each instance of service and XOS orchestrate the physical VNF using OpenStack.
- Each Service has multiple instances, in CORD the multiple instances for a service can have two reasons one is load distribution and second is a tenant for specified SLA. So, from the top the service dependency can describe the tenants and CORD VTN decides the selection of Service instances based on the tenancy. Secondly, one tenant service can have multiple instances so

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

CORD through its built-in load balancing functionality selects among the same type of instances as shown in figure 3.2.3.

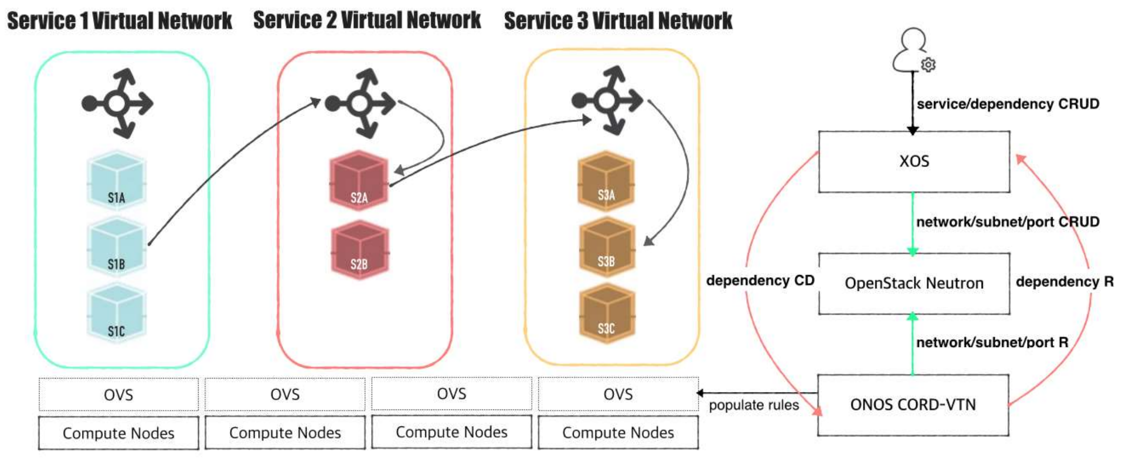


Figure 3. 2. 3: Shows how CORD VTN works.

- **onos-fabric:** it configures the fabric switches and also provides Internet-as-a-service in CORD through the vRouter application. Figure 3.2. 4 how CORD moved the CO office to Datacenter and how it reinvented datacenter to be used as CO. It consists of VNF deployed on the physical network and are being controlled by Trellis CORD ONOS application which consists of Fabric Control, vRouter, and VTN as building blocks.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

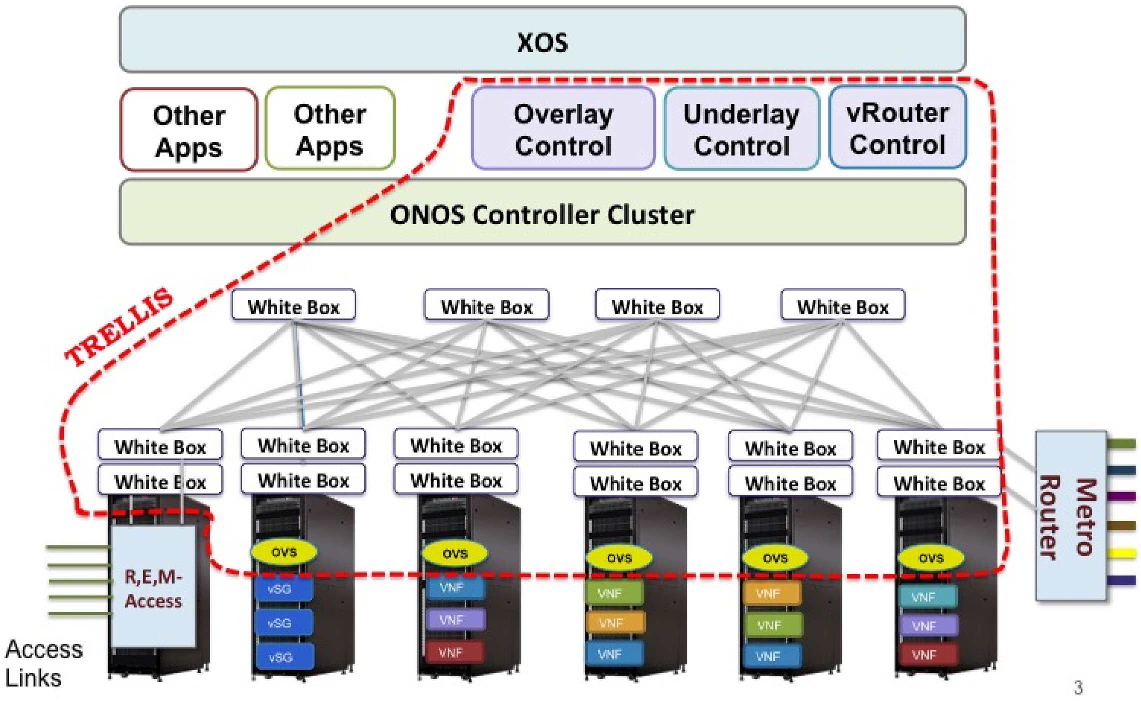


Figure 3. 2. 4: CORD Fabric with Trellis Control Application.

c. OpenStack

OpenStack in CORD serves as IaaS, it is open-source software that manages compute, network and storage in large scale data centers. It consists of several components each has a specified role and each component in CORD exists as LXD containers and can be seen in Figure 3. 2. 5. The communication of the components is also depicted in Figure 3. 2. 5. Few of them are considered important for explanation and are as follows.

- **Nova** (nova-cloud-controller): It provides and manages the compute resources and also it creates VM on physical machines. Basically, each VNF is developed as an OS image which implements the functionality of the VNF. So, Nova creates VM on Physical Machines for the required network functions

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

- **Neutron** (neutron-api): it is a network manager and provides VN's for connecting compute instances. However, it does all this in CORD on the instructions from ONOS controller as was explained above.
- **Keystone:** it is used for authenticating clients, discovering services and managing multi-tenant authorizations. As each tenant will be restricted to its own domain instances as was explained CORD VTN the service instance was selected based on tenant and load.
- **Glance:** it is for managing the discovery, registry, and retrieval of VM images.
- **Ceilometer:** it collects all the data that can be used for monitoring, customer billing, resource tracking and alarming capabilities from all OpenStack components. The data it can provide also includes VM resource utilization and it stores it in term of the time stamp, CPU utilization, RAM and Storage.
- **Nagios:** It is the built-in Monitoring agent for OpenStack that exists in CORD and it is used to fetch ceilometer info which may include CPU utilization, RAM and Memory logs.

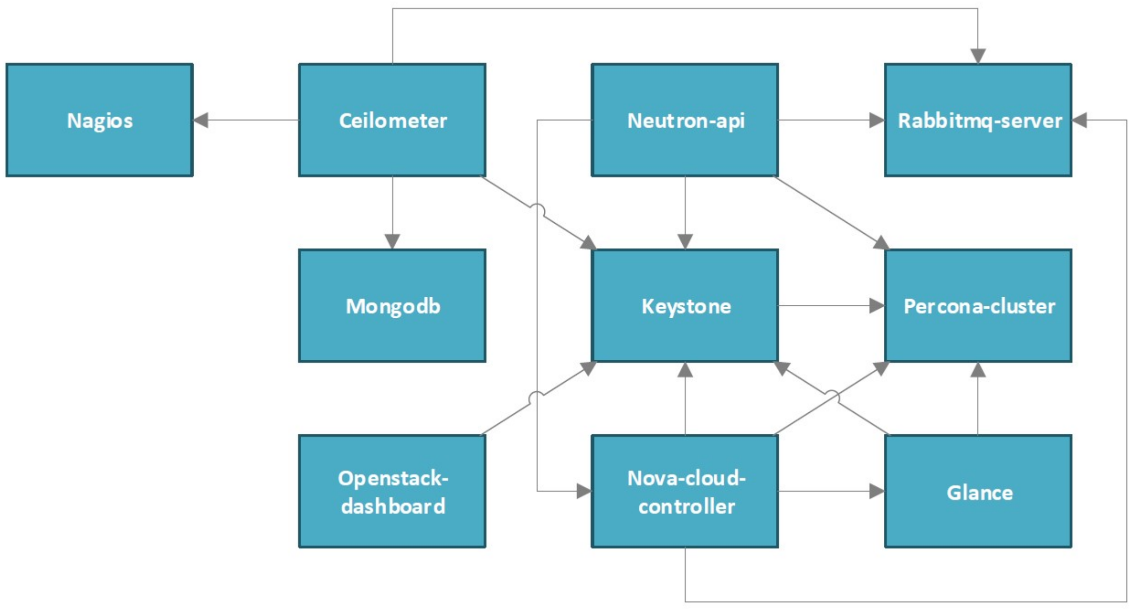


Figure 3. 2. 5: OpenStack internal architecture.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

3. 3 M-CORD based Network Slicing Scenario

As 5G standards were recently defined by 3GPP and the components and network function designed have not yet been developed and not available as open-source network functions. So for this research work, we used the open-source LTE virtual network function developed by OAI. OAI is an ETSI complaint organization which develops and provides network components. There are two types of the open-source network component that are used as part of our test-bed. Firstly, the access network that is provided in a package named as OAI-SIM which means OAI simulator. It basically simulates up to 20 user equipment and an eNodeB. Hence, it can simulate enough traffic for our core network. The second and most important open-source package it provides is OAI-EPC which is OAI- Evolved Packet Core. It consists of core network functions listed as vMME, vHSS, SPGWC, and SPGWU. The Source code and licensing information can be referenced from [25]-[28].The working of all the network functions is as follows:

- **OAI-SIM:** it can simulate the traffic from UE through eNodeB and sends a PDU session request to vMME. And after a successful session establishment, it sends traffic using the tunnel and SPGWU**.**
- **vMME:** upon receiving the connection request from eNodeB it forwards the request to vHHS for checking the subscription and it selects proper SPGWC for the UE on the confirmation.
- **vHSS:** it is a database server that contains all the subscription information and it allows a user to receive only those services which are subscribed for them.
- **SPGWC:** it receives the request from vMME and allocates a proper SPGWU and data-plane tunnel in accordance with the UE request.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

- **SPGWU:** it is the gateway which creates a tunnel between eNodeB and the SPGWU that connects it to the internet.

It is all provided by the OAI interface and is all up to 3GPPP standard document. Our test-bed not only considered these function we also included a unique network function that is NSSF (Network Slice Selection Function) [19]. This function was discussed in previous standards of 3GPP documents and also it is part of 3GPPP 5G architecture as shown in Figure 3. 3. 1.

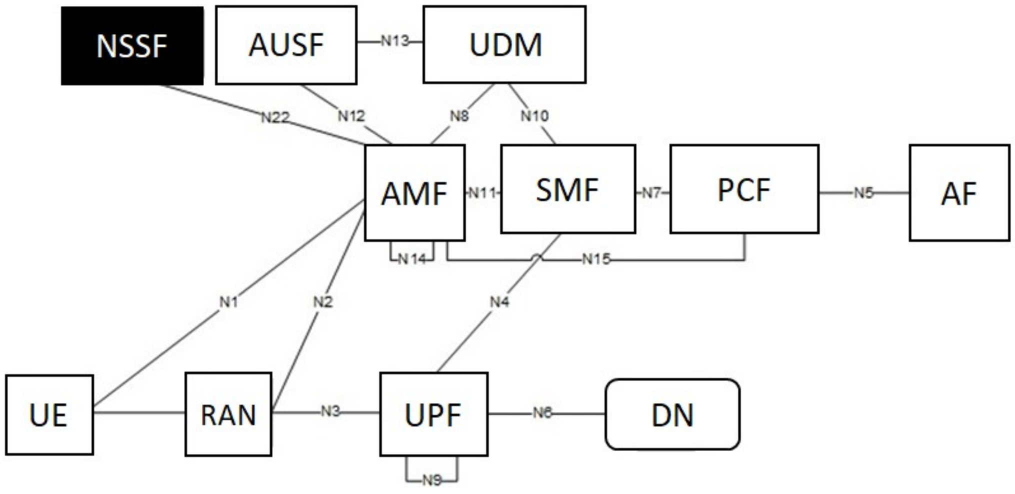


Figure 3. 3. 1: NSSF in 5G architecture [19]

This function basically selects the core network slices in accordance with the UE session request. For integrating this function in the test-bed we were required to make changes to the test-bed procedures and are as follows:

- The vMME internal procedure required some changes as NSSF sits on top of it. Before, NSSF inclusion vMME was selecting core network slices based on load balancing techniques. So, now it sends a request to NSSF for selecting a proper slice or core network instance, that will lead to the fulfillment of the desired service by user [30].

As OAI is platform independent development we have to use the functionality of each function inside M-CORD. As it is known that M-CORD allows everything as service so, the OAI

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

developed network was changed to M-CORD based service. We developed NSSF as a Service in M-CORD. The procedure followed to develop and integrate OAI into M-CORD based services can be referenced from the following [29].

The specialty of each service is that we created a Synchronizer for each of them and synchronizers accepts the Model from xos-db and replicate them on to physical layers. One of the special synchronizers that are very important to discuss is NSSF synchronizer [30].

The NSSF synchronizer fetches the core network model information from xos-db and stores them into its database, xos-db contains the information of the type of service and the host that is responsible to deliver the service. Hence, using this database NSSF is able to decide about the selection slice or network function in accordance with UE session request [30]. Also, synchronizer keeps a watch on updates in xos-db i.e.it keeps itself updated with any change in core network models as depicted in Figure 3. 3. 2.

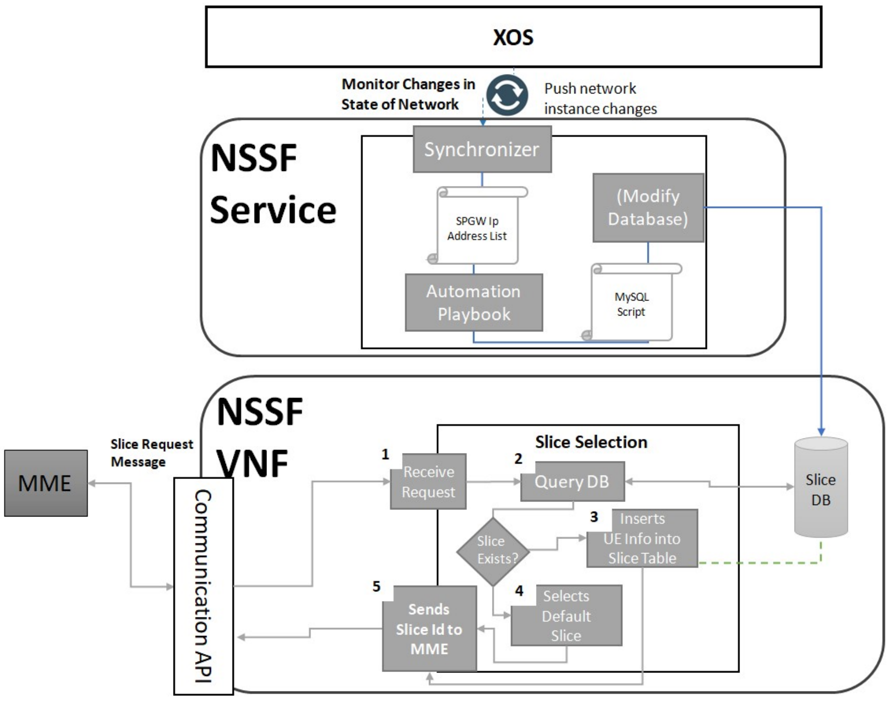


Figure 3. 3. 2: NSSF internal working

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Hence overall network slicing test-bed consists of OAI components with an enhancement for making it compatible with NSSF based slicing scenario. Also, all the components are integrated into M-CORD as services. Finally, M-CORD based network slicing scenario is depicted in Figure 3. 3. 3 where from top XOS contains the Models and Synchronizer including ONOS and OpenStack while real-time network function exists on the physical layer. Also, the VNFs are controlled using synchronizers, ONOS and OpenStack [30].

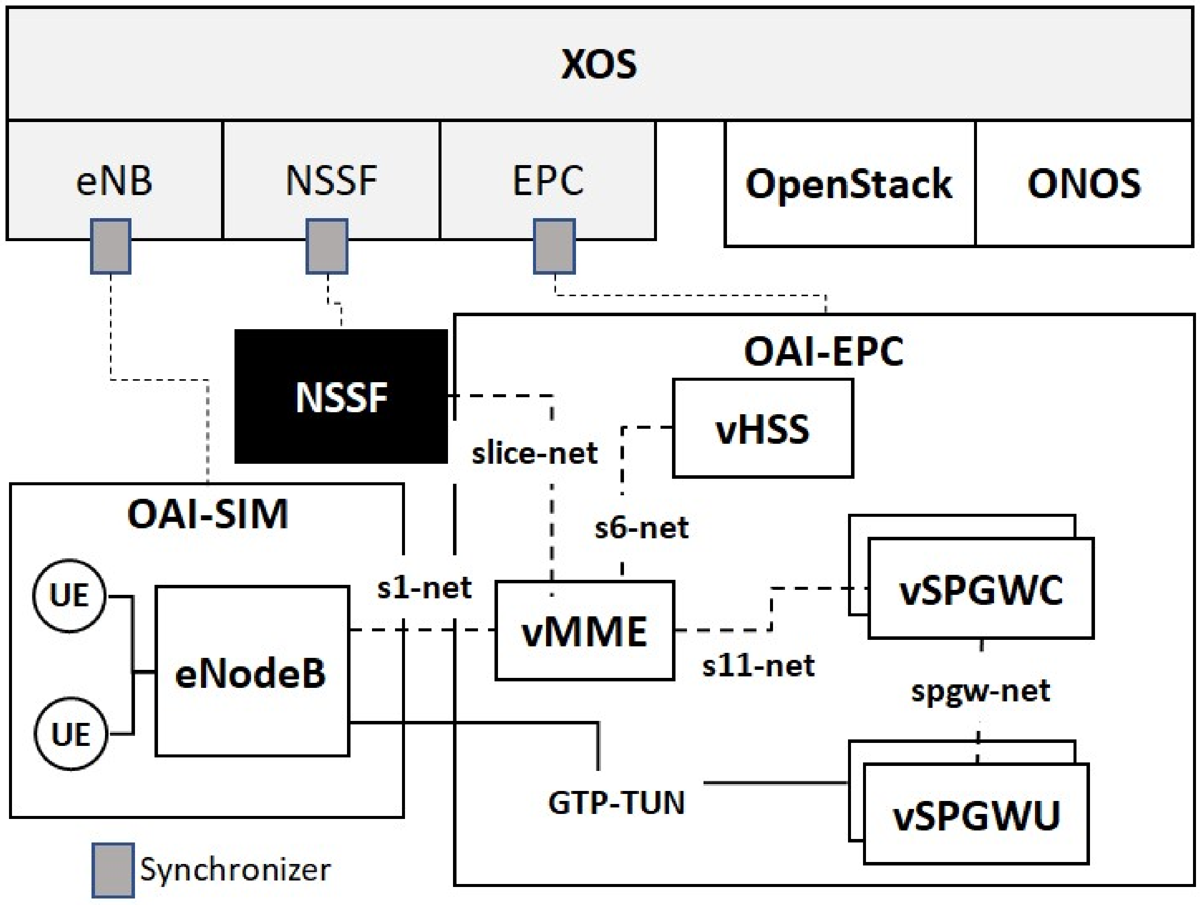


Figure 3. 3. 3: Overall Network Slicing test-bed including M-CORD NSSF and OAI

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Chapter 4
Machine Learning Approach and Experimental Results

Our goal in this research is not only to simplify the configuration but also to enable dynamic resource management for the M-CORD environment. So to achieve this goal we need to design a system that should have the following properties.

- Firstly, it should automatically detect the resource demand.
- Secondly, it should be able to predict the future failure state in terms of resource requirement.
- Third, it should dynamically update the system resources in accordance with the resource requirement.

To achieve the above-mentioned goals system considered a monitoring application that will monitor the system resources with time. Secondly, we integrated machine learning model inside the IBN-Application that upon receiving the system current resource state predicts the future resource status. In accordance with the future resource status, we designed a decision engine that decides whether the future state of the system leads to failure or not. According to the future state of the system, it makes the decision of scaling out of the system resources. Then it requests the IBN-Application to change the system configuration to increase the number of instance for that particular resource. Furthermore, this chapter includes the details about the monitoring app and the machine learning model training and the overall system is depicted in Figure 4. 0. 1.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

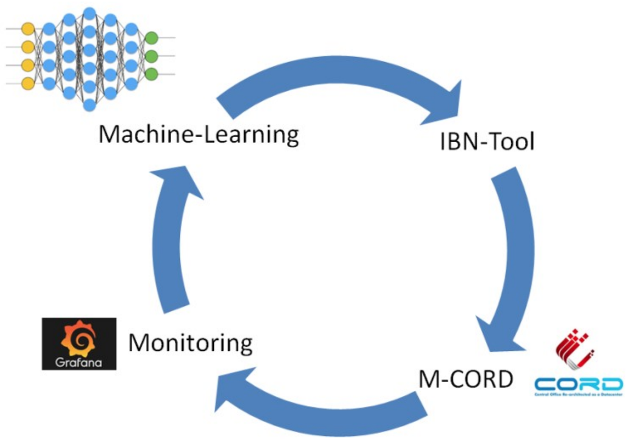


Figure 4. 0. 1: General Lifecycle dynamic resource management system

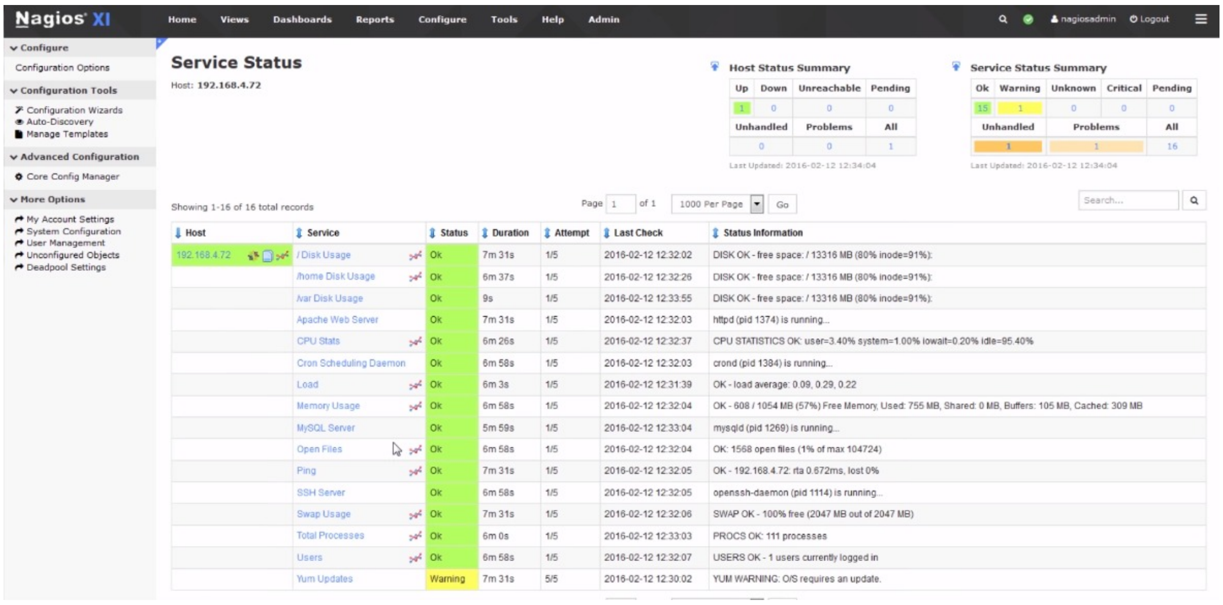
4. 1 Monitoring Application

OpenStack is the default IaaS provider for M-CORD, hence all the VM are under the control of OpenStack. Chapter 3, explained OpenStack components, Ceilometer is the one that logs the information about resources that can be used for monitoring purposes. There are many ways to fetch this monitoring information and perform the required operation on the information accordingly. For our work, we chooseNagios as a Monitoring tool, which is very flexible with CORD. It enables us to monitor each tenant and instance with the required monitoring information. Also, it is a very good tool to analyze the system state at any random time. But, as our goal is not only to check the current state of the system rather we need to be advanced to predict the future state of the system. So, that we can avoid any resource failure at runtime. Why is it important to predict the future state? It is because the orchestration and management of system scalability require time. If we will depend on the current state of the system. The orchestration time required will generate overhead in terms of latency and user will feel a little delay which does not fulfill the goals of 5G.

Nagios has a very interactive GUI and it also is enabled with rest-api we can receive and request using rest interfaces. So, we deployed Nagios inside the compute node and it monitors the hosts

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

and we can view analyze and collect the information. A performance matrix is depicted in figure 4.1.1. Where it is clear that it can monitor CPU, Load, Memory, and Storage. Also as it is designed as a generic monitoring tool it can also help us with Monitoring other information, including network bandwidth, and protocol monitoring. It can graph the utilization of the parameters at any time as shown in figure 4.1.2.



Service Status

Host: 192.168.4.72

Host Status Summary

Up	Down	Unreachable	Pending
1	0	0	0

Unhandled	Problems	All
0	0	1

Last Updated: 2016-02-12 12:34:04

Service Status Summary

Ok	Warning	Unknown	Critical	Pending
15	1	0	0	0

Unhandled	Problems	All
1	1	16

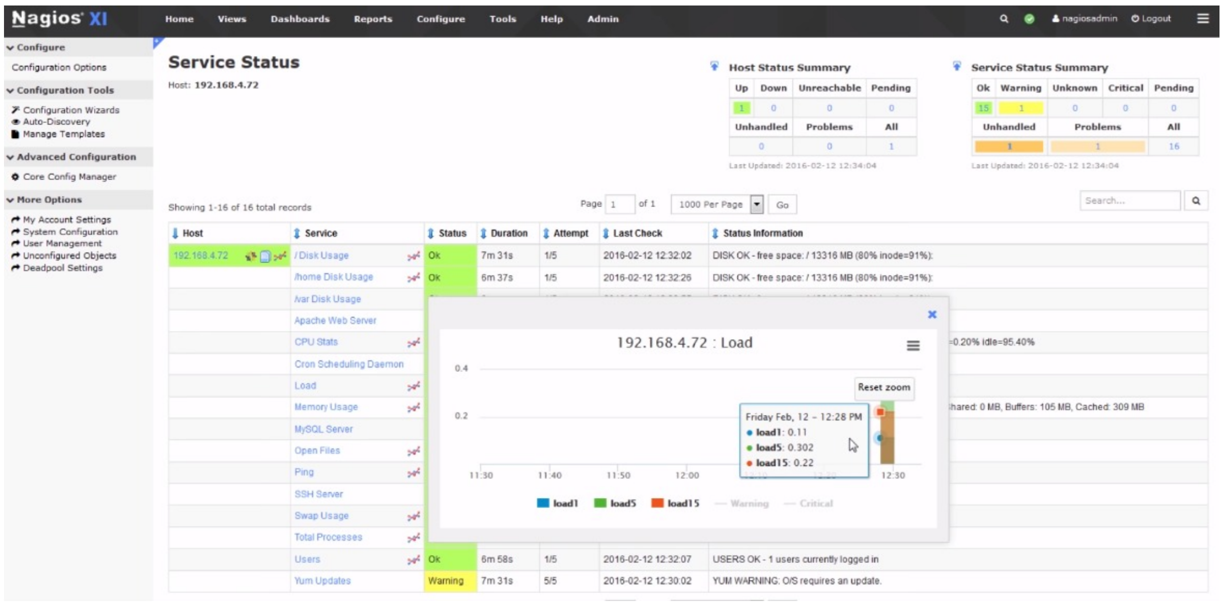
Last Updated: 2016-02-12 12:34:04

Showing 1-16 of 16 total records

Host	Service	Status	Duration	Attempt	Last Check	Status Information
192.168.4.72	/ Disk Usage	Ok	7m 31s	1/5	2016-02-12 12:32:02	DISK OK - free space: / 13316 MB (80% inode=91%):
	/home Disk Usage	Ok	6m 37s	1/5	2016-02-12 12:32:26	DISK OK - free space: / 13316 MB (80% inode=91%):
	/var Disk Usage	Ok	9s	1/5	2016-02-12 12:33:55	DISK OK - free space: / 13316 MB (80% inode=91%):
	Apache Web Server	Ok	7m 31s	1/5	2016-02-12 12:32:03	httpd (pid 1374) is running...
	CPU Stats	Ok	6m 26s	1/5	2016-02-12 12:32:37	CPU STATISTICS OK: user=3.40% system=1.00% iowait=0.20% idle=95.40%
	Cron Scheduling Daemon	Ok	6m 58s	1/5	2016-02-12 12:32:03	crond (pid 1384) is running...
	Load	Ok	6m 3s	1/5	2016-02-12 12:31:39	OK - load average: 0.09, 0.29, 0.22
	Memory Usage	Ok	6m 58s	1/5	2016-02-12 12:32:04	OK - 608 / 1054 MB (57%) Free Memory, Used: 755 MB, Shared: 0 MB, Buffers: 105 MB, Cached: 309 MB
	MySQL Server	Ok	5m 59s	1/5	2016-02-12 12:33:04	mysqld (pid 1269) is running...
	Open Files	Ok	6m 58s	1/5	2016-02-12 12:32:04	OK: 1568 open files (1% of max 104724)
	Ping	Ok	7m 31s	1/5	2016-02-12 12:32:05	OK - 192.168.4.72: rta 0.672ms, lost 0%
	SSH Server	Ok	6m 58s	1/5	2016-02-12 12:32:05	openssh-daemon (pid 1114) is running...
	Swap Usage	Ok	7m 31s	1/5	2016-02-12 12:32:06	SWAP OK - 100% free (2047 MB out of 2047 MB)
	Total Processes	Ok	6m 0s	1/5	2016-02-12 12:33:03	PROCS OK: 111 processes
	Users	Ok	6m 58s	1/5	2016-02-12 12:32:07	USERS OK - 1 users currently logged in
	Yum Updates	Warning	7m 31s	5/5	2016-02-12 12:30:02	YUM WARNING: O/S requires an update.

Figure 4. 1. 1: Front end of Nagios with different number of Monitoring Parameters

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY



Service Status

Host: **192.168.4.72**

Host Status Summary

Up	Down	Unreachable	Pending
1	0	0	0
Unhandled	Problems	All	
0	0	1	

Last Updated: 2016-02-12 12:34:04

Service Status Summary

Ok	Warning	Unknown	Critical	Pending
15	1	0	0	0
Unhandled	Problems	All		
1	1	16		

Last Updated: 2016-02-12 12:34:04

Showing 1-16 of 16 total records

Host	Service	Status	Duration	Attempt	Last Check	Status Information
192.168.4.72	/ Disk Usage	Ok	7m 31s	1/5	2016-02-12 12:32:02	DISK OK - free space: / 13316 MB (80% inode=91%):
	/home Disk Usage	Ok	6m 37s	1/5	2016-02-12 12:32:26	DISK OK - free space: / 13316 MB (80% inode=91%):
	/var Disk Usage					
	Apache Web Server					
	CPU Stats					=0.20% idle=95.40%
	Cron Scheduling Daemon					
	Load					
	Memory Usage					hared: 0 MB, Buffers: 105 MB, Cached: 309 MB
	MySQL Server					
	Open Files					
	Ping					
	SSH Server					
	Swap Usage					
	Total Processes					
	Users	Ok	6m 58s	1/5	2016-02-12 12:32:07	USERS OK - 1 users currently logged in
	Yum Updates	Warning	7m 31s	5/5	2016-02-12 12:30:02	YUM WARNING: O/S requires an update.

192.168.4.72 : Load

Friday Feb, 12 - 12:28 PM
● **load1**: 0.11
● **load5**: 0.302
● **load15**: 0.22

Figure 4. 1. 2: Showing graph value for a Load parameter at time 12:30 from the front end.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

4. 2 The Machine Learning approach

This section starts with data gathering, then it discusses the data preprocessing and further explains the Machine Learning approaches followed.

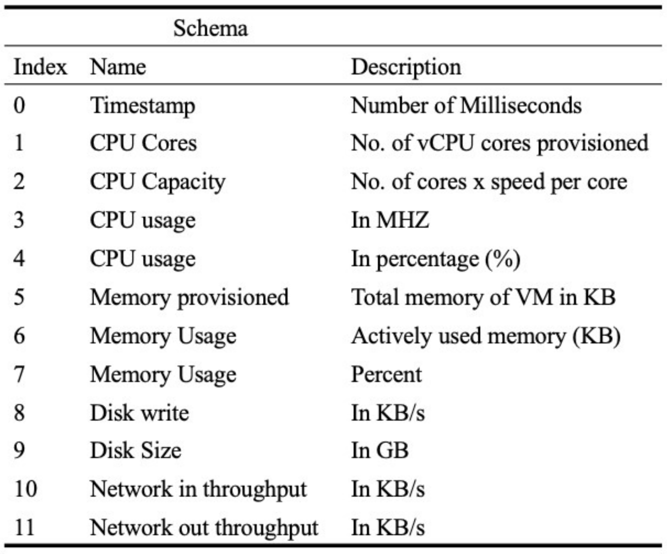
- **Data Collection:**

The open-source GWA-T-13 MATERNA data is used for this model training. The set is collected from three distributed Traces and each contains 520, 527 and 547 VMs data. Each Trace spans about a 1 month of monitoring data.Each, VM performance evaluation data is stored as .csv file.

- TRACE 1 on average contains 8340 rows for each VM record.
- TRACE 2 on average contains 8600 rows for each VM record.
- TRACE 3 on average contains 10000 rows for each VM record.

Each row contains 12 monitored parameters as elaborated in Table I. Special thanks to Materna a service provider for many German public sector users and banking applications for the open-sourced resource. The details about how data-collection and distributed datacenters can be referenced from [31].The first goal is to train a model that can efficiently predict CPU-utilization using the dataset. Hence, the target variable is CPU-Utilization for training the model.

Table I: Schema of MATERNA GWA-T-13



Schema		
Index	Name	Description
0	Timestamp	Number of Milliseconds
1	CPU Cores	No. of vCPU cores provisioned
2	CPU Capacity	No. of cores x speed per core
3	CPU usage	In MHZ
4	CPU usage	In percentage (%)
5	Memory provisioned	Total memory of VM in KB
6	Memory Usage	Actively used memory (KB)
7	Memory Usage	Percent
8	Disk write	In KB/s
9	Disk Size	In GB
10	Network in throughput	In KB/s
11	Network out throughput	In KB/s

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

- **Data Visualization:**

The view of the average CPU utilization compared with memory is shown in Figure 4. 2. 1. It can be perceived that most of the time pattern of usage of the avg CPU and memory stays the same except few timestamps. They follow the same up and down but the amount of the usage up for CPU and memory cannot stay the same. From this, we assumed that the memory can be used for prediction of CPU-Utilization.

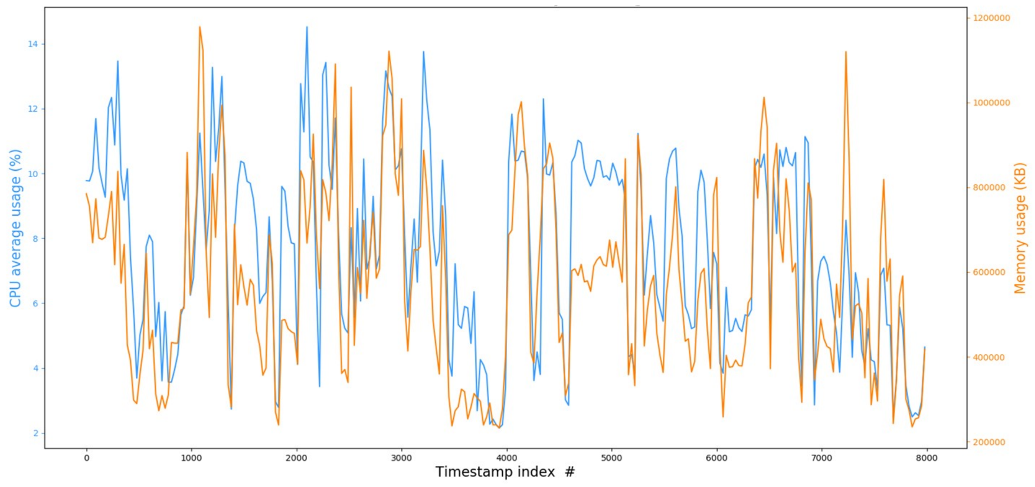


Figure 4. 2. 1: Comparison between the average CPU-utilization and memory usage with the time.

Similarly, other parameters showed different patterns with time as can be seen in Figure 4. 2. 2. However, for this work, we considered disk and CPU utilization as a critical part. When the streaming packet increases with a very high amount they form a queue in the VM which directly impacts memory and CPU utilization. However, we can see that network and disk utilization shows different behavior with the time sometime they match but have a lot of difference also compared to memory and CPU utilization they follow a different pattern.

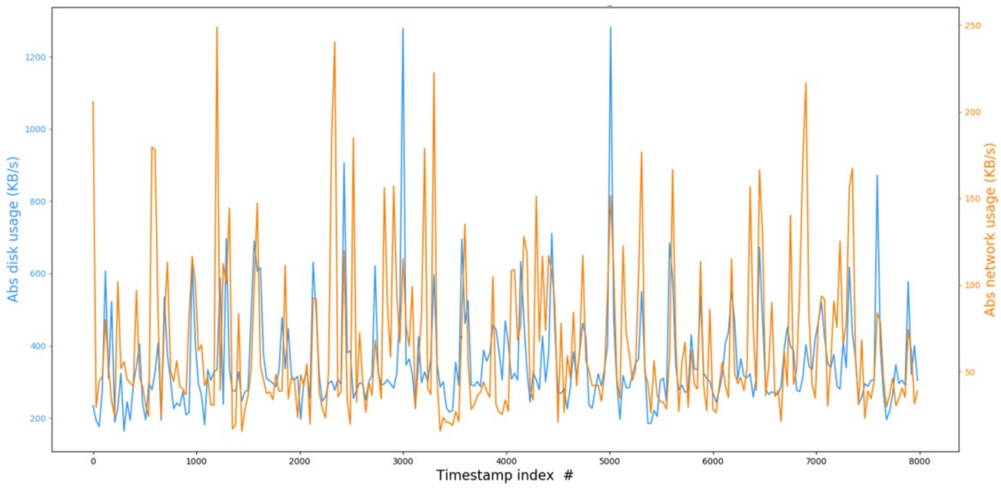


Figure 4. 2. 2: Comparison of disk and memory utilization with time stamp

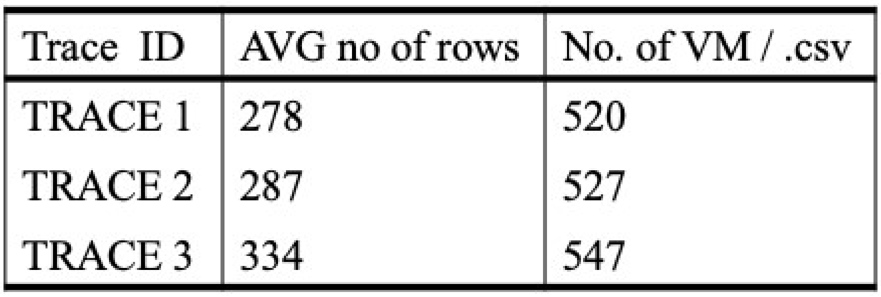
제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

- **Data Preprocessing:**

After investigating the dataset, the data-set rows look redundant for multiple numbers of rows. So the problem is that data is collected for every millisecond and the changes that occur takes a little time and also due to this fact we have a very large dataset. As a result, the averaging filter is used on the raw data. The averaging window size was set to 9 seconds which equals 30 rows of the data. This will calculate the average resource utilization on all the 520, 527 and 547 VMs of the three TRACES respectively.

After applying the averaging filter, the dataset is changed to the avg-filter-data, structure of the avg-data is illustrated in Table II.

Table II: Average Data-Set Structure



Trace ID	AVG no of rows	No. of VM / .csv
TRACE 1	278	520
TRACE 2	287	527
TRACE 3	334	547

After applying to average the next step is to generate input and output for the Model. To generate inputs all of the parameters were cascaded and aggregated in groups using-dataset. We kept one last row for CPU utilization which will be used for CPU utilization classification. The total number of rows considered for aggregation and cascading are 10 and the 11th will be used for CPU utilization. Hence the dimension of input data will be 60 and is shown in Table III.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Table III: Input Data Schema

	Schema
Index	Name
0	average-CPU-usage-0 (%)
1	average-memory-usage-0 (%)
2	average-disk-read-0 (KB/s)
3	average-disk-write-0 (KB/s)
4	average-network-in-0 (KB/s)
5	average-network-out-0 (KB/s)
6	average-CPU-usage-1 (%)
7	average-memory-usage-1 (%)
8	average-disk-read-1 (KB/s)
9	average-disk-write-1 (KB/s)
10	average-network-in-1 (KB/s)
11	average-network-out-1 (KB/s)
	….
54	average-CPU-usage-9 (%)
55	average- memory-usage-9 (%)
56	average-disk-read-9 (KB/s)
57	average-disk-write-9 (KB/s)
58	average-network-in-9 (KB/s)
59	average-network-out-9 (KB/s)

The output of the dataset is formulated, using aggregation and cascading on average CPU-utilization. It resulted in the classification of CPU-utilization into upper and lower bounds. The classification method used is Uniform classification method where the range between each group is 5 and are shown in Table IV. Not only this, but we also tried the exponential classification method but the results were not good also the exponential classification is not a good scheme for our proposed model.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Table IV: Classification of CPU-utilization

Class #	Percentage Range
0	0 – 5%
1	5 – 10%
2	10 – 15%
3	15 – 20 %
4	20 – 25 %
5	25 – 30 %
6	30 – 35 %
7	35 – 40 %
8	40 – 45 %
9	45 – 50 %
10	50 – 55 %
11	55 – 60 %
12	60 – 65 %
13	65 – 70 %
14	70 – 75 %
15	75 – 80 %
16	80 – 85 %
17	85 – 90 %
18	90 – 95 %
19	95 – 100 %

The input and output data capture all the required data to map the CPU, memory, disk and network utilization to predict any CPU utilization in the future.

- **ML Model Training:**

A very detailed explanation of different approaches and results achieved by a different research group in the prediction of CPU- utilization are referenced under [14] [17]. The dimension of input data is very high hence only neural networks are the most suitable approach in our scenario. For this case, we trained our model using ARNN (Artificial Recurrent Neural Network) and NN MLP (Neural Network Multilayer Perceptron) classifier. Both, the models were trained for predicting the CPU utilization. As in the data visualization section we have seen that there were

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

similarities and exceptions in the parameters and their dependence upon each other so we used different parameter selection approaches for training our models.

- The first configuration contains only CPU utilization; where each time stamp has only CPU utilization.
- Second, configuration contains CPU and memory; where each timestamp considers CPU and memory utilization.
- Third, the configuration contains all the parameter of the dataset across the given timestamp.

- **Evaluation and Results for each of the Model:**

Table V: Results in terms of train and test accuracy

ML-Algorithm	Input data	Train Accuracy	Test Accuracy
NN-MLP	cpu	0.955	0.951
	cpu-mem	0.921	0.916
	all	0.899	0.890
NN-ARNN	cpu	0.883	0.881
	cpu-mem	0.854	0.851
	all	0.846	0.834

Table V showed an inverse relationship between the number of parameters and accuracy. It shows that the increase in parameters adds more noise to the CPU utilization prediction. Even, memory utilization does not add to the results and behaved as noise for the prediction. Second, the thing that we recognized is that MLP-Classifier performed much better than ARNN in this case. Therefore, we decided to use MLP-NN as our default Model. The second step is to train a Model for memory prediction and disk utilization prediction. We followed a similar approach in the data preprocessing and performed cascading and aggregation on an averaging dataset which divided both the disk utilization and memory utilization into uniform classes similar to CPU

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

utilization. After that Models were trained for both the disk and memory utilization as the output parameters. Evaluation of the Models is shown in the following Table VI.

Table VI: Results of Model training for predicting Memory and Storage

		NN-MLP	
Output	Input data	Train Accuracy	Test Accuracy
memory	mem	0.924	0.911
disk	disk	0.899	0.884

As it was evident from data visualization that disk utilization patterns are different and also for an unknown reason the results of disk prediction are not as good as compared with memory and CPU-utilization. Furthermore, we can include other parameters and train models for increasing the decision boundary. Network utilization is one of the important parameters to be considered so for our final and single model configuration will be like. Timestamp, CPU, Memory, Storage, and Network as an input parameter and it will predict the future time utilization for each of the parameters. The Final Model architecture can be perceived in figure 4. 2. 3 which consists of 5 inputs and 3 hidden layers and 4 output layers. The next step is to decide that what predicted values can result in a Failure state or require a change in the configuration of the system. The decision module is responsible for deciding the predicted state of the VM on the bases of predicted values of CPU, Memory, and Storage.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

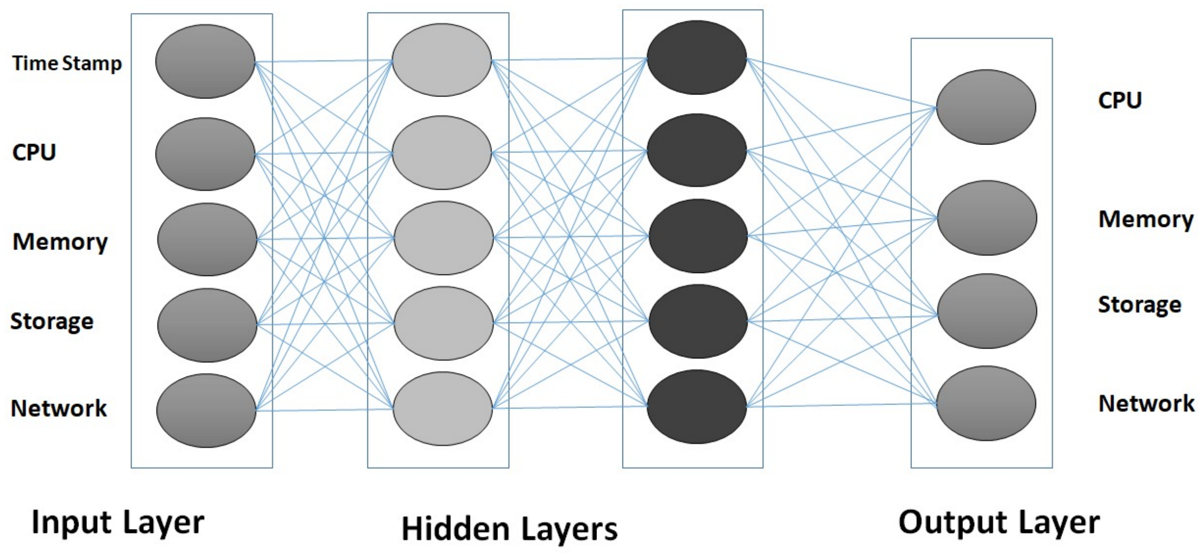


Figure 4. 2. 3: The NN architecture where we have 5 input neurons and each hidden layer is fully connected layer and 4 output layer neurons.

The results of the final model are depicted in Table VII. The results of the final single training are not that good compared to the results of the multi model approach. The training results shows a decline whenever we try to insert multiple parameter as input because they act as noise for each other. For further work it is recommended to train model using a different dataset and other ML approaches.

Table VII: The result of the final trained model.

		NN-MLP	
Output	Input data	Train Accuracy	Test Accuracy
cpu,mem,disk, network	cpu,mem,disk, network	0.85	0.83

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

4. 3 Decision Engine:

After predicting the future CPU, Memory and Disk utilization we need to decide the predicted status of the VM for which we were predicting. So, it classifies the predicted values into three classes in terms of utilization condition.

a) **Overload**

b) **Normal load**

c) **Low load**

If the predicted state is the overload condition this will mean that the system requires a scale-out for that kind of VM. But before scaling out it checks whether other VM are serving for that service are also overloaded or not and after that, it takes the decision and stores the prediction results in the directory. The Logic for the decision engine is shown in Figure 4. 3. 1. The decision of overload condition is measured using;

Max utilization	**Normal**	**Minimum**
C-max>= CPU utilization above 90 %	60<=C-medium<90	C-low<60
R-max>= Memory utilization above 85 %	60<=R-medium<85	R-low<60
S-max>= Storage Utilization above 90%	60<=S-medium<90	S-low<60

Failure State= C-max || R-max ||S-max

Normal State = ~ (C-max & R-Max& S-Max) & (C-medium || R-medium ||S-medium)

Underload State= (C-low & R-low & S-low)

Hence, the logic of the predicted state of the resource utilization can easily be converted to system state using the logic above.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

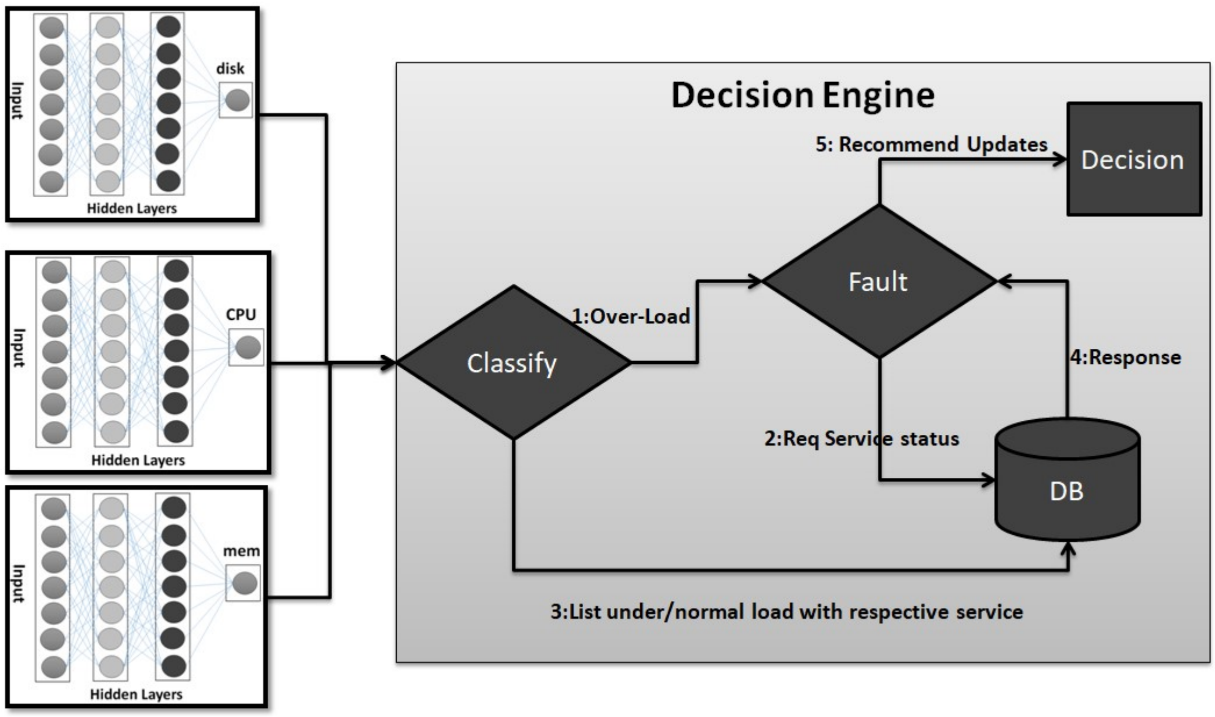


Figure 4. 3. 1: The Logic of the decision engine.

4. 4 The Experimental Test-Bed

The experiments are divided into three phases. Firstly, NSSF and implementation of OAI-components inside M-CORD have been considered. Secondly, the integration IBN-application from the top is explained for showing the simplification in the configuration and policy management of CORD. Third part interprets the results after integration of ML models and Monitoring tool.

The first experimental Test-Bed consists of OAI (OpenAir interface) eNB and EPC integrated with M-CORD [30]. The test-bed consists of two parts. First is the setting up of IBN application and second is the network slicing scenario. We based our development of network slicing scenario using M-CORD version 4.1 as a cord in a box (CiaB) deployment. This allows for a virtualized environment that can exist in a single physical server. The machine that houses the deployment has 64GB of RAM, 20 physical cores and is running on Linux 14.05. The EPC

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

represented by open-air-CN is running on a modified version based on commit a58735fe, also the eNB version of the OAISIM implementation is the latest master branch. The EPC is running on Linux 16.04 and eNodeB on Linux 14.04 both with kernel modifications i.e. 4.7.7 oaiepc kernel for the EPC and Linux 3.19.0-61-low-latency kernel for the eNB. The NSSF was developed using python and is running on Linux 16.04. Communication between MME and NSSF is done via RPyC (Python native RPC). A Python API is needed to enable communication between MME and NSSF. Also, XOS Services were created with a mix of python, and yaml scripting [30].

For setting up the environment four VM images had previously been created using the cloud-init package which is required for creating VM images that can be used by OpenStack. The EPC image was obtained from the oai-scenario repo of the developer aweimeow who also changed the standard OAI architecture for compatibility with M-CORD [29] [30]. The eNB, vMME and NSSF images were created as part of the research. The approach followed for slicing the network is based on 3GPP LTE architecture for which considered network slicing in the core part where there is no network slicing available on the access part.

When each VNF is instantiated i.e. The EPC comprised by the vHSS, vSPGW-C, and vSPGW-U will be provisioned by a unique VM image but each XOS service that represents the VNF will have some particular directive that affects the configuration of the instance, so even though all the EPC are been instantiated using the same VM, they will act differently thanks to the configuration specific to each VNF. For the vMME, NSSF and eNB case, the VM images are particular to each service for which they are not reused by any other VNF. Moreover, system configurations are detailed in Table VII.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Table VIII: Shows Image Distribution for the test-Bed

VM Image Distribution		
VNF		Image-name
EPC	vHSS	image-oaicn
	vSPGW-C	
	vSPGW-U	
vMME		image-vmme
eNB-OAISIM		image-oaisim
NSSF		image-nssf

To showcase the functionality of the NSSF, multiple slices were configured on the platform. Our purpose is to have two UE connecting to the mobile network and have each of them routed through different paths for connecting to the data network. The initial underlying topology for two slices of the test-bed can be appreciated in Figure 4. 4. 1.

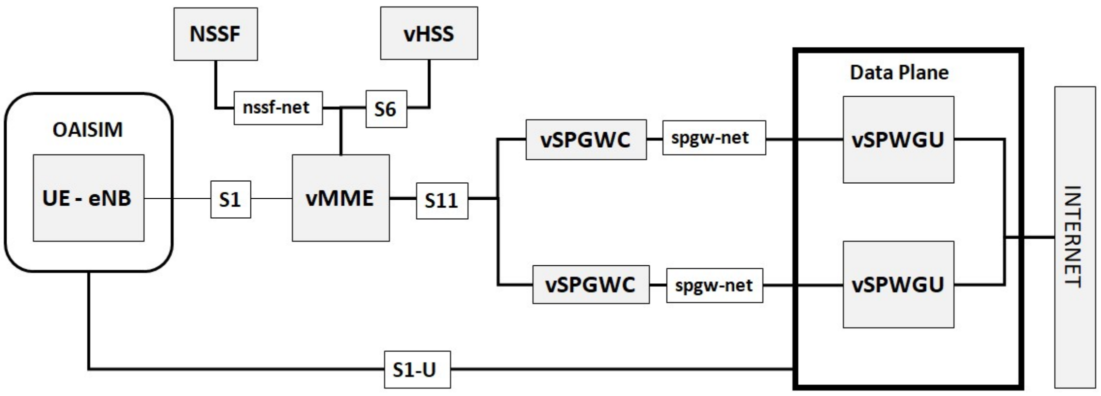


Figure 4. 4. 1: Initial network test-bed configuration with 2 slice example.

OAISIM is the actor that triggers the Mobile Network test. Initially, we have it configured with two UE that have different IMSI. The last digit of this id represents a type of service (Relative to this test scenario: 1 being eMBB, 2 being IoT). As the IMSI is a combination of the PLMN and IMSI, the first five digits are always common 20893. We are working in the same PLMN so the eNB and vMME need to match these five numbers before any UE starts registration into the network. The rest of the numerical that represent the IMSI, are particular to each UE. In our scenario, UE1 has 208930100001111 and UE2 has 208930100000402 defined as their IMSI.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

The vMME gets this information from the eNB and by communicating with the vHSS, verifies that this numerical Id exists in the UE Data Base (UE allowed to be served by the network operator), once the UE is registered into the network, the original OAI implementation of the vMME would select an SPGW on the basis of the TAC and TAI information that the eNB had to send during network registration. Instead, we changed this procedure in order to demonstrate our Slice Selection scenario. Modifications had to be done on the vMME for it to achieve communication with the NSSF, so we created a new vMME image that contains the modifications. TAC and TAI were discarded for SPGW selection and IMSI was used in their place. As vMME and NSSF are two different developments, an API was created to interact between both modules, and achieve exchange of messages. Figure 4 .4.2 illustrates this process.

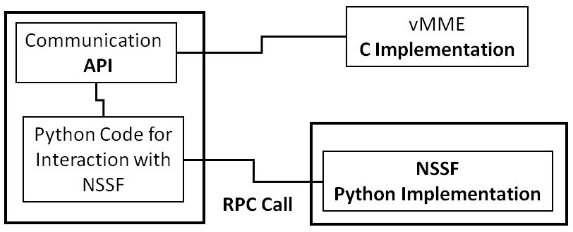

Communication API
vMME C Implementation
Python Code for Interaction with NSSF
RPC Call
NSSF Python Implementation

Figure 4. 4. 2: Python API between NSSF and vMME

The test Results after creating two slices and for the two user equipment are shown in the figures 4.4.3 and figure 4.4.2.

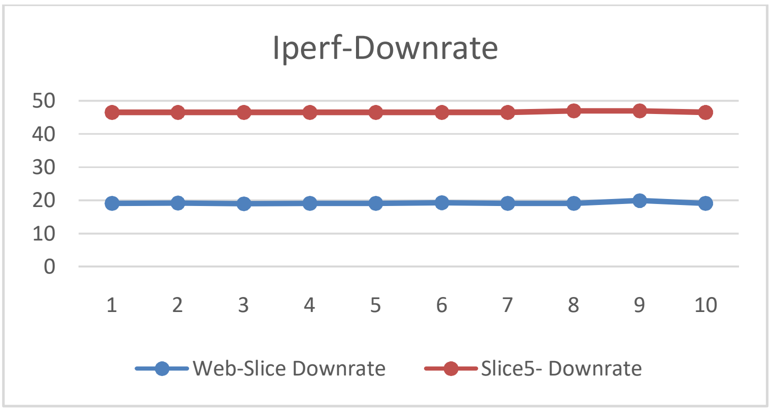


Figure 4. 4. 3: Down-Rate for the configured slices using the UE.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

The down-rate shown in figure 4.4.3 stayed stable from start to end which means that the system provides stable download rates. However, in the case of uprate we can clearly see a disruptive behavior at the beginning of the test when video slice started but after that, it maintains stable QoS. For performing these test the QoS parameters were not decided before the creation of the slices so through the iperf test we tried provided the input QoS and system behavior is shown in the figure below.

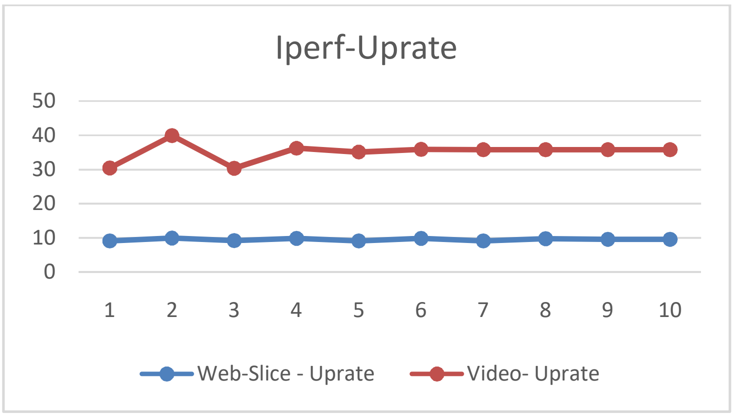


Figure 4. 4. 4: Up-rate graph for the slices.

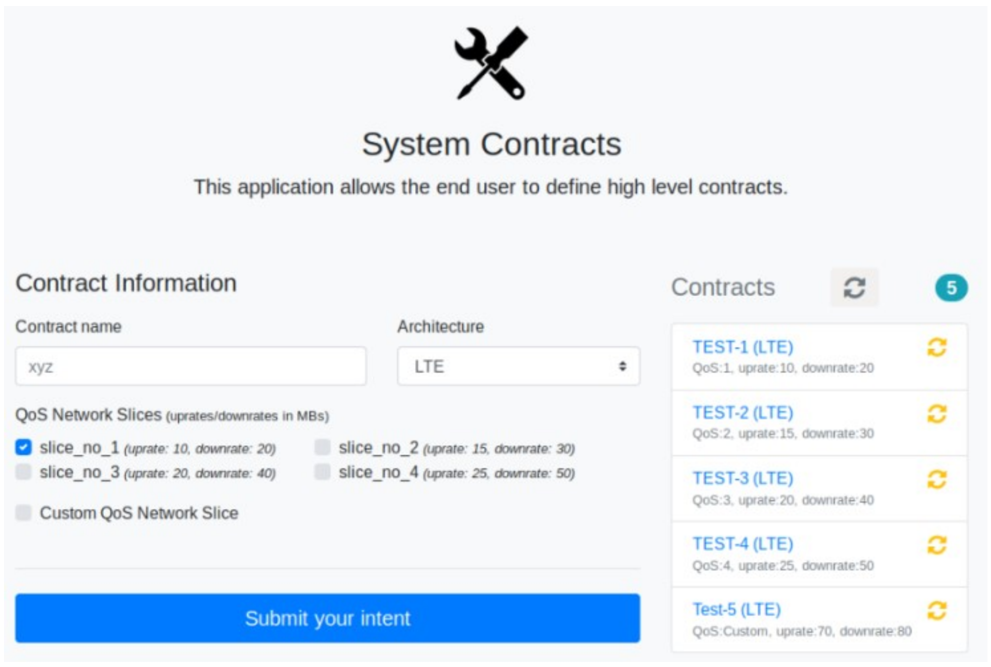
4. 5 Experiment after IBN Application integration

For this test the test-bed consists of IBN-application, M-CORD with NSSF and OAI based network slicing scenario, as in Figure 4. 0. 1. The purpose of the test-bed is to automate network configuration and management, for testing purpose we inserted 5 slice configurations from the top using the contracts as shown in Figure 4. 5. 1. Actually, by slices, we meant a number of a different combination of SPGWC and SPGWU each with different configurations from the top. This slicing model can be referenced from [5] [34]. Also, in 3GPP the concept of core network slicing is also the generation of multiple core network instances of SPGW with different QoS. So, we set a test-bed with the 5 slices with different QoS. Our goal of representation for IBN application, for instance, is that it should replicate the configurations to the physical layer. We have performed multiple iperf-test on the user-plane functions in the core network that is SPGW-U gateway. The results show that the information provided from the top is replicated on the physical layer properly. As Figure 4. 5. 2 and Figure 4. 5. 3 represents the results of QoS

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

allocation in terms of up-rate and down-rate for each of the slices with time intervals. Further, for more intensive bandwidth results we require very sophisticated hardware on RAN part and the core part, the capability of CORD can be referenced from the demo provided by Radisys [36].

Figure 4. 5. 1, shows the front end of the IBN-application, it accomplishes our challenge of simplification in the configuration of next-generation platforms. As we can see in the front end we configured 5 test slices for our system each having different QoS configurations. Also, the different slices were marked with S-NSSAI 1 only because we don't have the developed infrastructure for all the network slices.



System Contracts

This application allows the end user to define high level contracts.

Contract Information

Contract name: xyz

Architecture: LTE

QoS Network Slices (uprates/downrates in MBs)

- [✓] slice_no_1 *(uprate: 10, downrate: 20)*
- [ ] slice_no_2 *(uprate: 15, downrate: 30)*
- [ ] slice_no_3 *(uprate: 20, downrate: 40)*
- [ ] slice_no_4 *(uprate: 25, downrate: 50)*
- [ ] Custom QoS Network Slice

Submit your intent

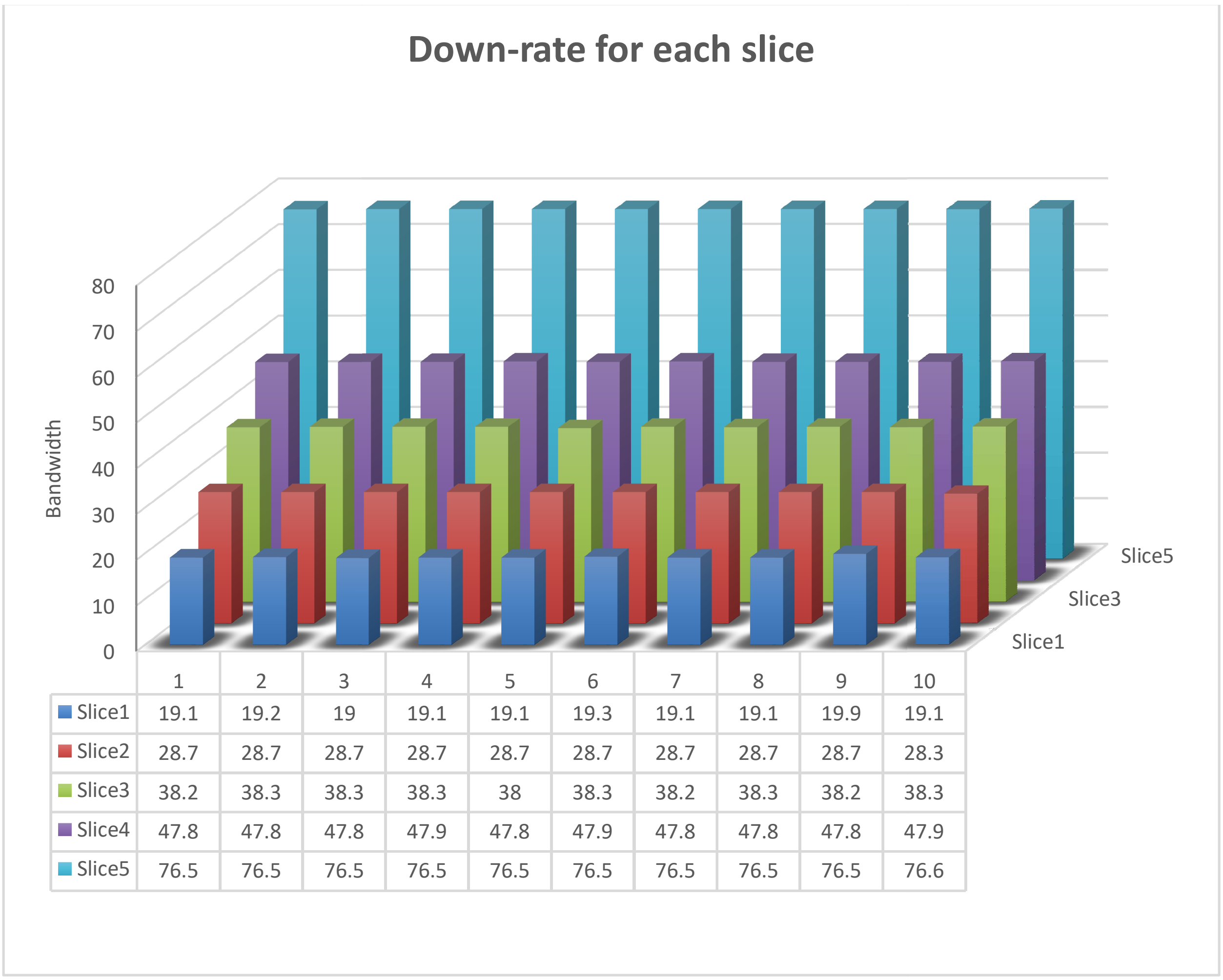
Contracts 5

- TEST-1 (LTE) QoS:1, uprate:10, downrate:20
- TEST-2 (LTE) QoS:2, uprate:15, downrate:30
- TEST-3 (LTE) QoS:3, uprate:20, downrate:40
- TEST-4 (LTE) QoS:4, uprate:25, downrate:50
- Test-5 (LTE) QoS:Custom, uprate:70, downrate:80

Figure 4. 5. 1: Shows the front end of the IBN-application with the five test slices.

In Figure 4. 5. 2, we can see that the entire slice configuration provided from top created five different slices and for each slice, the down-rate stays stable for all the 10 tests performed. We can see the values of the down rate provided from the top is maintained by the platform.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

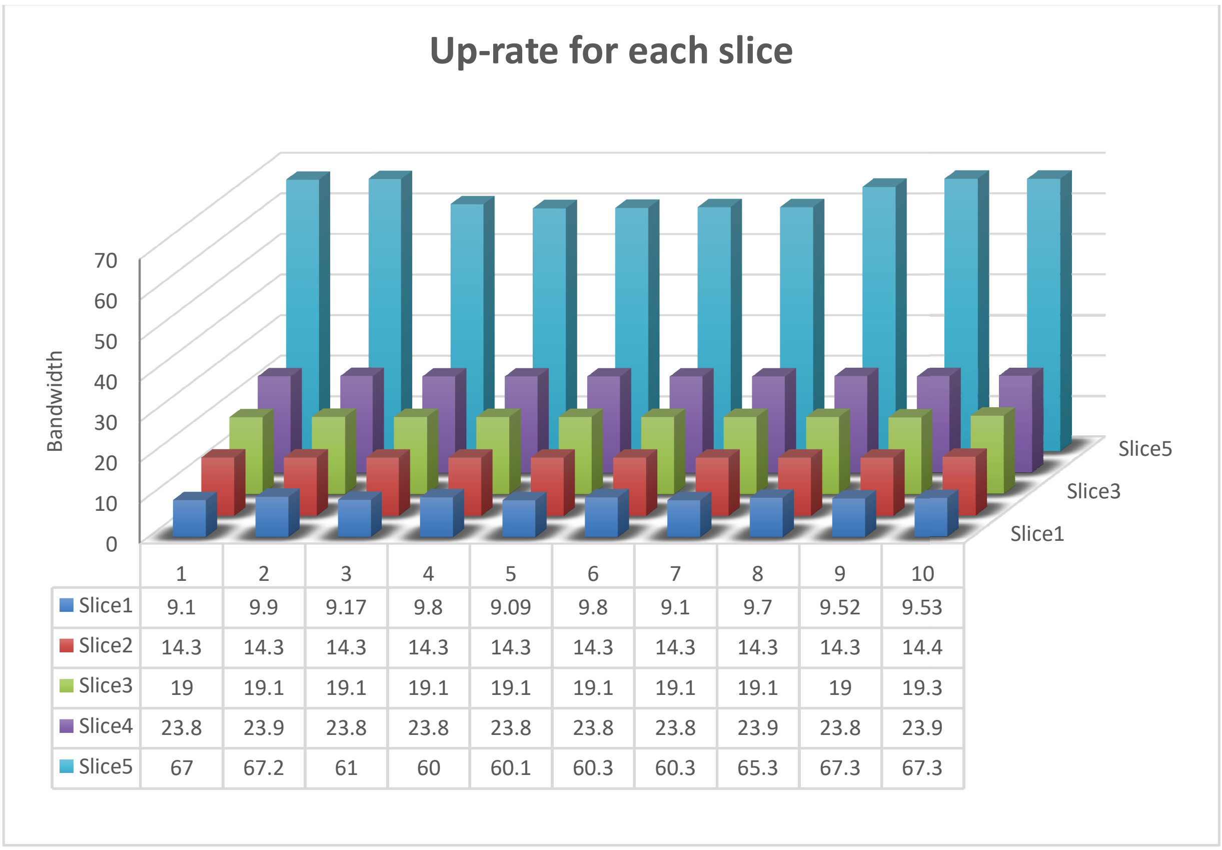


	1	2	3	4	5	6	7	8	9	10
🟦 Slice1	19.1	19.2	19	19.1	19.1	19.3	19.1	19.1	19.9	19.1
🟥 Slice2	28.7	28.7	28.7	28.7	28.7	28.7	28.7	28.7	28.7	28.3
🟩 Slice3	38.2	38.3	38.3	38.3	38	38.3	38.2	38.3	38.2	38.3
🟪 Slice4	47.8	47.8	47.8	47.9	47.8	47.9	47.8	47.8	47.8	47.9
🟦 Slice5	76.5	76.5	76.5	76.5	76.5	76.5	76.5	76.5	76.5	76.6

Figure 4. 5. 2: Shows the down-rate for each of the slice.

Similarly, Figure. 4. 5. 3shows the up-rate stability for each of the created slices. The end to end test can qualify the quality of service using the test-bed because OAI-SIM does not support more than 20 users and also don't provide the Bandwidth of more than 20, so the test shown is directly performed on the User-plane function i.e. SPGW-U and they perfectly show the mapping of the QoS provided from top is allocated properly on the physical layer.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY



	1	2	3	4	5	6	7	8	9	10
🟦 Slice1	9.1	9.9	9.17	9.8	9.09	9.8	9.1	9.7	9.52	9.53
🟥 Slice2	14.3	14.3	14.3	14.3	14.3	14.3	14.3	14.3	14.3	14.4
🟩 Slice3	19	19.1	19.1	19.1	19.1	19.1	19.1	19.1	19	19.3
🟪 Slice4	23.8	23.9	23.8	23.8	23.8	23.8	23.8	23.9	23.8	23.9
🟦 Slice5	67	67.2	61	60	60.1	60.3	60.3	65.3	67.3	67.3

Figure 4. 5. 3: shows the up-rate for each of the slice.

4. 6 The test-bed and results including ML and Monitoring enabled:

The test-bed for final results consists of the following

- The IBN and M-CORD
- The integration of ML models and Decision engine with IBN application
- The deployment of the monitoring tool inside CiaB test-bed.

The IBN application is developed using the web on the front end and Python as the back-end language with SQL database. All the modules mentioned in Figure 4.6.1 for the IBN-Application are Rest-Enabled. Similarly, python is used for developing the ML- Model using standard libraries and SciKit-Learn. Nagios is used as monitoring and communication between Nagios and ML model is done using Rest Interfaces. Figure 4. 6. 1 shows the detailed architecture of the

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

overall test-bed, where we have integrated the whole test-bed. So, whenever user traffic will increase the burden on the network will result in scaling of the network instances.

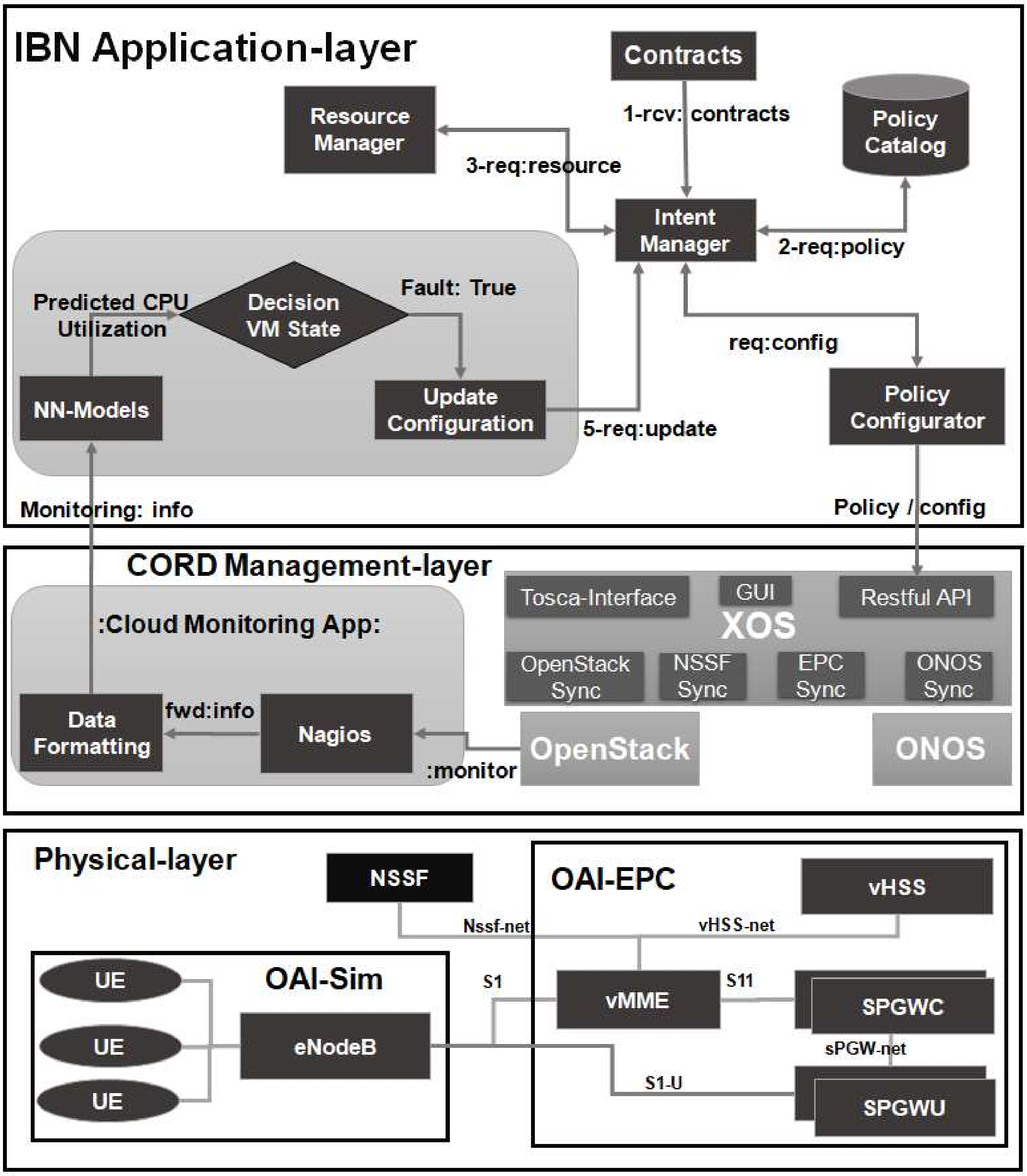


Figure 4. 6. 1: Overall test-bed with the working mechanism.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

The test is performed by initiating one slice that serves video and initially we initiated one SPGWU instance that serves as the provider gateway for video slice. For the test traffic was streamed using 1 UE's. We observed the values of CPU-Utilization, predicted values of CPU and also the number of instances of the slice was observed. Then gradually the No. of UE were increased until 3 and with the increased amount of streaming traffic on the slice hence requiring an update. Figure 4.6.2 shows the results of the test performed on the video slice where the number of user equipment and amount of traffic streamed was increased hence requiring an increase in no of instances. The results show how the system decided automatically to increase the number of instances that were serving the Video Slice. As we can see in the figure 4.6.2 after the 6th time interval the number instances were increased as the predicted CPU-Utilization went up to 98 percent. Hence, we can have a system that can automate the system resources autonomously using the platform described above.

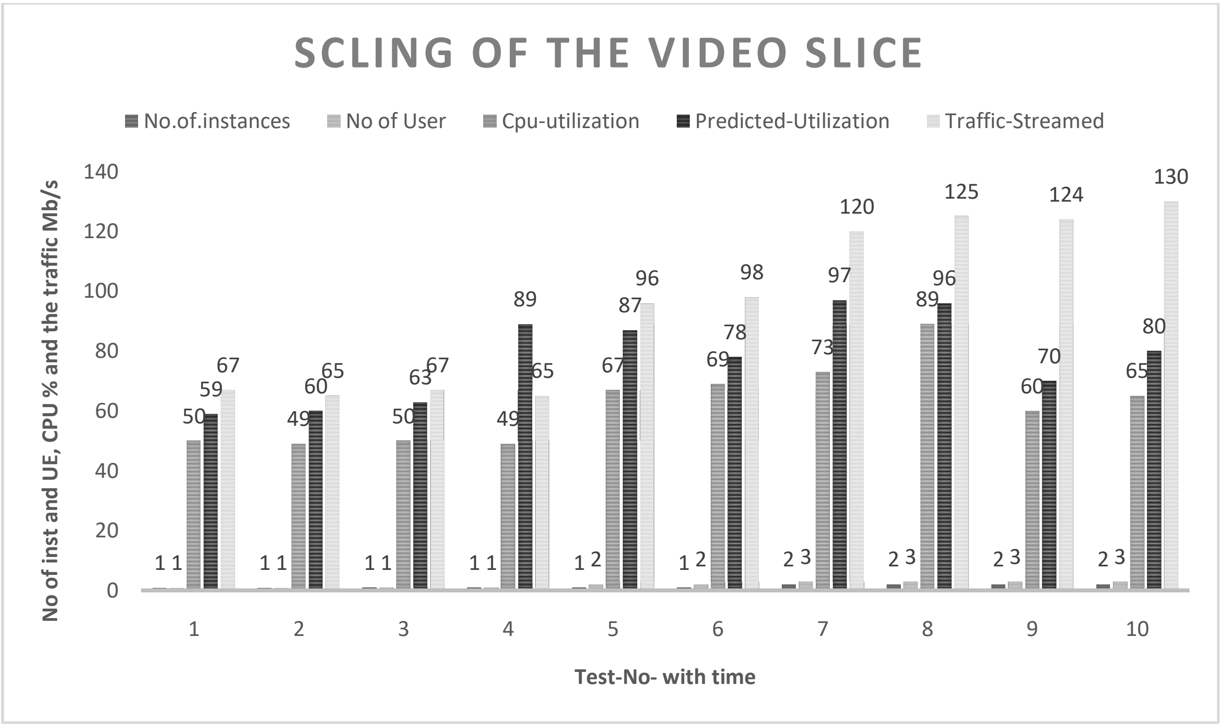


Figure 4. 6. 2: Figure Shows the dynamic update in the no instances with the increase in traffic and no of users

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

The rigorous tests were not performed considering physical resource availability. The video was streamed using FFMPEG which allow us to stream video using the Command-line. As the UE created under OAI-Sim are only enabled with Command-line Interface. While trying to further increase the user equipment or amount of traffic through the OAI-SIM it always crashes causing the failure in the test-bed. The future plan is to use the test-bed on more stable physical infrastructure so that we can test multiple slices and can create many UE and slices.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Chapter 5
Conclusions

The dynamic service provisioning in next-generation networks also required dynamic management for the resources. This thesis work developed an automatic procedure that can manage resources and can provide self-organization, self-healing and self-management capabilities for the VNFs serving different slices. One of the major achievement explained in this work is IBN-Application that provided simplification of configuration and management of the next-generation networks. Additionally, the IBN-Application is used here to replicate the users SLA to the physical system and it determines optimized resource allocation and also assures the QOE (quality of service). Furthermore, the system achieved the goal of catering the dynamic user demands beforehand that can cause a failure state. The Machine learning algorithms were used to predict future resource usage depending on current usage. It is evident that if the future usage is high resulting in resource demand and in light of the predicted demand system can decide the configuration and policy for a future time. Hence, the platform can automatically manage the dynamic resource demand and also automates the policy and configuration management for next-generation platforms.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Bibliography

[1] B. Blanco et al., “Technology pillars in the architecture of future 5G mobile networks: NFV, MEC and SDN,”Comput. Stand. Interfaces, vol. 54, no. April 2016, pp. 216–228, 2017.

[2] NGMN, 5G White Paper, 2015.

[3] White Paper, “Central Office Re-architected as a Datacenter (CORD)” March 14, 2016

[4] L. Peterson et al., “XOS: An Extensible Cloud Operating System,” ACM BigSystems 2015, June 2015.

[5] M-CORD, “M-CORD as an Open Reference Solution for 5G Enablement.”

[6] BessemSayadi and Laurent Roullet, Nokia Bell-Labs France ”5G: Platforms Not Protocol” IEEE Sofwarization, January 2018.

[7] 3GPP TR 28.801 V15.1.0 (2018-01), “Technical Specification Group Services and System Aspects; Telecommunication management; Management:Study on management and orchestration of network slicing for next generation networks;” Release 15, Nov. 20.

[8] “Impact of SDN and NFV on OSS/BSS”, ONF Solution Brief, March 1,2016.

[9] Cleverson Nahum, Jose Soares, Pedro Batista and AldebaroKlautau "Emulation of 4G/5G Network Using OpenAirInterface", SIMPO´ SIO BRASILEIRO DE TELECOMUNICAC¸ OES E PROCESSAMENTO DE SINAIS - SBrT2017, 3-6 DE SETEMBRO DE 2017, S ˜ AO PEDRO, SP.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

[10] NavidNikaein, Mahesh K. Marina, SaravanaManickam, Alex Dawson, Raymond Knopp, Christian Bonnet "OpenAirInterface: A Flexible Platform for 5G Research".

[11] Fred Aklamanu, Sabine Randriamasy, Éric Renault, Imran Latif, AbdelkrimHebbar, Alberto Conte, Bilal Al Jamal, WardaHamdaoui. "Demo: Intent-Based 5G IoT Application Slice Energy Monitoring". IFIP Networking 2018. Zurich, Switzerland: may 2018.

[12] Fred Aklamanu, Sabine Randriamasy, Éric Renault, Imran Latif and AbdelkrimHebbar. "Intent-Based Real-Time 5G Cloud Service Provisioning". IEEE Globecom Workshops 2018. Abu Dhabi, United Arab Emirates: dec. 2018, pages 1-6.

[13] John J. Prevost, KranthiManojNagothu, Brian Kelley and Mo Jamshidi "Prediction of Cloud Data Center Networks Loads Using Stochastic and Neural Models" 6th International Conference on System and Systems Engineering.

[14] Boutaba et al., Mohammad A. Salahuddin, NouraLimam, Sara Ayoubi, NashidShahriar, Felipe Estrada-Solano and Oscar M. Caicedo, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities", Journal of Internet Servicesand Applications (2018)

[15] Jun Liu, Shuyu Chen, Zhen Zhou, and Tianshu Wu, "An Anomaly Detection Algorithm of Cloud Platform Based on Self-Organizing Maps" Hindawi Publishing Corporation Mathematical Problems in Engineering, Volume 2016, Article ID 3570305.

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

[16] Ameen Alkasem, Hongwei Liu, DechengZuo and Basheer Algarash" Cloud Computing: A model Construct of Real-Time Monitoring for Big Dataset Analytics Using Apache Spark", IOP Conf. Series: Journal of Physics: Conf. Series 933 (2018) 012018.

[17] GokalpUrul, "Energy Efficient Dynamic Virtual Machine Allocation with CPU Usage Prediction In Cloud Data Centers" a thesis submitted to the graduate school of engineering and science of bilkent university.

[18] Kentis, A. M., OlloraZaballa, E., &Soler, J. (2018), "Policy Framework for the Next Generation Platform as a Service," In Proceedings of 27th European Conference on Networks and Communications (pp. 125-9), IEEE. DOI: 10.1109/EuCNC.2018.8443260

[19] TS23.501 Section 5.15.2 "Technical Specification Group Services and System Aspects, System Architecture for the 5G System, Stage 2 (Release 16)"

[20] "The Next Generation Platform as a Service" "Cloudifying Service Deployments in Telco-Operators Infrastructure "

[21] Van Rossem, S, Sayadi, B, Roullet, L, Kentis, AM, Paolino, M, Veitch, P, Berde, B, Labrador, I, Ramos, A, Tavernier, W, OlloraZaballa, E&Soler, J 2018, A Vision for the Next Generation Platform-as-a-Service. in Proceedings of 2018 IEEE 1st 5G World Forum. IEEE, pp. 14-19, 2018 IEEE 1st 5G World Forum, Santa Clara, United States, 09/07/2018.

[22] Kentis, AngelosMimidis; OlloraZaballa, Eder; Soler, José; Bessem, S.; Roullet, Laurent; Van Rossem, S.; Pinneterre, S.; Paolino, M.; Raho, D.; Du, X.; Mariani, L.; Ramos, A.; Labrador, I.; Broadbent, A.; Zembra, M. "The Next Generation Platform as a Service

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

Cloudifying Service Deployments in TelcoOperators Infrastructure” Proceedings of the 25th International Conference on Telecommunications (ICT 2018).

[23] Paul Lipton, Derek Palma, Matt Rutkowski, Damian Andrew Tamburri, “TOSCA Solves Big Problems in the Cloud and Beyond!” DOI :10.1109/MCC.2018.111121612, 12-Jan-2018.

[24] “Introducing ONOS - a SDN network operating system for Service Providers” ONF, White Paper 2014.

[25] OpenAirInterface public license v1.1 [online],Available:https://www.openairinterface.org/?page_id=698 (Accessed: 5 Jan, 2019)

[26] OpenAirInterface System Emulation[online], Available:https://gitlab.eurecom.fr/oai/openairinterface5g/wikis/OpenAirLTEEmulation (Accessed: 5 Jan, 2019)

[27] OpenAirInterfacefreeDiameter usage [online], Available:https://gitlab.eurecom.fr/oai/freediameter (Accessed: 9 September, 2018)

[28] OpenAirInterface Project [online], Available: https://gitlab.eurecom.fr/oai/openairinterface5g/wikis/home (Accessed: 5 Jan, 2019)

[29] https://github.com/aweimeow/oai_scenario/oai_scenario in M-CORD (Accessed: 5 Jan, 2019).

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

[30] J.DiazRavera, T. Ahmed Khan, A, Mehmood& Wang-Cheol S. (2018), "Network Slice Selection Function on M-CORD," KNOM Review '18-02 Vol.21 No.02.

[31] "http://gwa.ewi.tudelft.nl/datasets/gwa-t-13-materna" (Accessed: 16 May, 2019).

[32] ETSI GS ZSM 006 V1.1.1 (2018-05), "Zero touch network and Service Management (ZSM); Proof of Concept Framework".

[33] "https://docs.openstack.org/infra/system-config/grafana.html." (Accessed: 16 May, 2019).

[34] Radisys Solution Brief: Radisys M-CORD: The Open Platform for Emerging 5G Applications.

[35] Jim Hodges, "The Open Road Migration: Rearchtecting Service Innovation Models " A custom Heavy report for Radysis.

[36] "Radysis Demonstration of end-to-end network slicing at MWC-A 17 Multi-Access CORD" LINK:"https://hub.radisys.com/videos".

제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY