QoS Enabled Video Telephony with a Virtualized HSS in a 4G EPC Environment

Hanieh Alipour

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science : Computer Science at

Concordia University

Montréal, Québec, Canada

November 2013

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis is prepared

By: Hanieh Alipour

Entitled: "QoS Enabled Video Telephony with a Virtualized HSS in a 4G EPC Environment"

Submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

Complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final Examining Committee:

______________________________ Chair
Dr. R. Witte

______________________________ Examiner
Dr. T. Eavis

______________________________ Examiner
Dr. L. Narayanan

______________________________ Supervisor
Dr. R. Glitho

Approved by: ______________________________
Chair of Department

__________2013

____
Dr. Christopher W. Trueman
Dean, Faculty of Engineering and
Computer Science

ABSTRACT

QoS Enabled Video Telephony with a Virtualized HSS in a 4G EPC Environment

Hanieh Alipour

Video Telephony is the real time exchange of voice and video between end-users. It is the basis of a wide range of applications (e.g. Multiparty games, distance learning). Quality of service (QoS) enables network performance control for meeting specific applications and/or end-user requirements. It is a differentiating factor for service providers. Evolved Packet Core (EPC) is the new core network for 3GPP 4G networks. Home Subscriber Server (HSS) is the standardized master database of 3GPP next generation networks including video telephony networks and EPC. It contains the subscription related information that is needed to support the network entities when they handle sessions. The constant increase in the number of subscribers is one of the challenges for future mobile networks including video telephony networks and EPC. Virtualization is a technique used to emulate the physical characteristics of resources. It enables efficiency in resource usage and is a key technology for scalability and elasticity.

This thesis proposes an architecture for QoS Enabled video telephony with a Virtualized HSS (VHSS) in a 3GPP 4G environment. It makes two main contributions. Firstly, it proposes a differentiated QoS service delivery platform that relies on EPC. This platform enables the provisioning of a refined differentiated QoS scheme which allows prioritization between different sessions of a same video telephony application running on a same network. This new scheme is a differentiating factor for service providers. Second it proposes a preliminary mechanism for a scalable and elastic HSS in order to cope with the increasing number of subscribers. This is done by decomposing the HSS into three main layers (diameter layer, database computation layer and storage layer). Each of these layers are virtualized and can grow/shrink independently. We have built a proof of concept prototype to demonstrate the feasibility of the proposed architecture. Performance measurements have also been made to evaluate viability.

Acknowledgments

First, I would like to express my sincere gratitude to my supervisor Dr. Roch Glitho for all his continued support, assistance and encouragement. This thesis would not have been possible without his help, guidance and valuable advices. It's been an honor and a pleasure to work with him and learn from his knowledge and expertise during the research period.

Next, I would also like to thank Dr. Fatna Belqasmi for all her help, advices, and comments during the research period.

I’m grateful to Dr. Eavis and Dr. Narayanan for serving on my thesis committee. I have appreciated your valuable feedbacks and your generosity with your time.

I am grateful to Dr. Roch Glitho and Concordia University for their financial support.

Last but not least, my personal thanks go to my dearest parents and brother who never ceased to give me moral support and were there when I needed them. You always understood and supported the choices I made during this long and personal journey.

TABLE OF CONTENT

List of Figures

List of Tables

List of Acronyms and Abbreviations

3GPP	3rd Generation Partnership Project
AAA	Authentication, Authorization and Accounting
ACID	Atomicity, Consistency, Isolation and Durability
AF	Application Function
AS	Application Server
ARP	Allocation and Retention Priority
API	Application Programming Interfaces
CAS	Cache Accelerator Server
CIB	Context Information Base
DAS	Direct Attached Storage
DBMS	Database Management System
DiffServ	Differentiated Service model
DFS	Distributed Fault-tolerant Storage
ePDG	Evolved Packet Data Gateway
EPC	Evolved Packet Core
ElasTraS	Elastic TranSactional relational
FE	Front End
GBR	Guaranteed Bit Rate

GPRS	General packet radio service
HDFS	Hadoop Distributed File System
HSS	Home Subscriber Server
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
IETF	Internet Engineering Task Force
IntServ	Integrated Services model
IEEE	Institute of Electrical and Electronics Engineers
IMS	IP Multimedia System
IPSR	IP Traffic Symmetry Ratio
JSON	JavaScript Object Notation
LAN	Local Area Network
LTE	Long Term Evolution
MME	Mobility Management Entity
M2M	Machine to Machine
MM	Metadata Manager
MP	Message Processor
MVNO	Mobile Virtual Network Operator
NIC	Network Interface Controller

NAS	Network Attached Storage
NoSQL	Not only Structured Query Language
PDNGw	Packet Data Network Gateway
PCEF	Policy and Charging Enforcement Function
PCC	Policy and Charging Control
PF	Proportional Fair Scheduler
PFMR	Proportional Fair Scheduler with Minimum/Maximum Rate constraints
PD	Publication and Discovery
QoS	Quality of service
QCI	QoS Class identifier
OTM	Owning Transaction Managers
ROS	Reliable Object Store
REST	REpresentational State Transfer
SGw	Serving Gateway
SCTP	Stream Control Transport Protocol
SAN	Storage Area Network
SQL	Structured Query Language
SPF	Session Prioritization Function
SDP	Service Development Platform

SIP	Session Initiation Protocol
SP	Service Provider
SRVLOC	Service Location Protocol
TCP	Transmission Control Protocol
TI-MAC	Traffic Intelligent - MAC
UTRAN	Universal Terrestrial Radio Access Network
UE	User Equipment
UMTS	Universal Mobile Telecommunications System
UDC	User Data Convergence
URI	Uniform Resource Identifier
UDR	User Data Request
UDA	User Data Answer
VoLTE	Voice over LTE
VMM	Virtual Machine Manager/Monitor
VLAN	Virtual Local Area Network
VPN	Virtual Private Network
VM	Virtual Machine
VHSS	Virtualized Home Subscriber Server
XML	Extensible Markup Language
YATE	Yet Another Telephony Engine

Chapter 1

1 Introduction

This chapter first presents an overview of the research domains. It then discusses the problem statement and motivation before presenting the thesis contributions, and finally, the thesis organization.

1.1 Research Domain

Video telephony is the conversational exchange of voice and video between end-users. It is the basis of a wide range of applications (e.g. Business conference calls, Multiparty games). Quality of Service (QoS) is a key ingredient of video telephony. It refers to the ability to control network performance for meeting applications and/or end-user requirements [1]. 3GPP 4G Evolved Packet Core (EPC) is the new IP based core network of the 4th generation of mobile networks. The EPC network supports service provisioning with QoS which enables QoS enabled video telephony applications [2] .

Virtualization is a technology that decouples computing function from physical hardware, including CPU, memory, storage etc. The purpose of Virtualization is to increase resource utilization and for multiple software or operating systems to share a single physical system. There are many different types of Virtualization such as Server Virtualization, Operating System Virtualization and Database Virtualization. There are several benefits that can be achieved by using Virtualization technology such as increased scalability, reduced administration costs and improved high availability [3].

1.2 Problem Statement and Motivations

Video Telephony is a real time multimedia application which is sensitive to delay and needs QoS in order to provide seamless service. End-users of a video telephony service provider may have different requirements on a similar application depending on the circumstances in which they use it (i.e. The actual session). For instance, more stringent requirements may be put on a video telephony session at initiation time or upgrading the requirements on a video telephony session after its initiation. These changes might result in downgrading or even terminating other video telephony sessions running on the same network, due to limited network resources. Therefore, Quality of Service (QoS) is important and there is indeed a need for a refined differentiated QoS that enables prioritization between the different sessions of a same video telephony application and the priorities also can be changed even during ongoing sessions. None of the existing Video telephony applications support differentiated QoS. It is actually not possible to support QoS (guarantee and differentiated) over the Internet because the Internet provides Best-Effort. 3G telecommunication networks do offer guaranteed QoS to some extent, but they do not provide the refined differentiated QoS. Video telephony over the EPC network can support guaranteed and differentiated QoS.

On the other hand, on current EPC networks the Home Subscriber Server (HSS) is a database which stores service profiles, subscriber profiles and roaming users' information. When several video telephony providers are using a shared EPC network then several of these providers use the same HSS. An increase in the number of subscribers can drive the network provider to deploy multiple HSSs. Multiple HSS can engender several problems including increased hardware, higher costs and extra administration. By providing Virtualized HSS (VHSS), there are several benefits for video telephony providers and an EPC provider. Video telephony providers can work independently, reuse of resources can be maximized and an EPC provider is not limited when adding more video telephony providers.

1.3 Thesis Contributions

The main contributions of this thesis are:

- A set of requirements for implementing the QoS Enabled Video Telephony with VHSS which runs in an EPC environment.
- Review of the state of the art relevant to our work with an evaluation summary in agreement with our requirements.

- The overall system architecture for the QoS Enabled Video Telephony with VHSS which runs in an EPC environment, an architecture for the QoS Enabled Video Telephony application, and a preliminary architecture for the VHSS.
- Describes the software architecture of the main functional entities, the operational procedures that show how the entities interact, the prototype architecture and the proof of concept prototype. It includes the performance measurements.
- A preliminary performance evaluation of the proposed architecture.

1.4 Thesis Organization

The rest of the thesis is organized as follows:

Chapter 2 introduces background knowledge related to QoS, EPC and Virtualization that will explain the concepts and ideas relevant to this thesis.

Chapter 3 introduces the requirements and the state of the art related to QoS Enabled Video Telephony and VHSS. Furthermore, it presents the evaluation of related works compared to our requirements.

Chapter 4 presents the proposed architecture for the QoS Enabled Video Telephony with VHSS which runs in an EPC environment. It also includes the main components and communication interfaces.

Chapter 5 is dedicated to the software architecture of the system components and presents the operational procedures that show how the entities interact. In addition, it presents the implemented proof of concept prototype and also includes the performance measurements.

Chapter 6 concludes the thesis by highlighting our contributions and lists some ideas for future research possibilities on QoS Video Telephony Applications with VHSS.

Chapter 2

2 Background Information on EPC, QoS and Virtualization

This chapter presents the background information that is relevant to this thesis research area. The background information covers four topics: Evolved Packet Core (EPC), Quality of Service (QoS) and Virtualization.

2.1 Evolved Packet Core (ECP)

In this sub-section, we first introduce the definition of EPC. Then, we present the architecture of the EPC which includes the main components. After that, we will present the diameter protocol. The 3GPP selected Diameter as the foundation for signaling EPC architecture and interaction between nodes is based on diameter interfaces.

2.1.1 Definition

EPC is the new IP-based core network of the 4th generation of mobile networks. It comes with the Long Term Evolution (LTE) access network. However, it is access technology agnostic and it can be used with legacy 3GPP access networks (e.g. GPRS and UTRAN) and non-3GPP access networks (e.g. Wifi and Wimax) [1]. It has a flat and simplified architecture, with a clear separation between control and data plans. It merges the circuit-switched domain and packet- switched domain that exists on 2G and 3G networks into one IP-based network.

2.1.2 Architecture

Figure 2.1 depicts the main components of EPC. The components can be classified based on the functionality as follows:

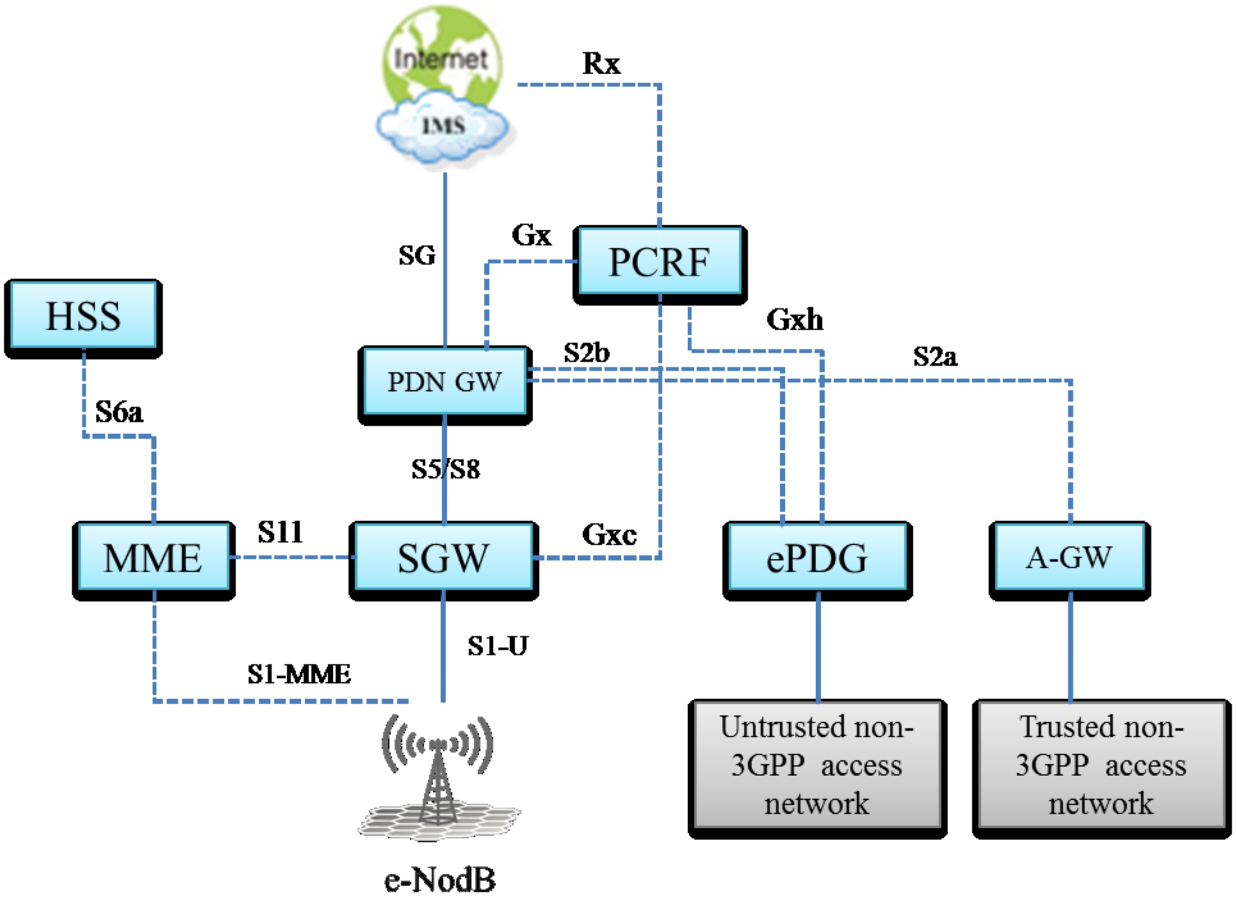

Figure 2.1: EPC Architecture

- Serving Gateway (SGw)

The SGw [1] is the access gateway for LTE technology. It is the connection point in the EPC that binds the 3GPP access networks and the core network. An end user joined to EPC is attached with one SGw which forwards and routes user data packets. Moreover, it acts as the local mobility anchor point for inter-eNodeB handover.

- Evolved Packet Data Gateway (ePDG)

The ePDG [1] is responsible for authenticating and authorizing end-users when connecting from untrusted non-3GPP access networks to the core network. It acts as a local mobility anchor point for handover for untrusted non-3GPP access networks.

- Packet Data Network Gateway (PDNGw)

The PDNGw [1] is the termination point of the packet data interface connecting EPC to external packet data networks. Each user can connect to multiple PDN-GW simultaneously to access multiple packet data networks. Furthermore, it performs various functionalities such as per-user-based packet filtering and is responsible for enforcing policy and charging control rules via the Policy and Charging Enforcement Function (PCEF).

- Mobility Management Entity (MME)

The MME [2] is the main entity for the EPC. It is responsible for selecting the PDN GW and the S-GW. In addition, the MME manages the tracking area list, supervises handovers between different base stations and is also responsible for paging subscribers in the IDLE state. It is engaged establishing and deactivating the bearer.

- Policy and Charging Rules Function (PCRF)

The PCRF [3] is a key entity in the EPC which provides policy control (e.g. QoS control) and charging control decisions. The PCRF includes flow control functionalities. Moreover, it takes the session information like bit rate and QoS requirements in order to create Policy and Charging Control (PCC) rules. It then forwards these rules to the PCEF which is part of the PDNGw to enforce the policy rules.

- Home Subscriber Server (HSS)

The HSS [4] is the master database for EPC. The HSS provides credentials and keys for some entities like MME to support authentication and authorization. It stores the subscription-related information such as user profile information, user location information, user security (network access control information for authentication and authorization) and user identification, numbering and addressing information. HSS uses Diameter Protocol for communication with other components. Figure 2.2 shows the interfaces of HSS. The network may need more than one HSS if the number of subscribers is high. When the operator uses multiple HSS then a Diameter Routing Agent can be deployed for finding the identity of the HSS which holds the subscription information.

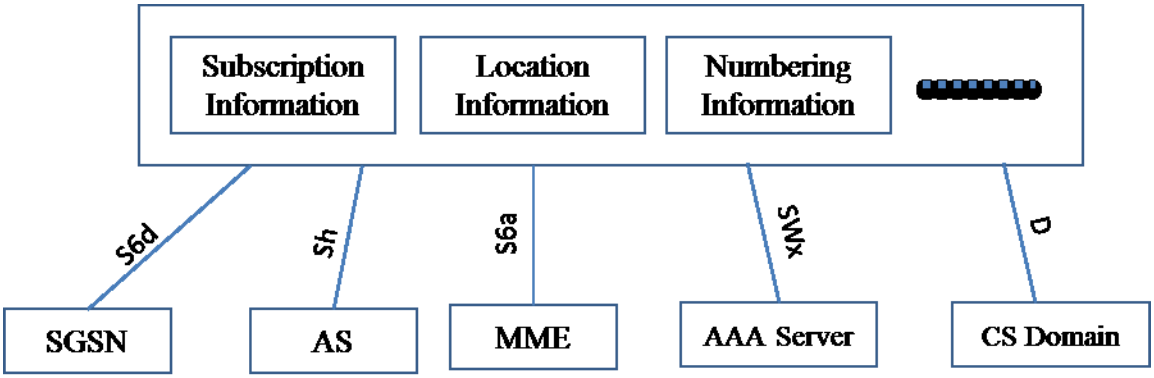

Figure 2.2: HSS Interfaces

2.1.3 Diameter Protocol

Diameter is a message-based protocol which is designed for Authentication, Authorization and Accounting (AAA) purposes. It has additional extensions (called application) which are created for supporting specific requirements [5]. Diameter runs on top of Transmission Control Protocol (TCP) or Stream Control Transport Protocol (SCTP) to provide reliable transport between two Diameter peers. Figure 2.3 shows the structure of Diameter Protocol.

The 3GPP selected Diameter as the foundation for signaling EPC architecture and interaction between nodes. This interaction is based on diameter interfaces. For instance, the HSS node includes two main parts: Diameter interfaces and the databases.

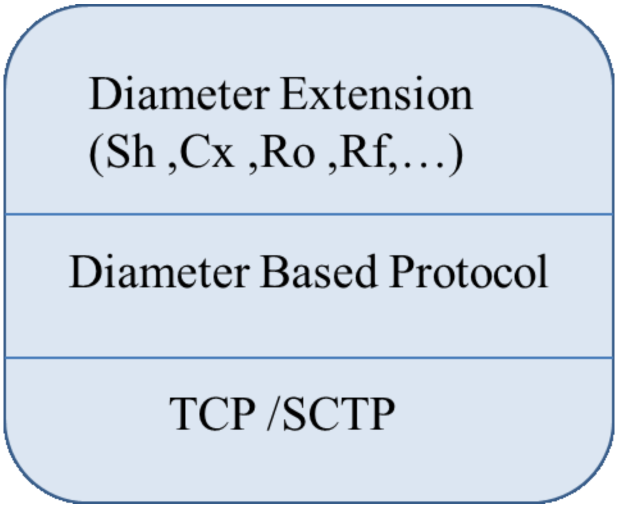

Figure 2.3: Diameter Protocol

We have three types of diameter nodes in the network: Client, Server and Diameter Agent. The client refers to the entity which requests the service and the Server is the entity which serves this request. The Diameter Agent is responsible for determining the right request and forwarding it to the correct destination. There are three types of agents as follows [1]:

- **Diameter Relay**: Based on the information in the message, it forwards it to the suitable destination.
- **Diameter Proxy**: It is similar to a Diameter Relay with additional policy rules.
- **Diameter Redirect**: Finds the proper destination address based on the routing function and replies to the node that sent the request.
- **Translator**: Performs translation between diameter and RADIUS protocols

2.2 Quality of Service (QoS)

In this sub-section we discuss QoS. We will first introduce the definition of QoS, and then present the main parameters. Next we will present the QoS Service levels before finally examining QoS in the EPC network.

2.2.1 Definition

In telecommunications, the word “Service” depends on the exchange of information through a telecommunications network which is provided by a service provider for the customer. In addition, the word “Quality” has a very broad meaning but in telecommunications Quality is the user service satisfaction [6]. There is no generally accepted definition for QoS but in general, QoS is the capability of the network to control performance based on application or user requirements. For meeting specific requirements, the QoS uses some mechanisms for controlling network performance. The QoS mechanisms are: packet classification, packet marking, packet scheduling, admission control and traffic shaping [7].

2.2.2 QoS Parameters

The QoS is described by a set of parameters which explain how the network treats packets while they are traveling over a network. The parameters are the following [6]:

- **Delay:** The experience of how long the packet takes to pass across the network.
- **Jitter:** Variations in the time between the IP packets arriving.
- **Packet loss rate:** The ratio of the number of packets that are dropped in the network and total transmitted packets.
- **Bit rate:** The bit rate of transferring packets through the network.

2.2.3 QoS service levels

The service level is the capability of the network to control QoS parameters and requirements. We have three service levels which are provided by the network [7] :

Best-effort service: the network (e.g. Internet) makes no guarantees regarding packet delivery. Problems such as network congestion have an effect on the best-effort service then some applications which are sensitive to the delay might not work properly.

Integrated service: It focuses on per-flow service that needs a certain resource reservation. The IETF Integrated Services model (IntServ) was developed by the IETF to support guaranteed service. The guaranteed service delivers strict guarantees on delay and bandwidth.

Differentiated service : The IETF offered a Differentiated Service model (DiffServ) that works on the provisioned-QoS model. In the DiffServ, the network traffic classifies the multiple classes of service based on varying QoS requirements.

2.2.4 QoS in EPC

The 3rd Generation Partnership Project (3GPP) has defined a bearer model for EPC to offer class-based differentiated QoS in which the EPC supports different traffic with different QoS levels. The bearer illustrates the level of granularity for QoS control and is a major element in EPC QoS.

The bearer is a virtual link between an end user and PDN-Gw in order to provide specific QoS. Furthermore, each bearer has set of parameters which describe the characteristics of the bearer such as: QoS Class identifier (QCI), Allocation and Retention Priority (ARP), Guaranteed Bit Rate (GBR), etc. For instance QCI is the number of QoS characteristics like delay and priority. All sessions with the same type of bearer receive the same treatment from the network [1]. We can categorize EPC bearers in two groups as shown in Figure 2.4: default bearer and dedicated bearer. The default bearer is set up when a user attaches to the EPC for the first time, and it is between the end-user and PDN-GW. The subscription information is used to allocate QoS parameters of the default bearer and the bearer remains as long as the user is attached to the network. On the other hand, in order to enhance QoS in the network a dedicated bearer is required. For example, for some services like VoLTE, the service provider wants to provide better user experience and this is where a dedicated bearer would be advantageous [8].

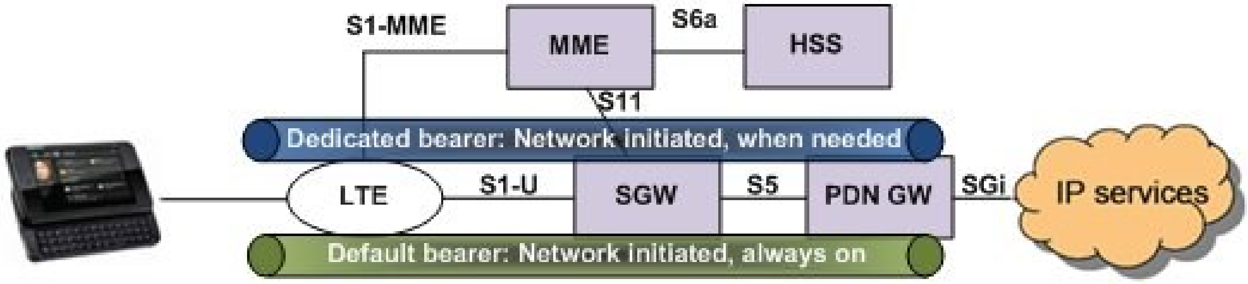

Figure 2.4: Default bearer and dedicated bearer

In the EPC, several nodes work together in implementing QoS. The PCRF checks the network information and policies which the operator configured and then defines the PCC rules. These rules are forwarded to PDN-GW which includes PCEF to establish bearers and mapping traffic to the bearers. Finally, the bearer is created between the user and PDN-GW .

2.3 Virtualization

This sub-section discusses Virtualization. First of all we will introduce the definition of Virtualization then outline its benefits. Next we will present different platforms for Virtualization. After that, we will discuss different types of Virtualization before finally talking about Virtualization and cloud computing.

2.3.1 Definition

Virtualization is a technology which creates a virtual resource from hardware, software, storage, network etc. Guest, Host and Virtualization layers are the three main components in a Virtualized environment (Figure 2.5) [9] . In other words, Virtualization is developed to achieve maximum resource utilization and to share these resources thereby ensuring business needs are met and resources are allocated dynamically [10].

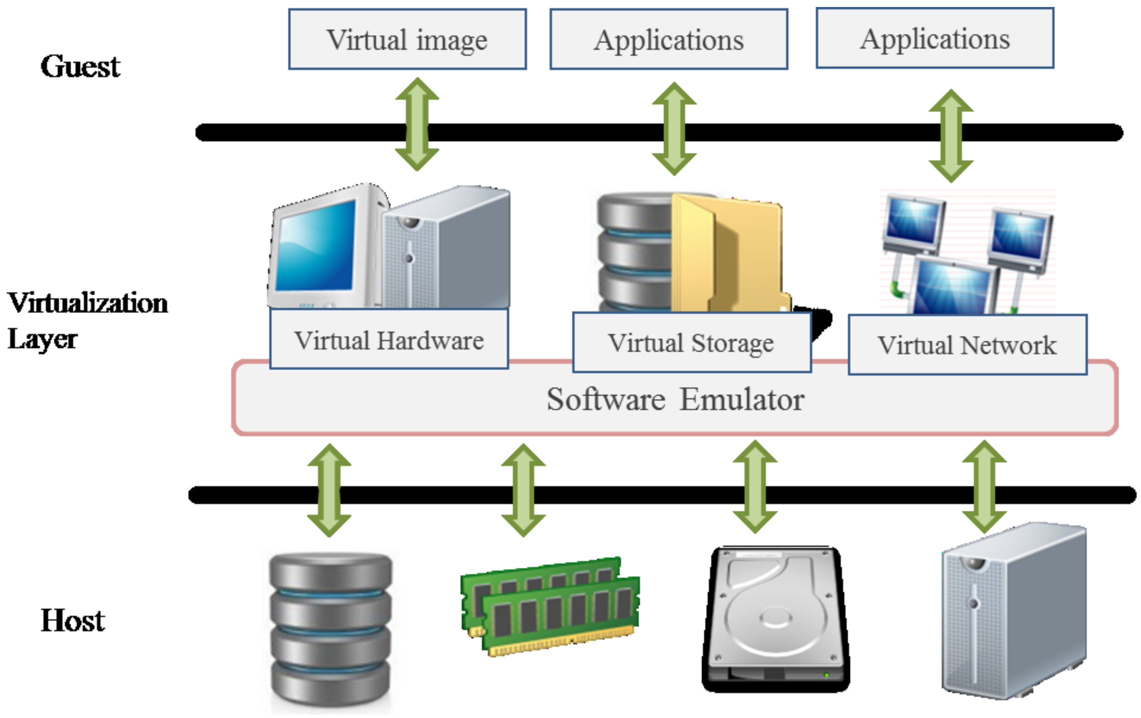

Figure 2.5: Virtualization

One of the main components of Virtualization is the hypervisor - also called a Virtual Machine Manager/Monitor (VMM). The hypervisor takes complete control of virtualized resources. Figure 2.6 shows the two types of the hypervisor. In the Type 1, the hypervisor runs on top of the hardware for monitoring the VMs which in turn run above the hypervisor. Whereas for Type 2, the hypervisor functions on top of an existing operating system environment and not directly on the hardware [10].

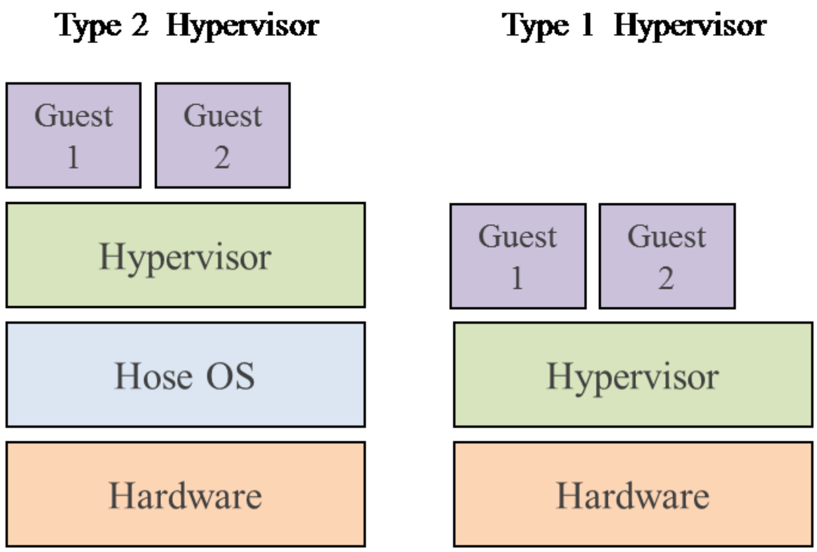

Figure 2.6: Different types of Hypervisor

We have two different levels of Virtualization: Fullvirtualization and Paravirtualization [11].

Fullvirtualization allows several guest operating systems to run on top of the hypervisor without modifications and they are completely isolated from each other. The hypervisor has the responsibility of handling all hardware requests then it must emulate hardware completely. Fullvirtualization brings several advantages such as full isolation, security and aligning various systems on the same platform.

Paravirtualization is a little different. A guest operating system needs modification in order to run in a specific environment and communicates with the hypervisor. Paravirtualization allows several guest operating systems to run simultaneously on hardware by providing more efficient use of system resources (memory, processors and storage) via effective resource sharing.

2.3.2 Benefits:

Virtualization has a several advantages as follows [11]:

Enhance resource utilization : By implementing the time-sharing, usage of hardware resources is increased. Resources can be shared and utilized in an efficient way because unused physical resources can used by another VM if needed. Moreover, when some physical resources are shared, the overall cost is low. In addition, IT administrators are able to provide and manage resources quickly.

Isolation: Multiple applications run on the same physical environment. Furthermore,

Virtualization allows for applications and operating systems to work independently from hardware which brings the benefit of software portability. Additionally, when two virtual machines are isolated from one other, communication is limited to processing which reduces the effects on each other.

Availability: The virtual instance can continue working even if sthe physical node needs to be shut down or if one virtual instance does not work for any reason, another instance can replace it directly without bringing down the system.

Scalability : The system has a capability to grow up based on its needs. Virtualization supports both types of scalability : scale up and scale out. Based on the load of the system, it may need to add more power to the virtual node or add more virtual nodes.

2.3.3 Virtualization Platforms

The following sub-section provides set of examples of Virtualization platforms.

Xen : Xen is an open source Virtualization platform which includes Type 1 hypervisor. It is at a paravirtualization level and needs modification to the guest operating system. Xen can dynamically provide resources for multiple VMs with different types of operating system [9,10]. One of the challenges in Paravirtualization is operating system modification but in a Xen minimal effort is needed in order to modify the operating system [12]. Figure 2.7 illustrates the Xen Architecture.

VMware : VMware is a well-known Virtualization platform in a Fullvirtualization group which does not need modification in a guest operating system and all Virtual Machines are completely isolated from each other. VMware has different products for Virtualization: VMware Workstation, VMware ESX Server and VMware Server.

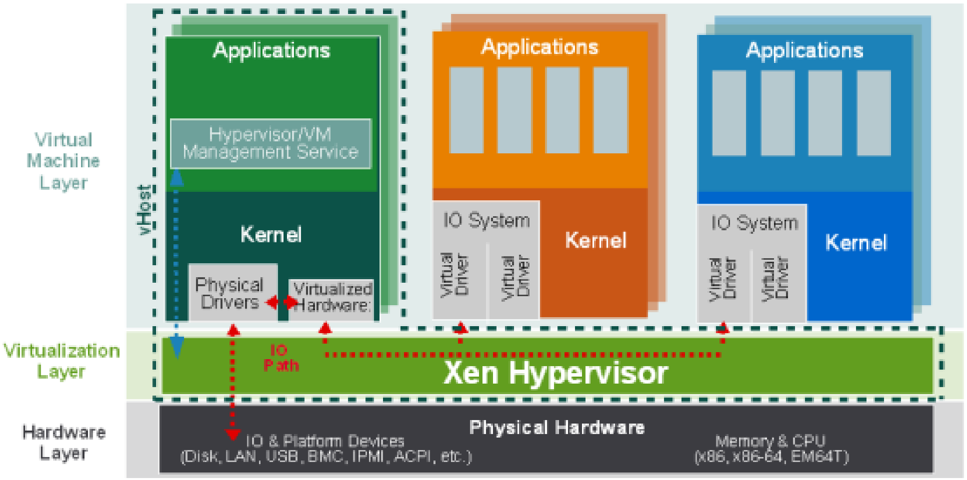

Figure 2.7: Xen Architecture

KVM: The Kernel-based Virtual Machine (KVM) is different from Xen or VMware. It is a Virtualization middleware for the Linux kernel.

2.3.4 Different Types of Virtualization

There are different types of Virtualization: Network Virtualization, Node Level Virtualization, Storage Virtualization, Desktop Virtualization and Database Virtualization [11].

2.3.4.1 Network Virtualization

Network Virtualization collects virtual hardware and virtual software to provide a virtual network. Several virtual networks can work on the same physical network. Network Virtualization is divided into two parts: Infrastructure Providers and Service Providers. The Infrastructure Provider has control on the physical infrastructure and the Service Provider collects resources and provides a virtual network [13].

Network Virtualization can be categorized into three groups. First, Virtual Local Area Network (VLAN) is a logic group of local area networks which appears as the same LAN but from a geographical point of view they are distributed. Secondly, Virtual Private Network (VPN) is a kind of virtual network which uses the internet to enable secure communication to the private network, the same as connecting directly to the private network.

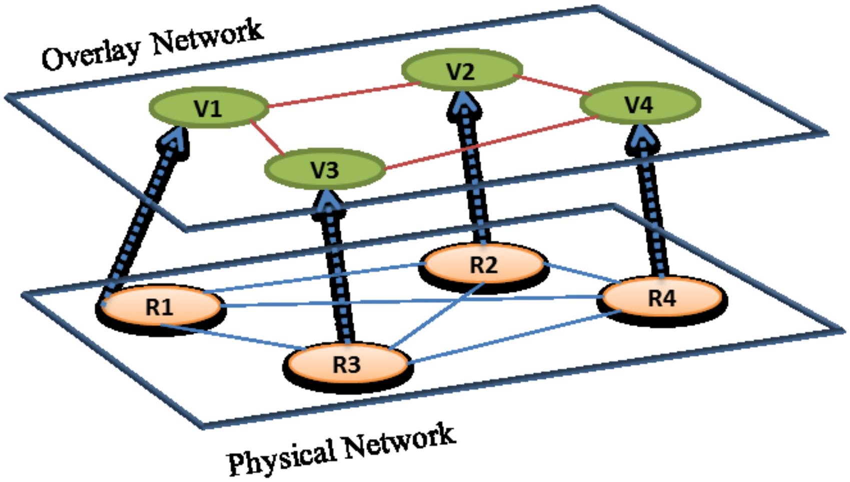

Figure 2.8: Overlay Network

Finally, the overlay network as shown in Figure 2.8 is a virtual network created on top of another network which does not need changes to the underlying network and entities may connect by a virtual or logical link.

2.3.4.2 Node Level Virtualization

Node Virtualization is at the level of physical resources partitioning and isolation. Hardware resources are partitioned and based on system requirements with each partition being allocated to a virtual node. The virtual node is a critical element in a virtual network, like a virtual router or a virtual switch [9]. On the other hand, virtual nodes help service providers to share sets of hardware resources and deploy separate customized protocols on them [10].

2.3.4.3 Storage Virtualization

Storage Virtualization is a logic pool of storage resources from different storage vendor products (e.g Disks, Storage Networks, Controllers) that working together to serve the needs of applications. There are three common storages: Direct Attached Storage (DAS), Network Attached Storage (NAS) and Storage Area Network (SAN). DAS is a storage device which directly connects to the network without any storage network. SAN is a network that interconnects various storage devices with data servers remotely in such a way that the devices appear as locally attached. NAS is a File-level computer data storage joined to a network providing data access to different clients. Each type of storage has its own benefits when virtualized. For example, virtualized SAN can reduce costs and hardware dramatically as well as improving maintenance performance.

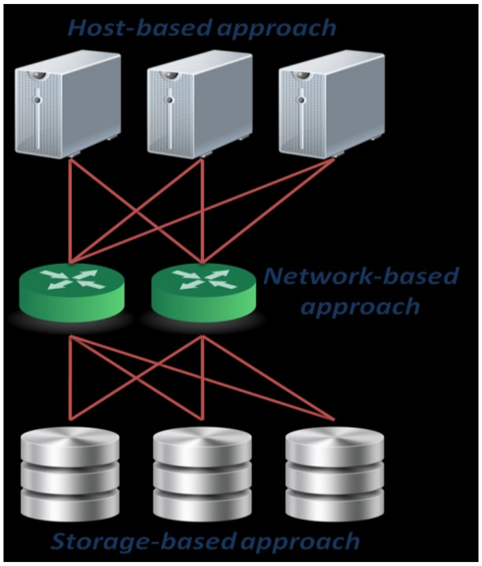

Figure 2.9: Implementation Approach

There are different approaches for implementing storage virtualization as shown in Figure 2.9:

A Host-based approach which is implemented as a software on host systems, a Network-based approach which is implemented on the network level and a Storage-based approach that is implemented in storage systems [11].

2.3.4.4 Desktop Virtualization

By decoupling physical machines from the desktop environment, Desktop Virtualization aims to create one or multiple virtual copies from the same desktop which are accessible from everywhere. This wide-ranging access is beneficial for organizations who have difficulty in controlling the desktop of clients and prefer all clients use same controllable desktop. Desktop Virtualization delivers several benefits such as cost-efficiency [11].

2.3.4.5 Database Virtualization

In general, databases can have various definitions because of different vendors providing their own database system definitions.

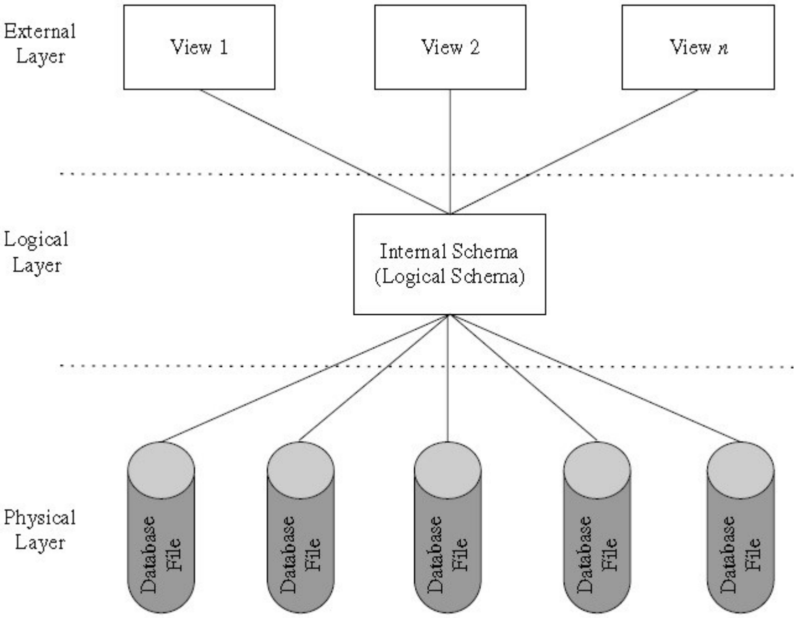

Figure 2.10: Database layers of abstraction

Usually, a database is a conglomeration of related data which is organized and operated as a single unit [16]. The Database Management System (DBMS) is a group of software which is designed for the user to create and manage a database [17]. The database is composed of three principle layers (see Figure 2.10): Physical Layer, Logical Layer and External Layer. The foundation of most DBMSs is based on these three layers 16].

Figure 2.11 illustrates the database structure [17]. The physical layer includes physical storages which maintain all the data in the database and play an important role in the Data Center and the Database. Additionally, there are two architectures for accessing the stored data: Shared-Disk and Shared-Nothing [18]. In Shared-Disk architecture multiple nodes share the same storage disk but in the Shared-Nothing architecture, none of the nodes share memory or disk storage.

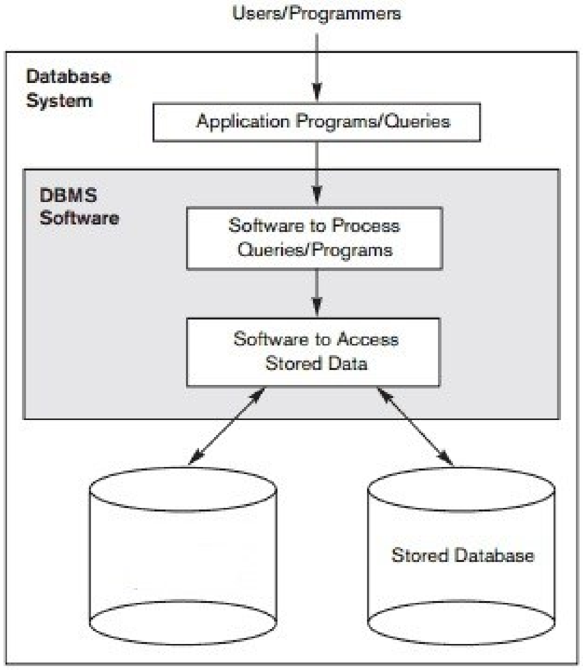

Figure 2.11: The Database Structure

Most of the databases follow the relational model which uses Structured Query Language (SQL). SQL is the standard language for database management systems. It supports atomicity, consistency, isolation and durability [17]. On the other hand, NoSQL (Not only SQL) is a good example of a non-relational database. NoSAQL is a new non-standard approach that supports only simple database transactions. NoSQL is a solution for the huge amount of the non-structured data [19]. Several researchers are interested in the Database Virtualization area because it plays a critical role in the information system and is one of the major resources in IT organization. Database Virtualization is a part of Resource Virtualization and will be used increasingly for improving managability and reducing costs.

Resource Virtualization (Database Virtualization) decouples applications from the resources (hardware or software) that are used by those applications and adds the layer to map virtual resources to real resources. The Virtualization Layer makes shared resources transparent for multiple applications which use them [21]. The layered approach of the database helps Virtualization to improve system utilization and performance [20,21] .

Additionally, the layered architecture of the database management system allows different classes of database Virtualization as shown in Figure 2.12 [20]. In the first class, database instance runs on the single Virtual Machine with dedicated Virtual resources. This is good in isolation but not in resource utilization. The second class of Virtualization is at the level of the database server Virtualization which can hold multiple databases. In the third class, databases are virtualized with each application having access to private tables in the virtual database. This class supports isolation, security and scalability. The fourth class includes a shared table database. When several tenants are sharing the same database schema this level of Virtualization can be useful .

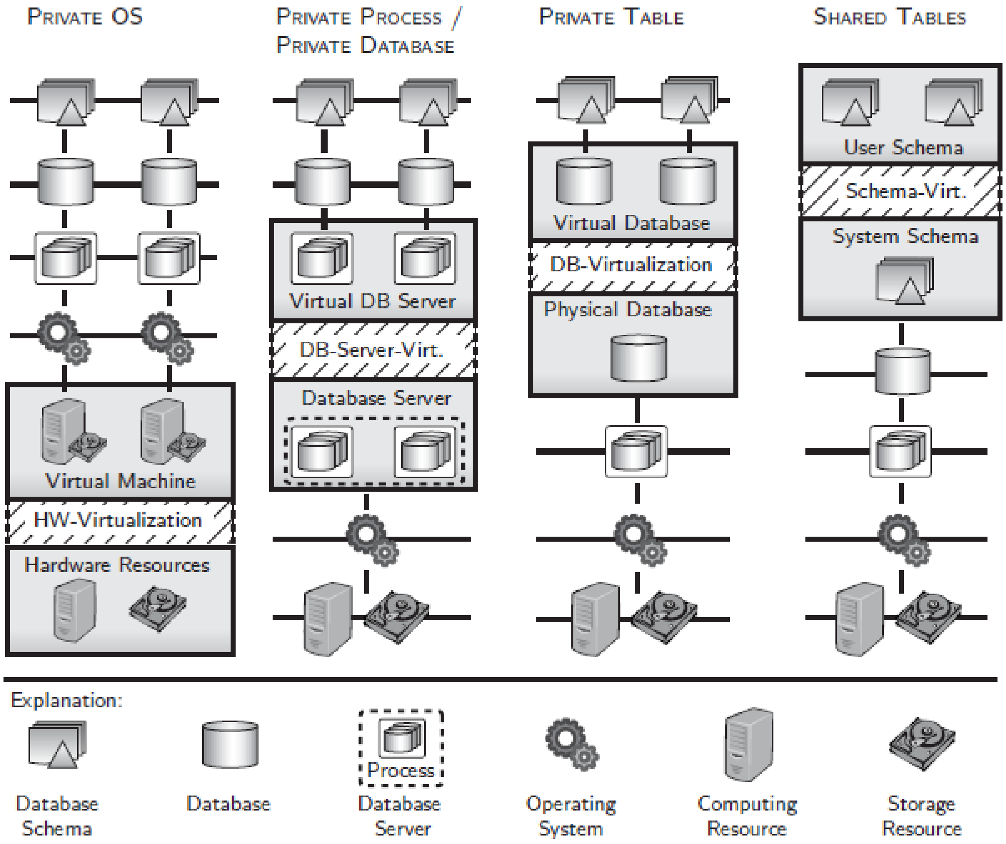

Figure 2.12: Different classes of database Virtualization

2.4 Chapter Summary

This chapter has successfully introduced the background information related to this thesis. We discussed the EPC definition and main components. We presented the definition of QoS and the key parameters. Then, we introduced three QoS service levels which are Best-effort service, Integrated service and Differentiated service. Then we explained QoS in EPC. In the last part, we introduced Virtualization and its benefits. We also explained the Virtualization platforms such as: Xen, VMware and we also discussed different types of Virtualization. In the next chapter we will propose a set of requirements for the QoS enabled video telephony which runs in an EPC environment with virtualized HSS. Afterward, we will discuss the state of the art most related to our research and evaluate them based on our requirements.

Chapter 3

3 Scenario, Requirements and State of the Art Evaluation

This chapter is divided into three sections. First, we present a set of scenarios that illustrate the use of the QoS enabled video telephony application with Virtualized HSS. Next, we propose a set of requirements for the entire system which include general and more specific requirements for QoS enabled video telephony application and Virtualized HSS. After that, we will review the state of art works related to the thesis research and evaluate them based on our requirements. Finally, we will present the summary at the end of this chapter.

3.1 Illustrative Scenarios

In this sub-section, we provide a scenario which illustrates how the differentiated QoS and Virtualized HSS is used in the video telephony application over an EPC network. We assume that there are two types of providers: the video telephony provider and the EPC provider. In our scenarios we consider three video telephony service providers (which do not have their own HSS) on top of the shared EPC network. Another assumption is that the EPC service provider offers the Virtualized HSS to other service providers.

We also assume that the end-users can select between the following classes of QoS profiles: Bronze, Silver and Gold. This selection can be made prior to session initiation, at session initiation or even during sessions.

3.1.1 The Scenario

Telecommunications is a highly competitive market in which several video telephony service providers are in a constant battle to maintain and grow their client base. In this scenario we will look at three service providers who each offer a different class of service. Service Provider A is a normal-sized company which can offer Bronze and Silver class which is affordable and perhaps aimed predominantly towards students. Service Provider B is an average-sized company and can offer only Bronze classes which is suitable for home users.

Service Provider C offers Gold and Silver classes and can guarantee the highest quality for company and business users. In the everyday functioning of a business video calls play an important role.

Because these three providers are using the shared the EPC at the same time then it is possible that a huge number of users connect and want to begin a call. Each service provider is battling to try to attract more users and make more money while also trying to maintain accessibility 99.9% of the time. Each service provider wants to offer the best possible service to its end users. During normal business hours a service provider receives a much larger number of requests than at midnight. Therefore growing and contracting of data usage becomes important.

Let us for example look at an ongoing video session between two Bronze end-users using Service Provider A. During the session, a Silver end-user from Service Provider A wants to initiate a voice session. First, the video telephony provider needs to have user profiles for establishing the session then they need to communicate with the Virtualized HSS to retrieve the information. However, the network does not have sufficient resources to support the two sessions. The system will downgrade the video communication between the two Bronze end-users to audio in order to enable the voice session initiated by the Silver end-user.

In a scenario involving Service Provider B, there is a video session between two Bronze end-users during which another Bronze end-user wants to initiate a voice session. The provider has to check the user profile to ensure that the user is eligible for establishing the session. For this purpose, it needs to communicate with its own Virtualized HSS. Let us assume the network does not have enough resources to support the two communications simultaneously. The system will reject the demand of the Bronze end-user.

A third simultaneous scenario involves a video session between two Silver end-users of Service Provider A. During this video session two Gold end-users of Service Provider C want to initiate a video session. For the session to be initiated the Virtualized HSS must be checked by Provider C. However, the network resources are limited. Therefore, the system will only allow voice for the session initiated by the Silver end-users in order to preserve the continuity of the video session between the Gold end-users.

3.2 The Requirements

As a result of the motivation scenario, a set of requirements can be defined. This section contributes three sets of requirements: general requirements, requirements for the QoS enabled video telephony application and the requirements of HSS Virtualization.

3.2.1 General Requirements

The first general requirement is that the architecture should have an overall standardized approach which supports popular standard interfaces and application programming interfaces (APIs). These APIs enable a rapid and cross-platform development of applications.

The second general requirement is that the architecture should re-use the subjacent network with as little change as possible. Imposing changes to the underlying network can only slow down the deployment and should be avoided.

3.2.2 The QoS Enabled Video Telephony Application Requirements

The first requirement is that the architecture should support standard video telephony protocols, since a plethora of video telephony protocols already exists. Second, it should permit session creation, modification and deletion.

Furthermore, the proposed architecture should support differentiated QoS of the video telephony service by giving end-users the opportunity to subscribe to given classes of service (e.g. Gold, Silver and

Bronze) and have their sessions served with the corresponding parameters. End-users should also be able to change their subscription class, even during sessions.

Yet another requirement is that sessions of end-users with lower priorities should preferably be downgraded instead of being terminated when there is a resource problem in the network.

3.2.3 HSS Virtualization Requirements

The first requirement in our proposed architecture is that the Virtualized HSS should able to support all functionality that is currently offered by the non-Virtualized HSS. Because the video telephony application in the Virtualized HSS is a scenario built based on working with real HSS, then Virtualized HSS should work exactly the same as a non-virtualized version with the same functionality.

The second requirement is the Virtualized HSS should be able to interact with other components of the video telephony network. This requirement is motivated by the Virtualized HSS scenario because some components of the video telephony service provider need to interact with Virtualized HSS, so it should be discoverable by other components of the video telephony network. The third requirement is each video telephony service provider can only have access to its own users’ data for security issues. In the Virtualized HSS scenario, The users of Provider X should be isolated from the users of Provider Y.

The fourth requirement is the system should scale in terms of increasing the number of end-users and service providers. The system must be able to accommodate larger loads just by adding resources. This could be done by either making the hardware stronger (scale up) or adding virtual nodes (scale out) when a new service provider is added to the shared EPC or by increasing the number of users in each service provider.

The fifth requirement is the system should support elasticity. The architecture needs to have sufficient ability to respond to load fluctuations on the video telephony service provider due to demand surges or troughs from the end-users' requests. The architecture must have the ability to be scaled-up by adding more virtual nodes or scaled-down by removing virtual nodes without any interruption when user numbers rise or fall.

It is important that the proposed architecture supports Telco Grade requirements. The first requirement is the architecture should support high availability and reliability. The architecture should be able to remain available and continue working even in the face of faults or failure of individual components. Additionally, in the case of failure, the architecture should be able to detect the failure and support failure recovery.

The second requirement is that the architecture supports scalability. Moreover, components of the architecture should grow independently from each other. For example, if the architecture needs more storage for the database, it is able to grow.

3.3 The State of the Art Review

Most of the current video telephony application solutions implemented are either Internet or 3G networks. However, based on our knowledge, none of these systems support differentiated QoS for different end-users. The Internet does not provide QoS because it is a best effort network and 3G telecommunication networks already offer guaranteed QoS to some extent, but they do not provide the differentiated QoS. On the other hand, Virtualization is a new approach in telecommunication networks and none of the current EPC networks or other telecommunication networks offer Virtualized HSS to the service provider who works on top of them.

In this section we organize the state of the art into two groups: The first group reviews the research works which relate to the guaranteed and differentiation QoS in video telephony applications. The second group deals with research works which relate to HSS Virtualization. Finally, we will evaluate these works based on our requirements.

3.3.1 Video telephony application

The most well-known differentiated QoS scheme is certainly the IETF differentiated QoS architecture [22]. They proposed an architecture for Differentiated Services on the Internet. Packets are prioritized and forwarded at each router depending on their priority level. This architecture, however, is coarse-grained. Packets are prioritized depending on the traffic type (e.g. voice vs. video). While it enables the prioritization of voice vs. video in the same session, it is not refined enough to enable prioritization between sessions.

They don’t discuss differentiated QoS and prioritization based on different classes of service. There is no discussion about the ability of end-users to change their subscription class, even during sessions.

A few references discuss the QoS aspects of video telephony in 4G. However, none of them deal with prioritization between video telephony sessions. Reference [23] for instance, considers video telephony over downlink LTE links. They focus on performance in both situations whereby QoS is or is not provided. They apply two algorithms on the simulated environment which are: Proportional Fair Scheduler (PF) and Proportional Fair Scheduler with Minimum/Maximum Rate constraints (PFMR). The PF is a famous scheduler algorithm for adjudicating between the cell throughput and fairness among users. The PF assigns the same resources to all of the users at the same time, thus achieving long-term justice. The PFMR is proposed for some applications like video streaming to provide a specified minimum serving rate. The difference between PF and PFMR is that PF does not provide QoS for video telephony but PFMR does. The simulation result shows that it is beneficial to provide QoS and PFMR can increase the capacity more than PF.

They do not propose a new architecture for video telephony application. Furthermore, they don’t discuss differentiated QoS and prioritization based on different classes of service. There is no discussion about the ability of end-users to change their subscription class, even during sessions.

There are studies of the QoS aspects of video telephony in wireless environments in general. Reference [24] presents the IP Traffic Symmetry Ratio (IPSR) which is an attribute for real-time data performance. Based on the IPSR, they propose a scheduler to provide QoS for video IP telephony in wireless environments. They call the scheduler TI-MAC (Traffic Intelligent - MAC). TI-MAC is a round robin scheduler with minimal overhead, but differentiating between different sessions is not considered. For proofing the concept of the TI-MAC algorithm, they compare this algorithm with the IEEE 802.11e EDCA algorithm on the same hardware environment.

However, they also don’t discuss differentiated QoS and prioritization based on different classes of service. There is no discussion about the ability of end-users to change their subscription class, even during sessions.

Several differentiation schemes have been proposed for 3G networks. However, as shown in reference [25], they proposed a new solution which allows session level differentiation in an IP Multimedia System (IMS) -based 3G network. The proposed solution has two main parts : a new call differentiation scheme and an extension of the IMS architecture. The extension is for enabling the proposed scheme. Figure 3.1 depicts the proposed architecture which includes two new components : the Session Prioritization Function (SPF) and the Context Information Base (CIB). The CIB manages the contextual information for SPF. The SPF is responsible for making the decision to allocate and re-allocate resources based on the CIB information and the session’s QoS profiles. Additionally, they enhance some of the existing IMS components, for example the QoS negotiation capability added to UE.

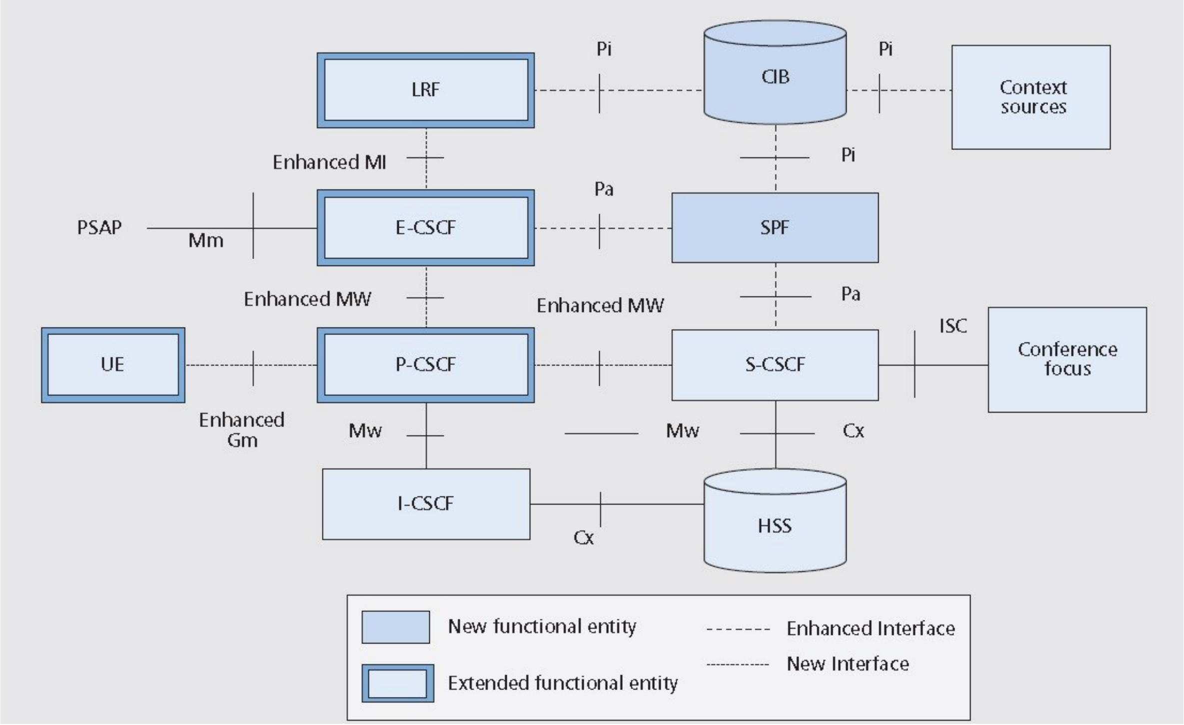

Figure 3.1: The Extended IMS Architecture

The proposed architecture supports differentiated QoS and prioritization based on different classes of service. Based on one of our requirements, the architecture should reuse the subjacent network with as little change as possible. Imposing changes to the underlying network can only slow down the deployment and should be avoided. They propose heavy extensions to the IMS portion of a 3G network, and it does not meet our requirements.

According to [26], an architecture for a differentiated QoS scheme for a mobile video surveillance application is proposed over the EPC network. This architecture supports guaranteed QoS that isn’t supported by the Internet and also supports the differentiated QoS that isn't supported by 3G networks. Figure 3.2 shows the proposed architecture and its main components. The two main components of the proposed architecture are: the Service Development Platform (SDP) and the Machine to Machine (M2M) gateway.

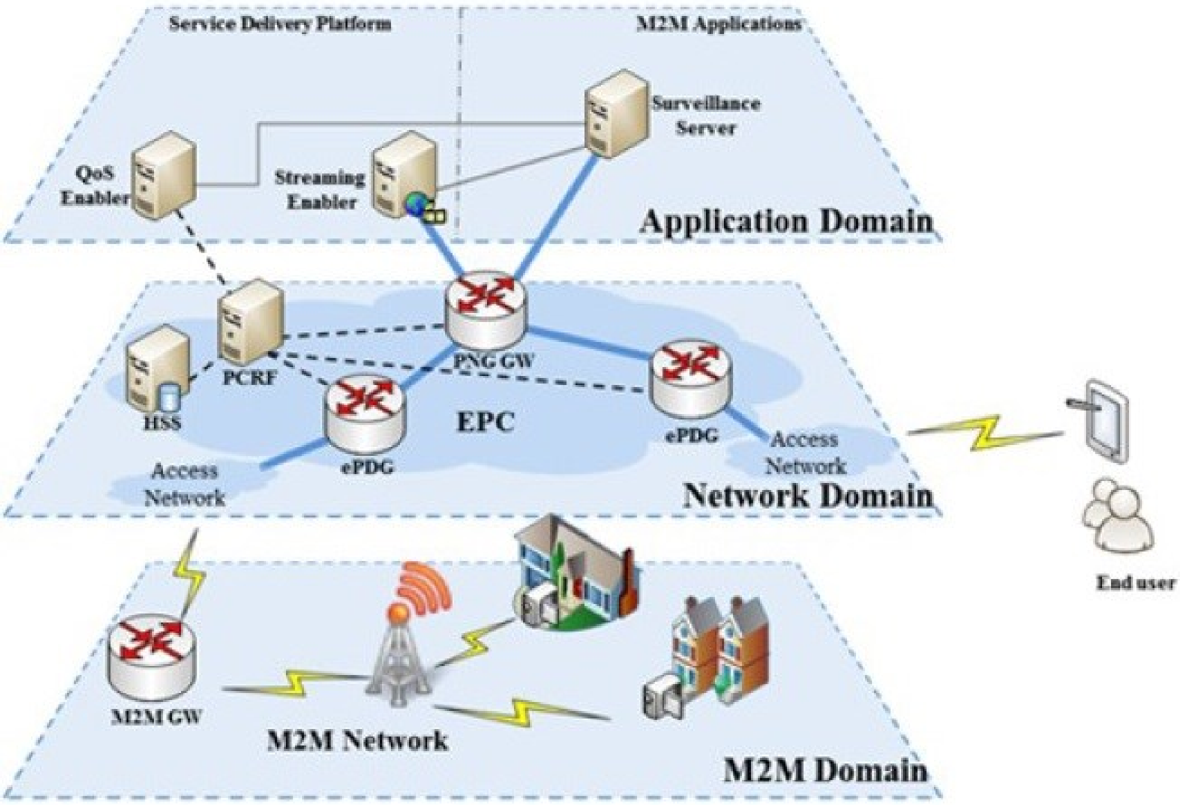

Figure 3.2: The Overall Architecture

The SDP consists of the QoS enabler and the streaming enabler. The QoS enabler's role is to provide differentiated QoS. It stands between the service provider and the EPC and enables the differentiation QoS feature for service providers. This architecture provides a differentiated QoS scheme for mobile video surveillance applications. Video surveillance applications are streaming applications and are fundamentally different from telephony applications which are real-time bilateral media exchange applications.

In [27], they proposed a refined QoS mapping algorithm from UMTS (access network) to IP as shown in Figure 3.3b. This article focuses on voice and video telephony services in UMTS. They proposed a new algorithm to make a differentiation between voice and video.

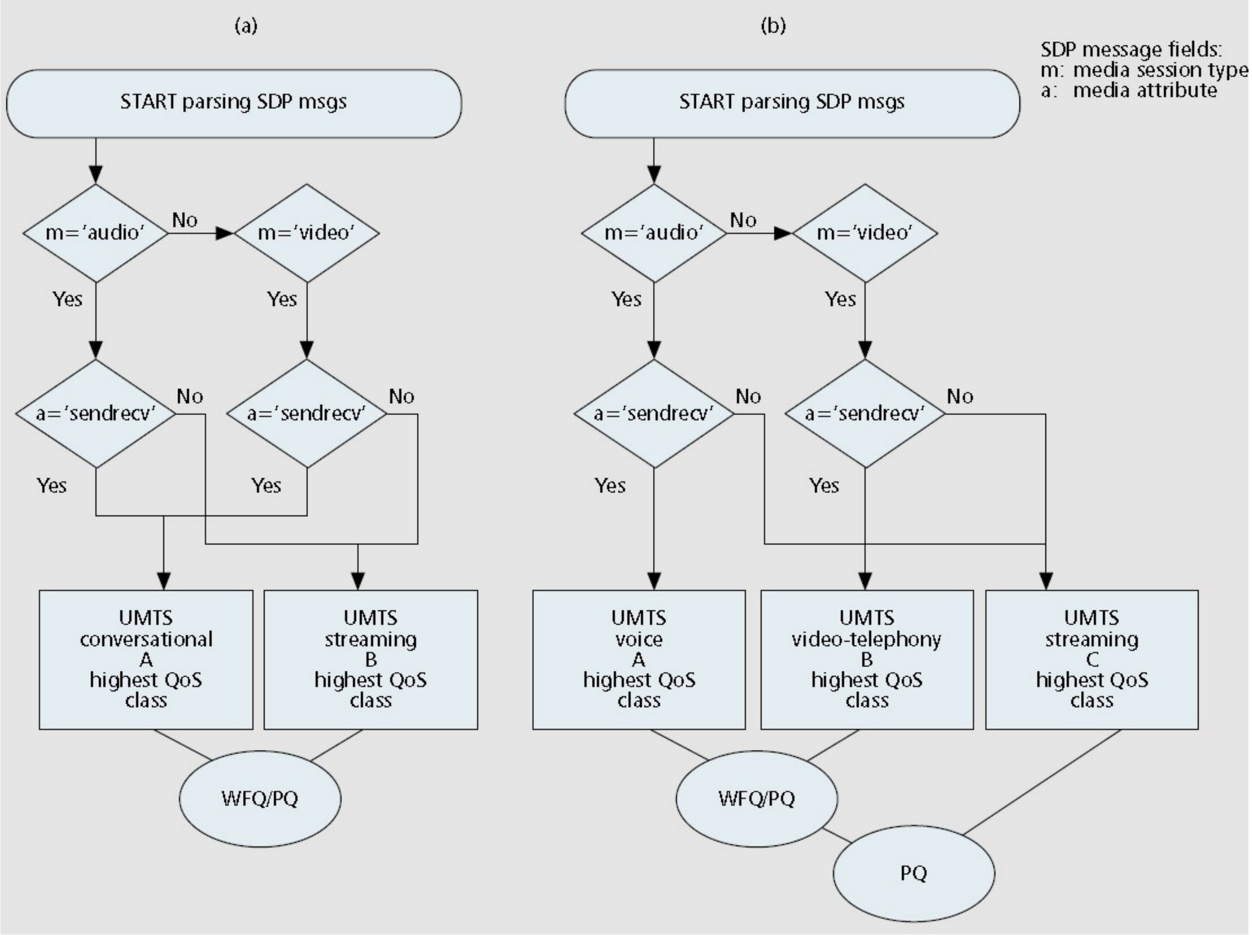

Figure 3.3: The Algorithm

Figure 3.3 has two parts, part (a) includes standard algorithm which does not consist of differentiation between voice and video and all the users use the same class of service. In part (b), the differentiation between voice and video is applied.

However, they don’t discuss differentiated QoS and prioritization based on different classes of service. There is no discussion about the ability of end-users to change their subscription class, even during sessions.

3.3.2 HSS Virtualization

In [28] a new cloud base architecture for HSS is proposed which focuses on improving the performance of data storage which affects the performance of the IMS system. The architecture is composed of two layers : the resource pool layer and the management layer. The resource pool layer has the physical resources and includes several groups of these resources. Each group contains resources which have a similar structure, for example a computing resource pool and data resource pool. The management layer has a responsibility to manage resources, tasks and users.

Figure 3.4 depicts the proposed architecture which uses the benefits of cloud storage to improve performance.

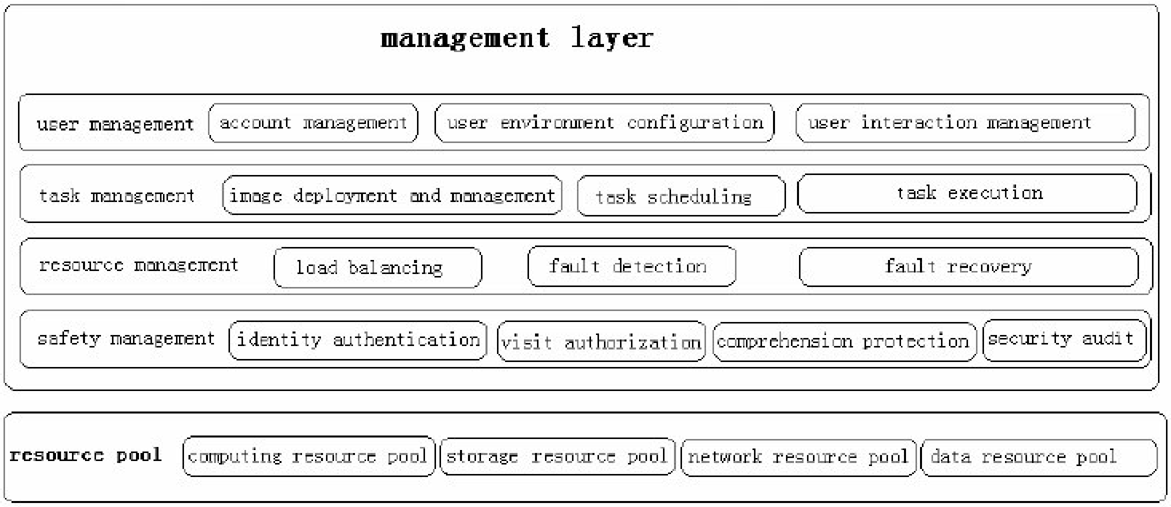

Figure 3.4: Improved HSS Architecture

The proposed architecture does not discuss how the new architecture interacts with other components and also doesn’t discuss high availability. The system tries to use the cloud storage based approach but they don’t support scalability or elasticity. They did not use Virtualized HSS, they just tried to improve the architecture .

In other research work [29,30], a distributed MME for next-generation LTE cellular networks is proposed. In this proposed architecture, the processing of UE control plane messages and storage of UE state are separated. The task of processing control plane is divided among the several numbers of servers (also called Replica). Figure 3.5 shows the proposed architecture. There are two main components in this architecture : Reliable Object Store (ROS) and Message Processor (MP). ROS maintains a current snapshot of all users' state and all the methods that are required by MME are implemented in the MP. Benefits of a distributed implementation of the MME are locality, reliability, elasticity and cost-reduction.

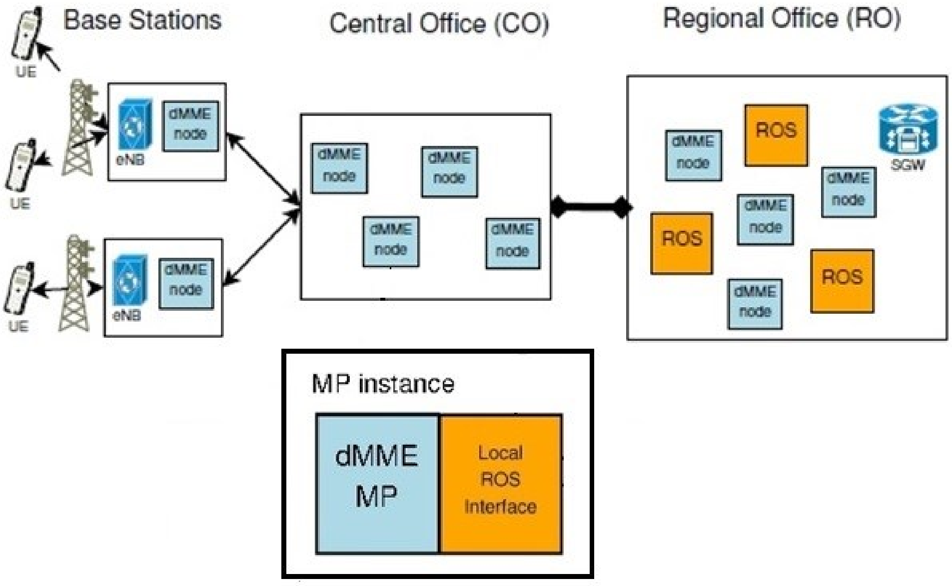

Figure 3.5: The DMME Architecture Components

Because the architecture decouples MME into two parts, it can support scalability and elasticity. However, the proposed architecture does not support all functionality of current Non-Virtualized HSS, for instance Diameter interfaces are not supported. Supporting isolation between different providers is not discussed. The architecture is not Standards-compliant, due to the fact they mention they don't fully support all 3GPP standards and there is no standard interface between DMME and ROS.

According to [31], an Elastic TranSactional relational (ElasTraS) database is proposed which is scalable and elastic. The ElasTraS is a lightweight Distributed Data Store for the cloud. As shown in Figure 3.6, the architecture includes four main components : Distributed Fault-tolerant Storage (DFS), Owning Transaction Managers (OTM), TM Master and Metadata Manager (MM). DFS is a fault-tolerant storage which is similar to a Hadoop Distributed File System (HDFS). OTM is responsible for the execution of transactions. TM Master is responsible for assigning partitions to OTMs, partition reassignment for load balancing and elasticity and detecting and recovering from OTM failures. Metadata Manager (MM) is responsible for stores system state, the mapping of partitions to OTMs and leasing information for the OTMs.

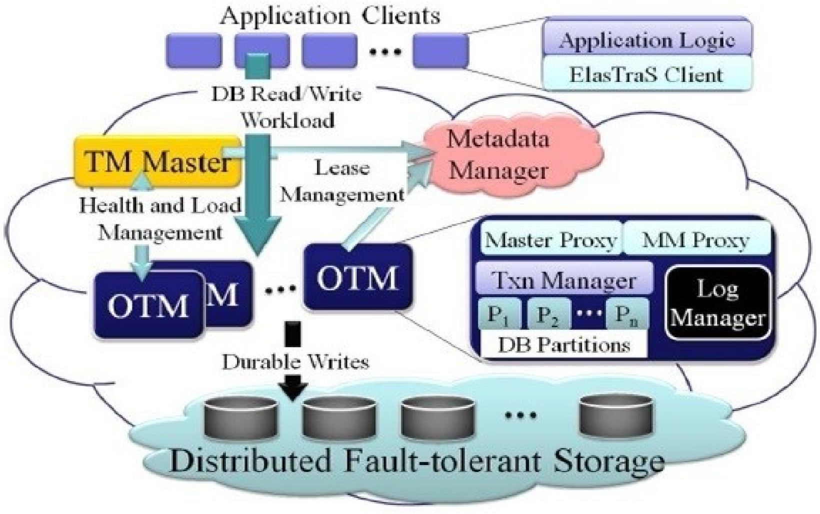

Figure 3.6: The ElasTraS Architecture Overview

The proposed architecture covers both big and small databases in the cloud. Database partitioning plays the main role in this proposed architecture. They partition the database and this allows for scalability and high availability by limiting transactions to a single partition. This architecture offers some of the functionalities that are supported by Non-Virtualized HSS (e.g. Storage, Database computation) but they do not offer others (e.g. support of Diameter). The architecture does not support standard approach for database computation. It follows shared-nothing approach in which scalability is problematic because partitioning is required for scaling out. Partitioning is therefore impossible or complex when we have hundreds of thousands of tables that are all interrelated. High availability is also problematic. A master-slave approach is needed and updating two databases can be complex. This architecture does not support multi-tenancy, meaning there is no isolation between data of two MVNOs. This architecture supports elasticity and efficient load balancing.

In [32], a shared-data clustered database is proposed. In this proposed architecture, database computing and storage are separated which enables the system to scale independently, allowing both functions to be virtualized. The shared-data architecture enables multiple nodes to share the same physical data. Figure 3.7 shows the architecture which includes three main parts. Each node has the ScaleDB engine that works with a DBMS. The ScaleDB Cluster Manager orchestrates the interaction of nodes. The Cache Accelerator Server (CAS) is designed to share the data.

This architecture offers some of the functionalities that are supported by Non-Virtualized HSS (e.g. storage and database computing) but they don't offer the functionality of the Diameter interface. This architecture supports isolation, high availability, scalability and elasticity. A standard approach for database computation (both SQL and NoSQL) is also supported. Because this architecture meets some of our of our requirements, as mentioned above, we will reuse it as one of the building blocks of our proposed architecture.

According to [33], User Data Convergence (UDC) supports a layered approach, separating the user data storage from the application logic which UDC can apply to HSS. It is possible for a network to have more than one UDC. As seen in Figure 3.8, the UDC functional entities are User Data Repository (UDR), Front End (FE) and one reference point (Ud). The FE executes the application logic and the Ud is a reference between UDR and Front ends.

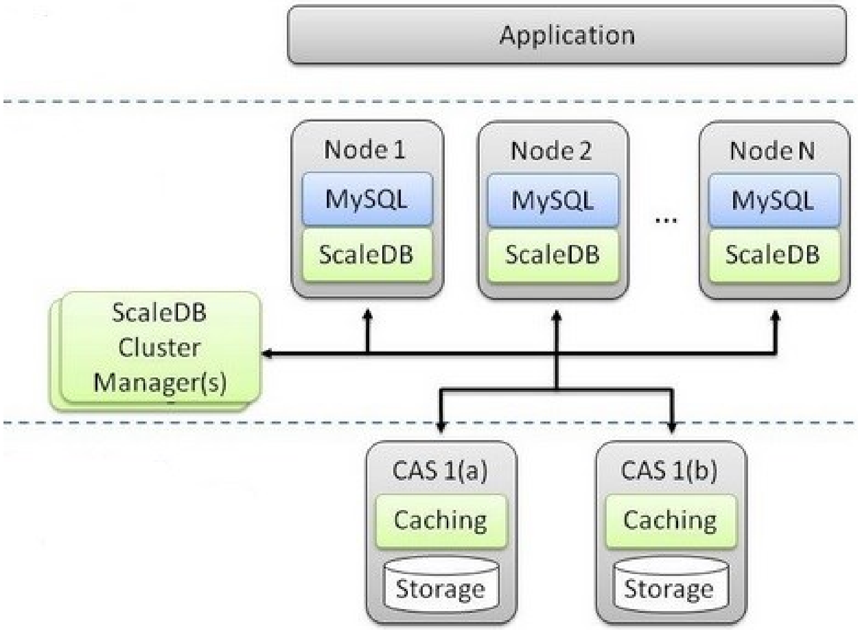

Figure 3.7: The Scaledb Architecture

UDR is a single logical repository from a FE perspective which can only access relevant data. UDR can be distributed over different locations or centralized and UDR's interaction with other entities is through existing 3GPP reference points.

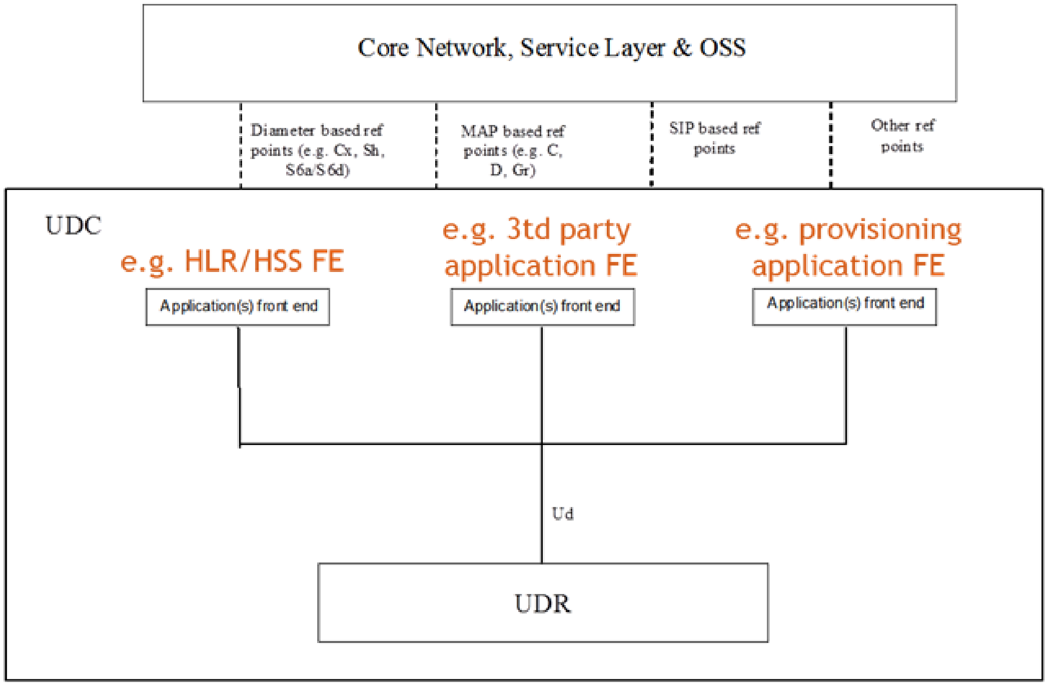

Figure 3.8: The User Data Convergence Architecture

When evaluating the architecture, the following results are evident:

- The architecture supports all functionality that is currently offered by the Non-Virtualized HSS.
- The architecture is able to interact with other components of the network.
- The architecture supports isolation between MVNOs.
- The architecture is Standards-compliant.
- This architecture does not support elasticity and scalability because of its decoupling into two layers. Diameter function cannot therefore be scaled independently.

Their idea of layering the architecture and splitting the functionalities (as shown in Figure 3.8) will be reused in our proposed architecture. However, we will split the functionality in a finer grained manner, in order to enable scalability.

In [34], the article studies the state of the arts in the virtual network interfaces research area. Figure 3.9 is an example of network interface Virtualization .

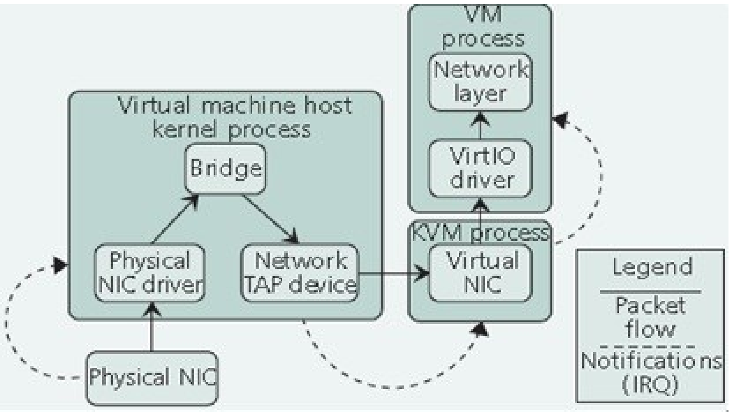

Figure 3.9: The Example of Network Interface Virtualization

They classify network interface Virtualization into two categories: software-based virtual interfaces and hardware-assisted interfaces. The software-based virtual interfaces are divided into two groups : First group, “Emulated”, which emulates the interface and a hypervisor must use hardware virtualization extensions to trap each attempt to access the device. In the second group “Paravirtualized”, the VM can have more control over when a VM exit is created due to an interface request. It can bundle several packets to be sent into a single VM exit and can reduce the number of context switches between the hypervisor and the VM. The hardware-assisted interfaces are industry-based standard and they create devices that help to significantly decrease virtual NIC overhead.

The architectures do not support all functionality of current Non-Virtualized HSS, for example database functionality. They do not discuss isolation, high availability and load balancing.

3.3.3 Evaluation summary

Table 3.1 summarizes our evaluation of the state of art relevant to the thesis research based on our requirements. We can see that none of them fully satisfy our requirements.

Related Work Requirements	A New Architecture of HSS Based on Cloud	DMME: Virtualizing LTE Mobility Management	Elastras: An elastic , Scalable, And Self Managing Transactional Database in the Cloud	ScaleDB	User Data Convergence (UDC)	Network Interface Virtualization: Challenges and Solutions	An Architecture For Differentiated Services, IETF, RFC	Video Telephony over Downlink LTE with/without QoS provisioning	A Novel Optimized Scheduler to Provide QoS for Video IP Telephony over Wireless Networks	Control Level Call Differentiation in IMS-Based 3G Core Networks	A 3GPP 4G Evolved Packet Core Based – Architecture for QoS Enabled	UMTS-to-IP QoS Mapping for Voice and Video Telephony Services
Standard video telephony protocols							Not Satisfied	Satisfied	Satisfied	Satisfied	Not Satisfied	Satisfied
session creation, modification and deletion							Not Satisfied	Satisfied	Satisfied	Satisfied	Satisfied	Satisfied
Differentiated QoS							Partially satisfied	Partially satisfied	Partially satisfied	Satisfied	Satisfied	Partially satisfied
Able to change their subscription class							Not Satisfied	Not Satisfied	Not Satisfied	Not Satisfied	Satisfied	Not Satisfied
Re-use the sub-jacent network							Satisfied	Not Discussed	Not Discussed	Not Satisfied	Satisfied	Not Discussed
support all functionality	Satisfied	Partially satisfied	Partially satisfied	Partially satisfied	Satisfied	Partially satisfied						
Scalability	Not Discussed	Satisfied	Satisfied	Satisfied	Not Satisfied	Not Satisfied						
High Availability	Not Discussed	Satisfied	Satisfied	Satisfied	Not Satisfied	Not Discussed						
Elasticity	Not Discussed	Satisfied	Satisfied	Satisfied	Not Satisfied	Not Satisfied						
Isolation	Not Discussed	Not Discussed	Not Satisfied	Satisfied	Satisfied	Not Discussed						
Standards-compliance	Not Discussed	Not Satisfied	Not Satisfied	Satisfied	Satisfied	Not Discussed						
Efficient load balancing	Not Discussed	Satisfied	Satisfied	Satisfied	Not Discussed	Not Discussed						
be able to interact	Not Discussed	Not Discussed	Not Discussed	Not Discussed	Satisfied	Not Satisfied						

Table 3-1: Summarizes evaluation of the state of art

3.4 Chapter Summary

In this chapter we presented scenarios that illustrate the need of differentiated QoS in video telephony and Virtualization HSS. Then, we derived requirements from these scenarios. We categorized these requirements into two groups: requirements for the QoS enabled Video telephony application and the requirements of HSS Virtualization. We categorized the state of the art into two groups: the QoS enabled Video telephony application and HSS Virtualization. We evaluated the state of the art using the requirements. We concluded that none of the works presented in the state of the art meet all the requirements we identified.

Chapter 4

4 Proposed Architecture

zan EPC environment with Virtualized HSS. This architecture is based on the requirements discussed in the previous chapter. This chapter is organized into six sections. First, we present a general overview architecture of the QoS Enabled Video Telephony with Virtualized HSS which includes the functional entities and the communication interfaces. Second, we present the new architecture for the Video Telephony Service and the system's main procedures. Third, we present the new preliminary architecture for HSS Virtualization which includes the functional entities, the communication interfaces and the elasticity procedure. Next we summarize how the requirements are met by the architecture. Finally, we present an illustrative scenario that shows how entities in the system architecture interact with each other.

4.1 General Architecture

In this section, we propose a novel architecture for QoS Enabled Video Telephony in a 4G EPC environment with a Virtualized HSS (VHSS). Figure 4.1 depicts the overall architecture. It includes the Video Telephony Service that uses the VHSS, the Differentiated QoS Service Delivery Platform, the VHSS Enabler and the other EPC components like PCRF.

We propose a new voice and video telephony service that is based on the multimedia telephony service standard. It introduces specific features to the end user such as prioritization and a different type of class of services. Furthermore, the proposed architecture supports differentiated QoS of the Video Telephony Service by giving end-users the opportunity to subscribe to given classes of service (e.g. Gold, Silver and Bronze) and have their sessions served with the corresponding parameters. We will explain more about the Video Telephony Service in Section 4.2.

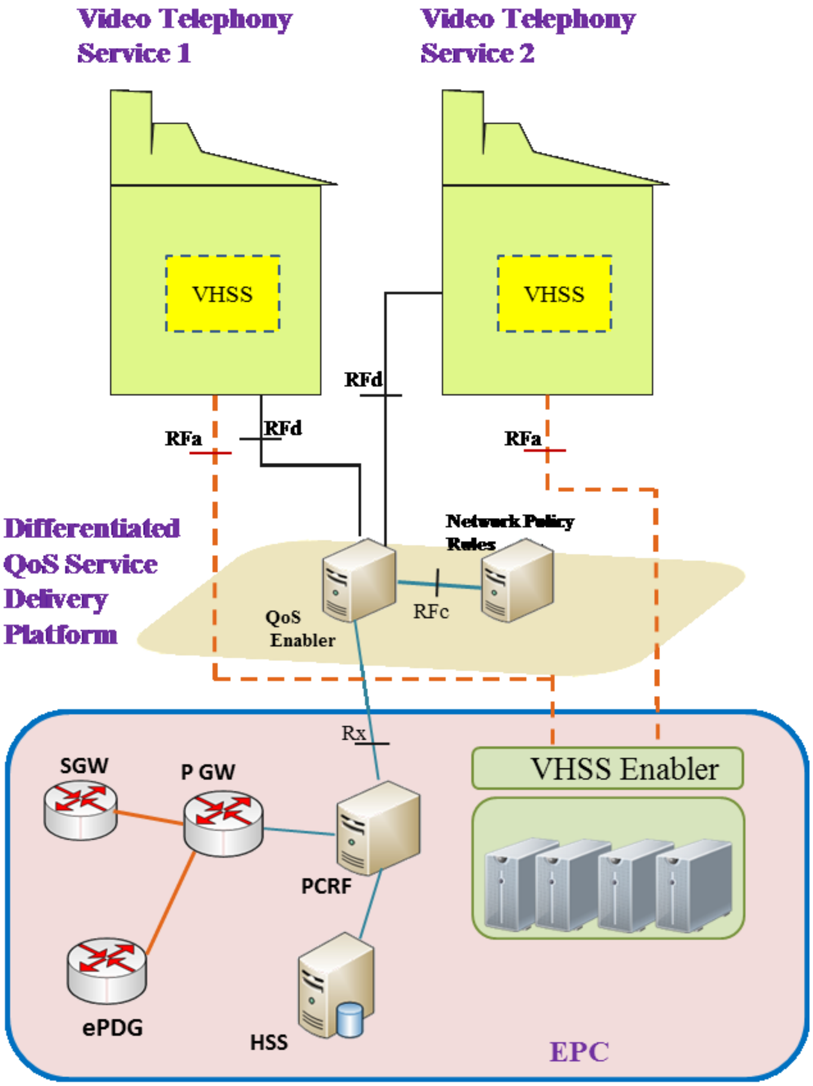

Figure 4.1: The Overall Architecture

The Differentiated QoS Service Delivery Platform is made up of two components: The QoS Enabler and the Network Policy Rules. The QoS Enabler communicates with the PCRF to establish or update sessions with the desired QoS profiles and reserves network resources. The QoS Enabler exposes the EPC network capabilities to Video Telephony Services in order to provide differentiated QoS. If not enough resources are available, the QoS Enabler consults the Network Policy Rules to identify the service provider whose sessions could be downgraded/terminated to liberate the required resources. The Network Policy Rules prioritize the Video Telephony Service providers based on their service level agreement. After the consultation, the Network Policy Rules has a responsibility to select the service provider which satisfies a number of specific criteria. For example, the selected Video Telephony Service provider should have a lower priority and a lower QoS profile based on its service level agreement.

The VHSS Enabler proposed in our architecture is an integral part of the EPC network. It provides the required infrastructure to enable an EPC provider to offer VHSS to the Video Telephony Service providers. The VHSS Enabler allows an EPC provider to offer flexible and scalable HSS to these providers. It is the VHSS Enabler that creates the virtual instance of the HSS, the details of which will be discussed further in Section 4.3.

4.1.1 Communication Interfaces

The interfaces between the Video Telephony Service and the VHSS enabler (i.e RFa) and the QoS Enabler (i.e RFd) are REpresentational State Transfer (REST) -based, as are the interfaces between the QoS Enabler and the Network Policy Rules (i.e. RFc). We selected REST because it is standard-based, lightweight and can support multiple data representations (e.g. Plain text, JSON and XML). REST is a client-server architectural style for loosely coupled distributed systems. It models the information to act on as resources, and identifies each resource using a uniform resource identifier (URI). The resources are then accessed via a uniform interface. REST is not bound to a specific communication protocol, but is mostly used with HTTP. The uniform interface is mainly composed of the following HTTP methods:

- **POST:** It is used to create a new resource on the server.
- **PUT:** It is used to modify a resource on the server.
- **GET:** It is used to retrieve a representation of a resource.
- **DELETE:** It is used to delete an existing resource.

There is no one-to-one mapping between the methods and the operations [35]. Table 4-1 is an example of the REST resources in our architecture. It summarizes the main resources we defined for the QoS Enabler.

Resources	Operation	HTTP Method: resource URI
List of sessions	Create: establish a new session	POST: /Services/Sessions
	Read: get information (E.g. QoS profile, download bandwidth) about a specific session	GET: /Services/Sessions/{SessionID}
A specific session	Update: modify a specific session (e.g. QoS profile, download bandwidth)	PUT: /Services/Sessions/{SessionID}
	Delete: end a specific session	DELETE: /Services/Sessions/{SessionID}

Table 4-1: Main QoS Enabler Resources

The communication between the QoS Enabler and the PCRF network (i.e. Rx) is ensured via a Diameter interface. Diameter protocol is an Authentication, Authorization, and Accounting (AAA) protocol used in new generation networks and intended for applications such as network access and mobility in IP networks. The 4G PCRF comes with Diameter interfaces which are based on the 3GPP standard. The 3GPP standard defines the Rx interface between an Application Function (AF) and PCRF for implementing QoS and Policy. Rx is utilized because it meets our architecture's needs.

4.2 The Video Telephony Service

In this section we describe our proposed architecture for the Video Telephony Service as shown in Figure 4.2. It is made up of the following components : Video Telephony Application, Application Policy Rules, Session Information Repository and VHSS.

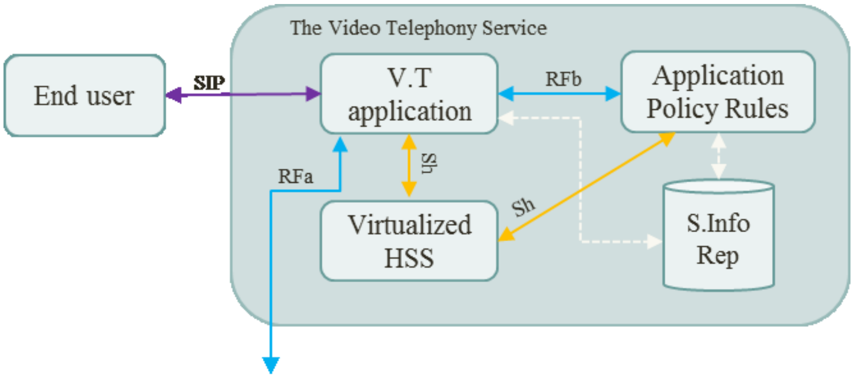

Figure 4.2: The Video Telephony Service Architecture

The Video Telephony Application creates and manages QoS-enabled video sessions between end-users. The end-users will therefore communicate with the Video Telephony Application using SIP.

The Session Initiation Protocol (SIP) [36] is used as the signaling protocol for session establishment. The SIP is a signaling protocol used to create, modify and terminate sessions with one or more users. SIP sessions can include voice communications, internet telephone calls or multimedia applications. The information used when creating sessions like the session ID is stored in the session information repository. We chose SIP because it is the most widely used session signaling protocol. There are many reasons for its popularity. It is a much simpler protocol than its prior competitors. SIP was not built solely as a replacement for the telephone system and it therefore allows extensions, as well as offering additional services beyond just simple calls.

VHSS has the same functionality as non-Virtualized HSS. It plays a database role for storing two types of data: temporary and permanent. Temporary data relates to the user information that is stored by the application during registration (e.g. IP address and port number). Permanent data includes the user subscription information, such as the QoS profile and the media type the user has subscribed to (e.g. Audio only, audio/video). We will explain more about VHSS in Section 4.3.

The interfaces between the VHSS and the Video Telephony Application and the Application Policy Rules are Sh Diameter interfaces. We use Diameter because it is a 3GPP standardized interface. The

3GPP standard defines the Sh interface which is the reference for interactions between the HSS and SIP Application Server (AS) [37].

The Application Policy Rules entity includes the policies and rules that allow the application to choose the appropriate session by downgrading or terminating. By selecting the appropriate session resources can be freed up when needed. In order to make this decision the Application Policy Rules entity retrieves the information about available sessions from the Session Information Repository. The entity must also communicate with VHSS to retrieve the QoS profile. The communication between the Application Policy Rules and the Video Telephony Application (i.e RFb) is REST.

4.2.1 Procedures

This section discusses two main procedures: new session establishment and an ongoing session update. The supporting procedures (i.e. Resource release) are also presented.

<u>*New session establishment:*</u> To establish a new session, the end-user sends an INVITE request with the ID of the target end-user (i.e. The callee) to the Video Telephony Application server of his/her service provider. The application server asks the QoS Enabler to establish a new session between the two users using the QoS profile registered in the caller's profile supplied by the VHSS. The QoS Enabler communicates with the PCRF in order to execute the request. If not enough network resources are available, the PCRF will reject the request. The QoS Enabler will then release (if possible) the resources needed for the new session and reiterate its request to the PCRF. If not enough resources can be released, the new session request is rejected. Resource release is performed using the "resource release" procedure described below.

<u>*Session update:*</u> To update (downgrade or upgrade) an ongoing session, the end-user sends an update request to his/her service provider, which in turn transfers the request to the QoS Enabler. The update request should include the new QoS profile that the end-user is willing to use which is retrieved from VHSS. The update procedure continues along the same path as the new session establishment procedure.

The QoS Enabler issues a request for the session update with the new profile. If the request is rejected, the QoS Enabler tries to release the necessary resources to support the update and then reiterates the request. If no resources can be released, the update request is rejected.

Resource release: Resource release is an action based on the end-user and his/her service provider priority, and is performed in three steps. First, the QoS Enabler identifies the service provider(s) whose sessions should be modified (i.e. Downgraded or terminated). Second, it sends a resource release request to the selected providers with the amount of resources to be freed up. Third, the target service providers identify and modify the appropriate sessions and report back to the QoS Enabler. These three steps are detailed below.

Selection of the service provider(s) whose sessions should be downgraded/terminated: When a session establishment request is rejected by the PCRF, the QoS Enabler asks the Network Policy Rules to choose the proper service provider whose sessions should be downgraded (or eventually terminated). Below is an example of choosing a service provider when a session downgrade or termination is needed. Let EUi be the end-user whose request was rejected and SPi be the EUi's service provider. The Network Policy Rules entity chooses the service provider SPj that satisfies the following four conditions: 1) SPj has one or more ongoing sessions; 2) SPj has a lower priority than SPi; 3) Downgrading (or eventually terminating) 'n' sessions belonging to SPj will free up the amount of resources needed to establish the EUi session; and 4) SPj has the lowest priority among the other service providers that satisfy the first three conditions. If the Network Policy Rules entity finds SPj, it sends the SPj ID to the QoS Enabler. If no SPj is found, the SPi ID is sent back to the QoS Enabler.

Send a resource release request to the selected service providers and identify the sessions to be modified: The QoS Enabler sends a resource release request to the chosen SPj application server (APj) with the amount of resources to be released. The ASj uses the Application Policy Rules to identify the sessions to be downgraded (or terminated) as well as the number of these sessions. The session termination alternative is considered only if there are no sessions whose downgrading could release the requested resources. The sessions selected are those with the lowest priorities. If SPj=SPi, the chosen sessions should also have a lower priority than the session to be established. The information about the ongoing sessions is retrieved from the Session Information Repository.

When the sessions to be modified have been identified, the ASj sends the appropriate modification request (i.e. Session update or release) to the QoS Enabler, which realizes the request via a call to the PCRF. Figure 4.3 summarizes the remaining steps for a new session establishment procedure after a request has been rejected by the PCRF.

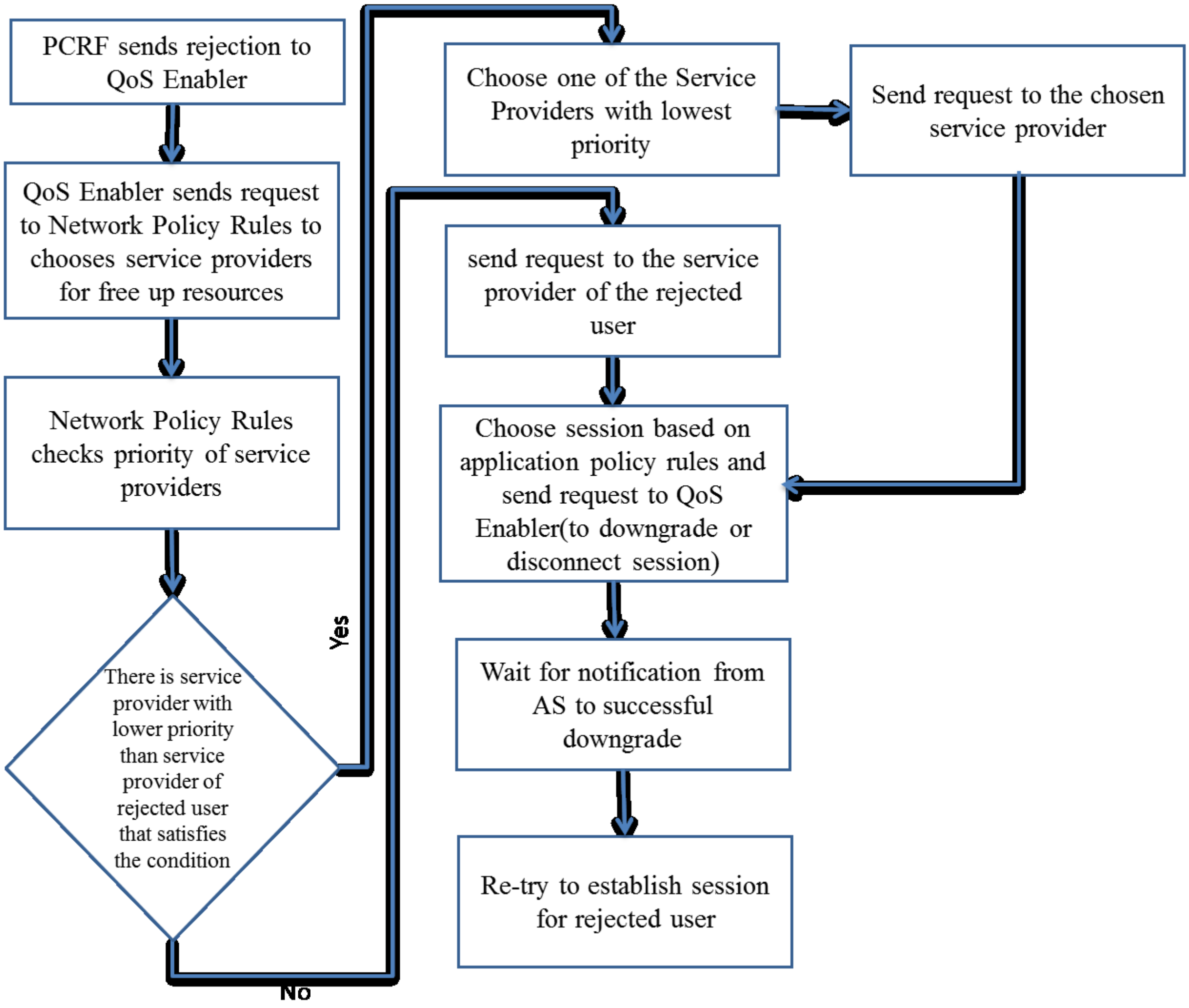

Figure 4.3: Session establishment procedure when not enough resources are available

4.3 VHSS Enabler

In this section, we will provide an overview of the functions and roles of the VHSS Enabler and describe how it creates the proposed VHSS. In the following subsection we will describe the entities, the interfaces, and the operational procedures of VHSS.

4.3.1 Overview

As mentioned in Section 4.1, the VHSS Enabler is part of the EPC provider and enables the provider to offer the Virtualized instance of the HSS to Video Telephony Service providers. The VHSS Enabler has access to a pool of the available physical resources. Based on our proposed architecture it uses these resources to create an appropriate VHSS for Video Telephony Services. The interaction between the Video Telephony Service and the VHSS Enabler is a REST-based interface. The following subsection describes the proposed architecture of the VHSS.

4.3.1.1 Virtualized HSS

Figure 4.4 shows the main components of the HSS which includes the Diameter server and the database.

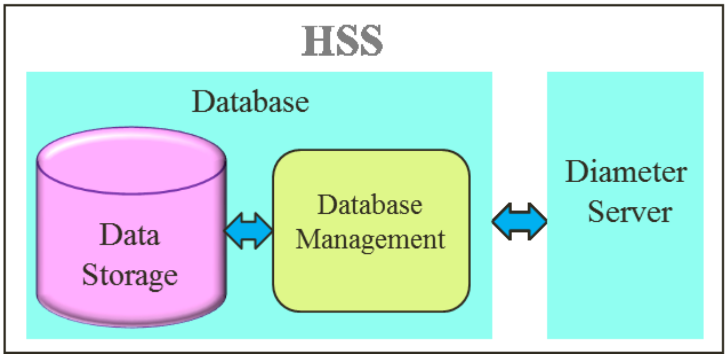

Figure 4.4: The HSS components

The database is decoupled into two parts : The database management system and storage. Based on the architecture of the HSS, the VHSS which we proposed is composed of three layers as represented in Figure 4.5: The Diameter Layer (top layer), the Database Management Layer (middle layer) and the Storage Layer (lower layer). Each layer has a responsibility for one main part of the HSS functionality.

The lower layer is the distributed storage which provides fault-tolerant shared disks. The middle layer is the database management which includes multiple nodes that provide full ACID (Atomicity, Consistency, Isolation and Durability) -compliance guarantees and ensures that all database transactions are processed reliably. In the top layer of the architecture we have several Diameter entities which are responsible for receiving the Diameter requests and processing them based on the different types of applications (Sh, Cx,..). We will divide VHSS into three subsections. First we will discuss the Functional Entities of each layer in VHSS. Then we will look at the Communication Interface and finally we will examine the Elasticity Procedure.

4.3.1.2 Functional Entities:

The Diameter Layer consists of three main functional entities: Virtual Diameter Routing Agent, Virtual Diameter Entity and Diameter Layer Management Entity

- Virtual Diameter Routing Agent :

  The Virtual Diameter Routing Agent is the first entry point of Virtualized HSS. It is a functional element that ensures requests are received and forwarded to the suitable Virtual Diameter entity. When the components need to communicate with the VHSS, they discover the list of available Virtual Diameter Routing Agents and forward the request to the entity with the least load. Moreover, the Virtual Diameter Routing Agent is responsible for finding the next-hop.

  To find the available Virtual Diameter entities, the Virtual Diameter Routing Agent runs the Service Location (SRVLOC) peer discovery mechanism which will be explained later. After having discovered the appropriate entities, the Virtual Diameter Routing Agent also needs to know which entity has the least load. Subsequently, it will forward the request to the Virtual Diameter entity selected by the load balancing algorithm.

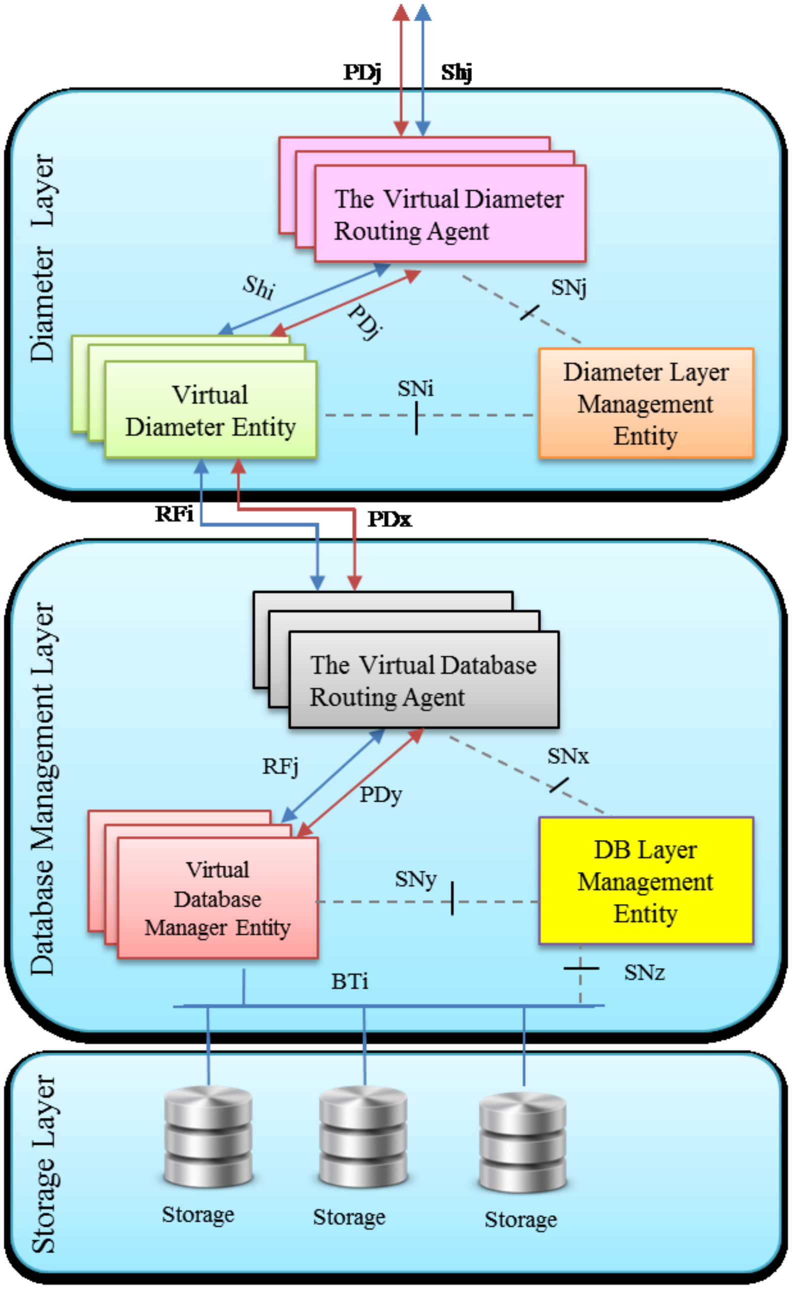

Figure 4.5: The VHSS Architecture

Load Balancing is required to spread the load in an efficient way and thus balance the total load across several entities. For choosing the appropriate entities the Virtual Diameter Routing Agent needs to know the load on the layer. For this reason it communicates with another entity of this layer, called the Diameter Layer Management Entity which is responsible for calculating the load on the entities.

After the Virtual Diameter Routing Agent discovers the list of available Virtual Diameter entities, it chooses the suitable Virtual Diameter entity based on the load balancing algorithm and forwards the request to this chosen Virtual Diameter entity.

- Diameter Layer Management Entity:

  The Diameter Layer contains multiple entities and the number of these entities relates to the specific load on the layer. The Diameter Layer Management entity tracks this load by receiving notifications from the other entities of the Diameter Layer and runs the customized algorithm in order to make the decision about growing or shrinking the number of Virtual Diameter entities or Virtual Diameter Routing Agents on the layer.

  Entities of the Diameter Layer periodically send notifications to the Diameter Layer Management entity which shows the entity's load and health. The Diameter Layer Management entity then initiates the Virtual Diameter Routing Agent and the Virtual Diameter entity.

  The Diameter Layer Management has a redundant entity in case of a failure, thus avoiding a single point of failure in the system.

- Virtual Diameter Entity:

  The Virtual Diameter entity acts as both client and server. The client side is used for communicating with the lower layer while the server side is for receiving and serving the requests from the Virtual Diameter Routing Agent. The Virtual Diameter entity provides the Diameter base functionality and Diameter applications. The distributed architecture provides high capacity and fault tolerance, meaning if one entity fails another entity can quickly be found to replace it. The Virtual Diameter entity is responsible for communicating with the Diameter Layer Management entity periodically for giving load information.

The Virtual Diameter entity also replies to the service discovery requests from the Virtual Diameter Routing Agent. In addition, it also needs to discover the Virtual Database Routing Agent in the lower layer.

The Database Management Layer consists of three main functional entities: Virtual Database Routing Agent, Virtual Database Manager Entity and Database Layer Management Entity.

- <u>Virtual Database Routing Agent:</u>

  The Virtual Database Routing Agent is the first entry point of the Database Management Layer. It is responsible for ensuring requests are received and forwarded to the appropriate Virtual Database Manager entity. When the Virtual Diameter entities need to communicate with the lower layer, they discover the list of available Virtual Database Routing Agents and forward the request to the one of them. Moreover, the Virtual Database Routing Agent is responsible for finding the next-hop. To do this, it will run the SRVLOC peer discovery mechanism to get the available list of Virtual Database Manager entities.

  Load Balancing is also required for spreading the requests. The Virtual Database Routing Agents are responsible for running the Load Balancing algorithm to find the appropriate entity. The Load Balancing algorithm needs to be aware of the load of the layer so that the Virtual Database Routing Agent can communicate with the Database Layer Management entity and ask about the layer's load.

- <u>Virtual Database Manager Entity</u>:

  In our proposed Database Manager Layer there are multiple instances of Virtual Database Manager entities processing requests. The entities process these database requests by retrieving or updating the data from the Storage Layer. Multiple Virtual Database Manager entities help us to scale the Database Manager Layer, thus allowing the system to manage excess demand. A

Virtual Database Manager entity can support relational databases and non-relational databases (SQL or NoSQL). The Virtual Database Manager entity should support ACID transaction. The Virtual Database Manager must communicate with another entity called the Database Layer Management so it can inform it of its loads at a specific time.

- Database Layer Management Entity:

  The Diameter Layer contains multiple entities and the number of these entities relates to the specific load on the layer. The Database Layer Management entity tracks this load by receiving notifications from the other entities of the Database Management layer and runs the customized algorithm in order to make the decision about growing or shrinking the number of Virtual Database Manager entities and Virtual Database Routing Agents on the layer.

  Entities of the Database Management layer periodically send notifications to the Database Layer Management entity which shows the entity's load and health. The Database Layer Management entity then initiates the Virtual Database Routing Agent and the Virtual Database entity.

  The Database Layer Management has a redundant entity in case of a failure, thus avoiding a single point of failure in the system.

The Storage layer is made up of multiple storage entities.

- Storage Entity:

  The Storage entity hosts related data of all users which is required by the other components. It behaves as a distributed shared storage with data being broken into several blocks and these blocks are distributed between storages. The storages are mirrored, meaning there is no master-slave approach.

  The shared-nothing architecture is not suitable for Virtualized environments because the storage or specific partition of the storage needs to be hardwired to a specific Database Manager entity.

In the shared-disk approach, scalability is achieved by having multiple storage entities which can add more entities on the fly based on demand.

4.3.1.3 Communication Interface

The interfaces between the Virtual Database Routing Agent and the Virtual Diameter entity (i.e RFi) and the Virtual Database Manager (i.e RFj) are REST-based. As we said before, we selected REST because it is standard-based, lightweight and can support multiple data formats. Table 4-2 is an example of the REST resources in our architecture. It summarizes the main resources we defined for the Virtual Database Routing Agent [35].

Resources	Operation	HTTP Action
A user	Create: Store information of the user	POST: /Profile/UserProfiles
	Read: Get information of the user	GET: /Profile/UserProfiles/{userID}
	Delete: Remove information of the user	DELETE: /Profile/UserProfiles/{userID}
	Update: Modify information like class of service	PUT: /Profile/UserProfiles/{userID}

Table 4-2: Main Database Routing Agent Resources

The communication between the Virtual Diameter Routing Agent and the Application (i.e. Shj) and the Virtual Diameter entity (i.e. Shi) are ensured via a Sh Diameter interface. Diameter is a 3GPP standardized interface for EPC. The 3GPP standard defines the Sh interface which is the reference for interactions between the HSS and SIP Application Server (AS). Table 4-3 is an example of some of the Sh messages based on 3GPP standard [37].

Command Name	Abbreviation	Source	Destination
User-Data-Request	UDR	AS	HSS
User-Data-Answer	UDA	HSS	AS
Profile-Update-Request	PUR	AS	HSS
Profile-Update-Answer	PUA	HSS	AS

Table 4-3: Diameter Sh Messages

The communication between the Virtual Diameter Routing Agent and Virtual Diameter entity (i.e PDi) and the applications (i.e PDj) are publication and discovery interfaces, as are the interfaces between the Virtual Database Routing Agent and the Virtual Diameter entity (i.e PDx) and the Virtual Database Manager entity (i.e PDy). For discovery and publication we chose the SRVLOC mechanism.

SRVLOC is a service discovery protocol that provides a scalable way for computers to find the information about existence and location in a network dynamically. Where there are many different clients and/or services available, the protocol is adapted to make use of nearby Directory Agents that offer a centralized repository for advertised services.

SRVLOC uses three types of nodes: User Agent, Service Agent and Directory Agent. The User Agent is a process that requires the service provided by the Service Agent. The role of the Directory Agent is to collect service advertisements. These nodes are used in two different kinds of networks, small or large. In a small network, in order for the User Agent to connect directly to the Service Agent the User Agent BroadCasts/Multicasts the SrvRqst over the network. And in response, the Service Agent which has the suitable service will send the Unicast SrvRply to the User Agent.

Whereas in a larger network, one or more directory agents are used. The User and Service Agents can discover Directory Agents in two ways. First, they issue a Multicast Service Request for the ‘Directory

Agent' service when they start up. Second, the Directory Agent sends an advertisement infrequently, which the User and Service Agents scan for. After discovering the Directory Agent the Service Agents send register messages (SrvReg) containing all their services and receive acknowledgements in reply (SrvAck). User Agent unicast requests to Directory Agents instead of Service Agents if any Directory Agents are known[38].

The interfaces between the Diameter Layer Management entity and the Virtual Diameter Routing Agent (i.e SNj) and the Virtual Diameter entity (i.e SNi) are Simple Network Management Protocol (SNMP) - based. The interfaces between the Database Layer Management entity and the Virtual Database Routing Agent (i.e SNx) and the Virtual Database Manager entity (i.e SNy) and the Storage (i.e SNz) are also SNMP-based. SNMP is widely used in the networks and allows us to monitor the network entities from a management host [39]. The communication between the Virtual Database Manager entity and the Storage (i.e Bi) is based on the B-tree indexing technology. B-tree is a tree data structure that keeps data sorted and allows searches [40].

4.3.1.4 Elasticity Procedure

This section discusses the procedure when the system discovers an unexpectedly high or low load on one of the layers and needs to add or remove new entities.

In order to do this, the entities of each layer must periodically send notifications which include information on the layer's load. The Layer Management entity can be either the Diameter Layer Management entity or the Database Layer Management entity. This Layer Management entity collects all the information and runs an algorithm to ensure the load on the entities is equal to their capacities. The Layer Management entity runs the algorithm which includes the defined threshold to measure the load of the layer. If the layer's load exceeds the threshold then the Layer Management entity will recognize the layer needs more entities and it will add them. But if the load is less than the threshold, the

Layer Management entity will select the entity with no load and terminate it. The flowchart shown in Figure 4.6 summarizes how the Layer Management entity works.

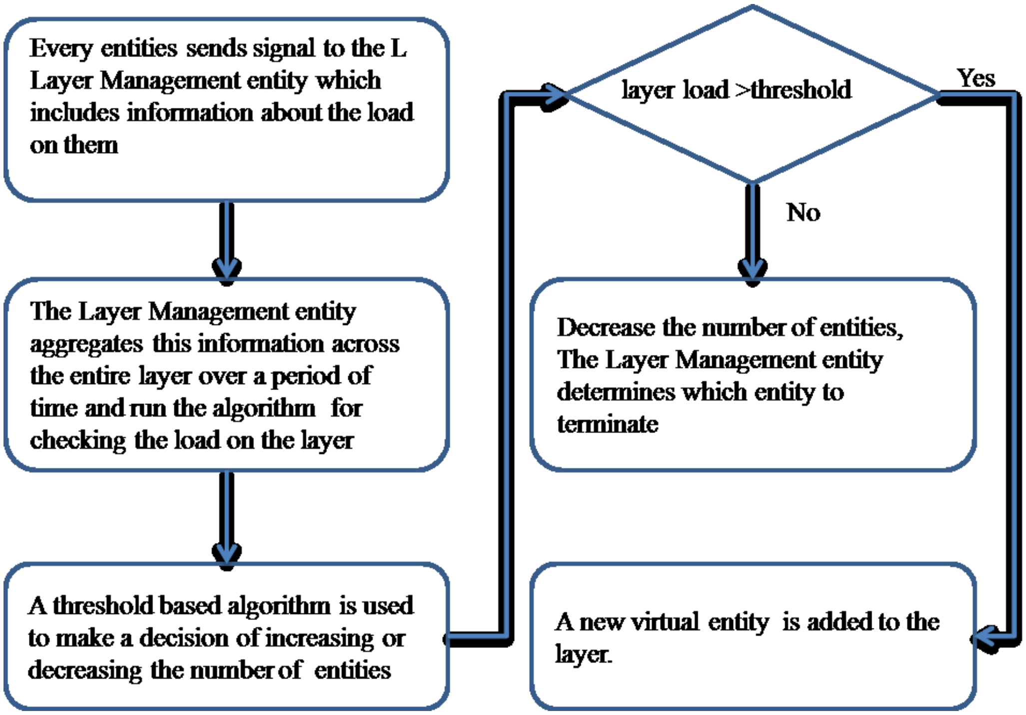

Figure 4.6: The summary of how the Layer Management entity works

To clarify the explanation given in the Elasticity Procedure subsection we have an example (Figure 4.7) outlining how the Database Layer Management entity works in the Database Management Layer. In this example we look at one of the Video Telephony Service providers which experiences an unexpected increase in load. Let us say during an afternoon a service provider suddenly receives a large number of call requests which it needs to initiate simultaneously. This creates an excessive load on the VHSS and the Database Management Layer is flooded and struggles to handle all the database requests. At the specific time when the entities have to send their notifications to the Database Layer Management entity (step 1,2,3), the Database Layer Management entity will discover the sudden excessive load based on the information collected from the notification. It will then make a decision to add more Virtual Database Manager entities to the Database Management Layer so the layer can cope with the load increase.

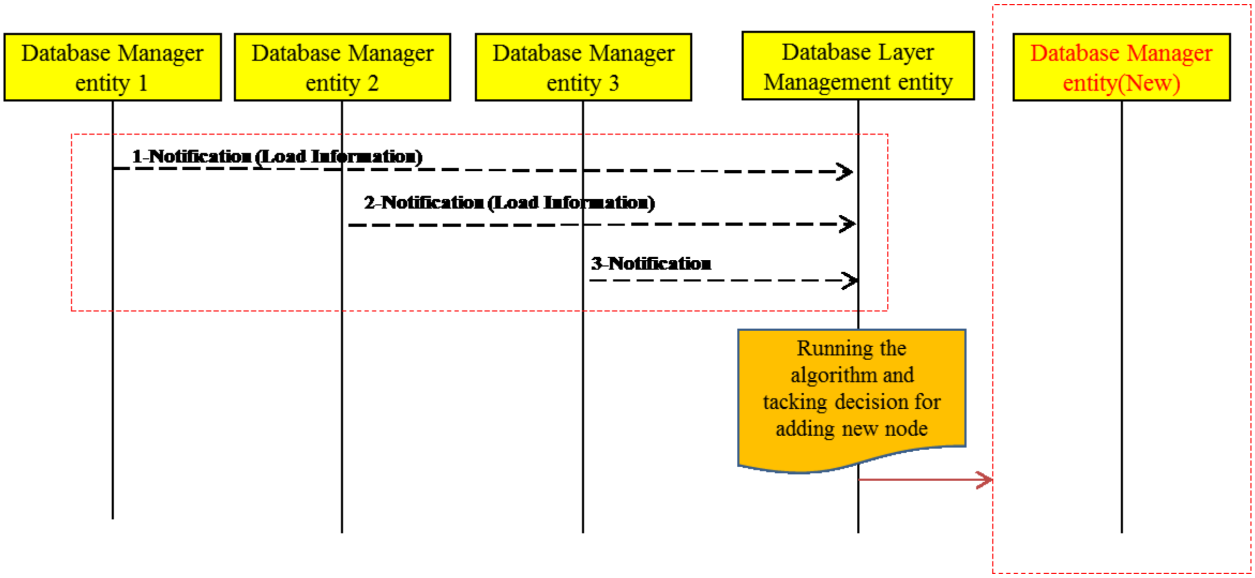

Figure 4.7: The Sequence diagram of adding a new entity

4.4 Requirements met by the architecture

The refined architecture satisfies all the requirements described in the previous chapter. Firstly, the Video Telephony Application supports standard video telephony protocols and has the ability to create, delete or modify sessions. By using the Differentiated QoS Service Delivery Platform this architecture enables the differentiation QoS feature in the EPC and gives end-users the opportunity to subscribe to given classes of service (e.g. Gold, Silver and Bronze) and have their sessions served with the corresponding parameters. The priorities of the session and service provider are matched using the Application Policy Rules and the Network Policy Rules.

Secondly, the VHSS is a loosely coupled architecture which supports all functionality of non-VHSS by spreading the main functions of the HSS into three defined layers. The architecture supports isolation between different Video Telephony Service providers by applying a multi-tenancy approach. In addition, the architecture has no single point of failure. For example, the storage nodes are mirrored and if one storage node fails no data will be lost. Moreover, because the Virtual Database Manager entities all have an access to the shared-data, if one entity fails the requests are routed to remaining entities. The

Diameter Layer Management entity and the Database Layer Management entity also have a redundant entity to avoid a single point of failure.

Layered architecture helps each layer grow up or shrink independently and a change in one layer does not influence any other layers. Another key point of this architecture is standard-compliance. The communication between two layers is based on standard interfaces. Additionally, the Layer Management entities (Diameter Layer Management entity and Database Layer Management entity) are required in each layer in our proposed architecture because of the elasticity feature it brings. Henceforth, layers can grow or shrink independently with the help of the Layer Management Entity.

4.5 Illustrative Scenario

This section studies a scenario and will show how the Video Telephony Application with differentiated QoS and VHSS can be realized using the proposed architecture.

4.5.1 Scenario:

Figure 4.8 shows the associated sequence diagram for this scenario. During the audio/video session between the Silver end-users (Alice and Bob), a Gold end-user (Emma) sends an INVITE message to the Video Telephony Application to have an audio session with another Gold end-user (Charlie) (Step 1). The application gets the profiles of the users involved in the new session (Emma and Charlie) from the VHSS (Steps 2 and 3), and extracts the necessary information for the establishment of the new session (e.g. The end-users’ QoS profiles, IP addresses and port numbers). It then uses this information to construct a POST request that it sends to the QoS Enabler (Step 4). The QoS Enabler issues an Authentication-Authorization-Request (AAR) Diameter message to the PCRF, in order to reserve the network resources required for the new session (Step 5). The PCRF replies with an Authentication-Authorization-Answer (AAA) to reject the QoS Enabler’s demand (Step 6). We assume that this rejection is because there are insufficient resources available to allow the establishment of the new session. The QoS Enabler then sends a request to the Network Policy Rules entity to get the list of service providers whose ongoing sessions it can modify (Steps 7 and 8). In this scenario, only one service provider ID is returned. Therefore the QoS Enabler sends a POST request to the Video Telephony Application (V.T Application) of the selected service provider to request session modification (Step 9). The Video Telephony Application consults the Application Policy Rules entity (Step 10), which checks the session information repository for the session to modify (Steps 11 and 12). The Application Policy Rules entity replies with the ID of the session between Alice and Bob (Step 13), and the information that this session should be downgraded from audio/video to audio only. The application server then sends a re-INVITE request to both users with the new parameters (i.e. Audio only). For the sake of clarity, only the communication with Alice is shown in Figure 4.6 (Steps 14 and 15). Next, the application server sends a PUT request to the QoS Enabler to ask for bearer updates for both Alice and Bob (Step 16), which result in the release of the required resources. The QoS Enabler then proceeds with the re-establishment of the God session between Emma and Charlie (Steps 17 to 20).

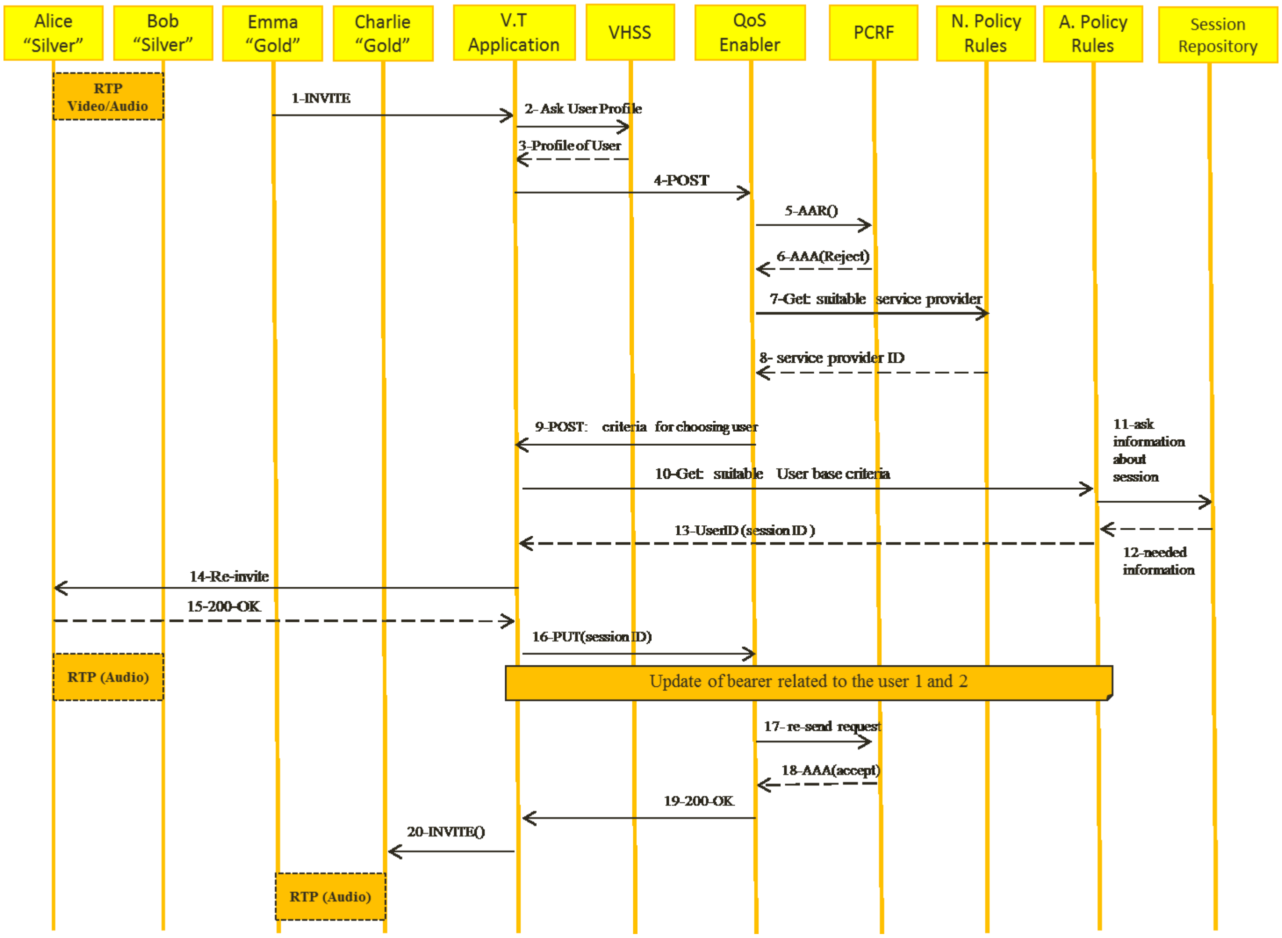

Figure 4.8: The sequence diagram

4.5.1.1 The VHSS Steps

In the above sequence diagram (Fig. 4.9) VHSS is shown as a block box whereas in Figure 4.5 we look in detail at VHSS depicted by several steps illustrating the VHSS operation.

The Video Telephony Application sends a multicast service request to identify the list of available Virtual Diameter Routing Agents (Step 1). When the Diameter Routing Agents send back the unicast reply message (Step 2), the Video Telephony Application will forward the request to the appropriate Virtual Diameter Routing Agent (Step 3). The same procedure will be undertaken by the Virtual Diameter Routing Agent to find the Virtual Diameter entity (Steps 4 and 5). However, in order to know the load of the layer the Virtual Diameter Routing Agent must send a request to the Diameter Layer Management entity for this information (Step 6).

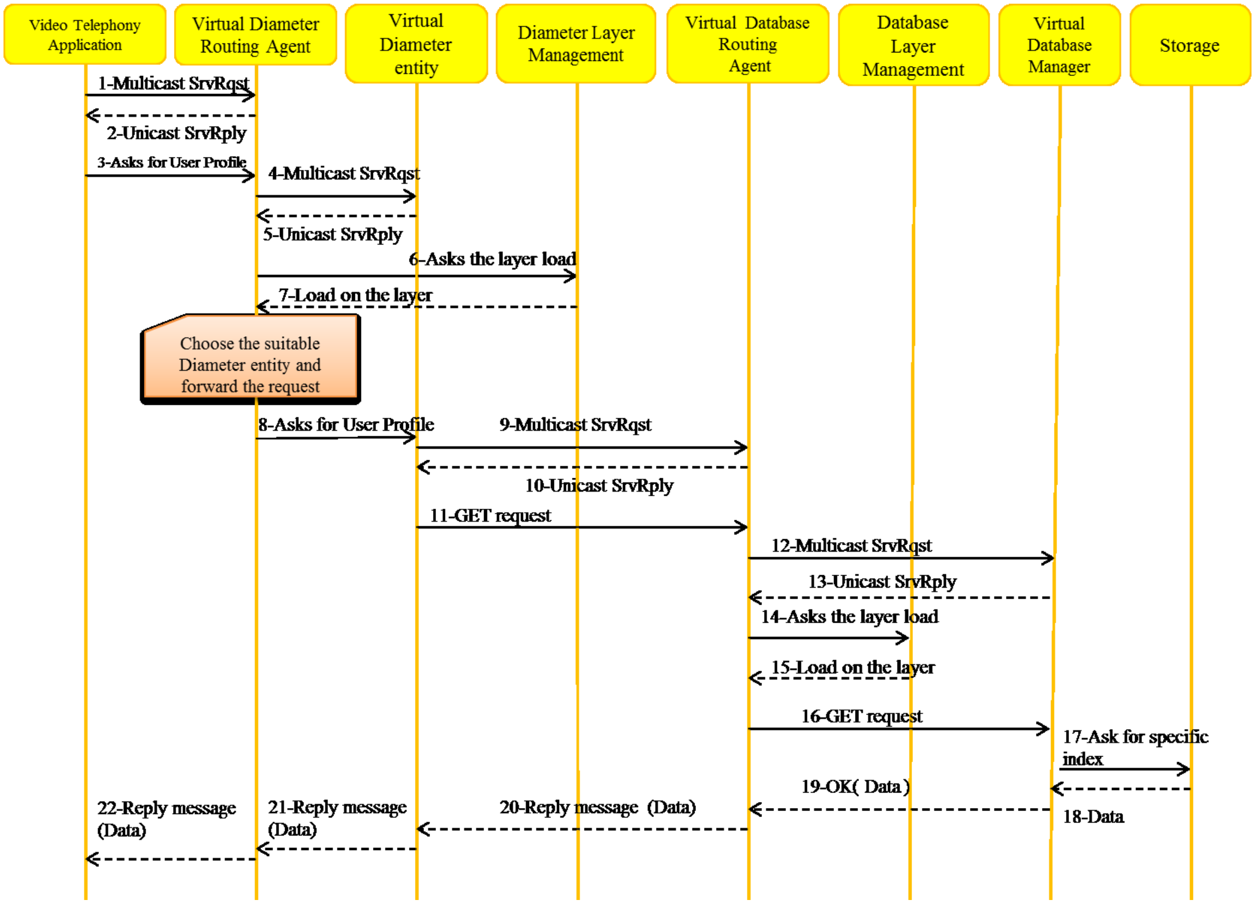

Figure 4.9: The sequence diagram for VHSS (retrieve the data)

When this load information is received (Step 7) the Virtual Diameter Routing Agent will choose the appropriate Virtual Diameter entity and forward the request to it (Step 8). After the Virtual Diameter entity processes the request it needs to discover the first entry point of the lower layer. To do this, the discovery procedure used by Virtual Diameter Routing Agent and Video Telephony Application is repeated by the Virtual Diameter entity (Steps 9 and 10). When the desired entry point is selected the Virtual Diameter entity will forward a request to the Virtual Database Routing Agent (Step 11). The Virtual Database Routing Agent discovers the list of available Virtual Database Manager entities and communicates with the Database Layer Management entities to take the load information (Steps 12,13,14 and 15). Once the appropriate entity has been found, the Virtual Database Routing Agent forwards the request to it (Step 16). In the next step the Virtual Database Manager entity processes the request. In order to retrieve the data from the Storage, the Virtual Database Manager uses an index-based search (Step 17). When this data is located by the Storage based on the index supplied by the Virtual Database Manager, it is sent back through the layers to the Video Telephony Application (Steps 18,19,20,21 and 22). Figure 4.10 shows the sequence diagram when the Video Telephony Application wants to store data in the VHSS instead of retrieving the data from VHSS.

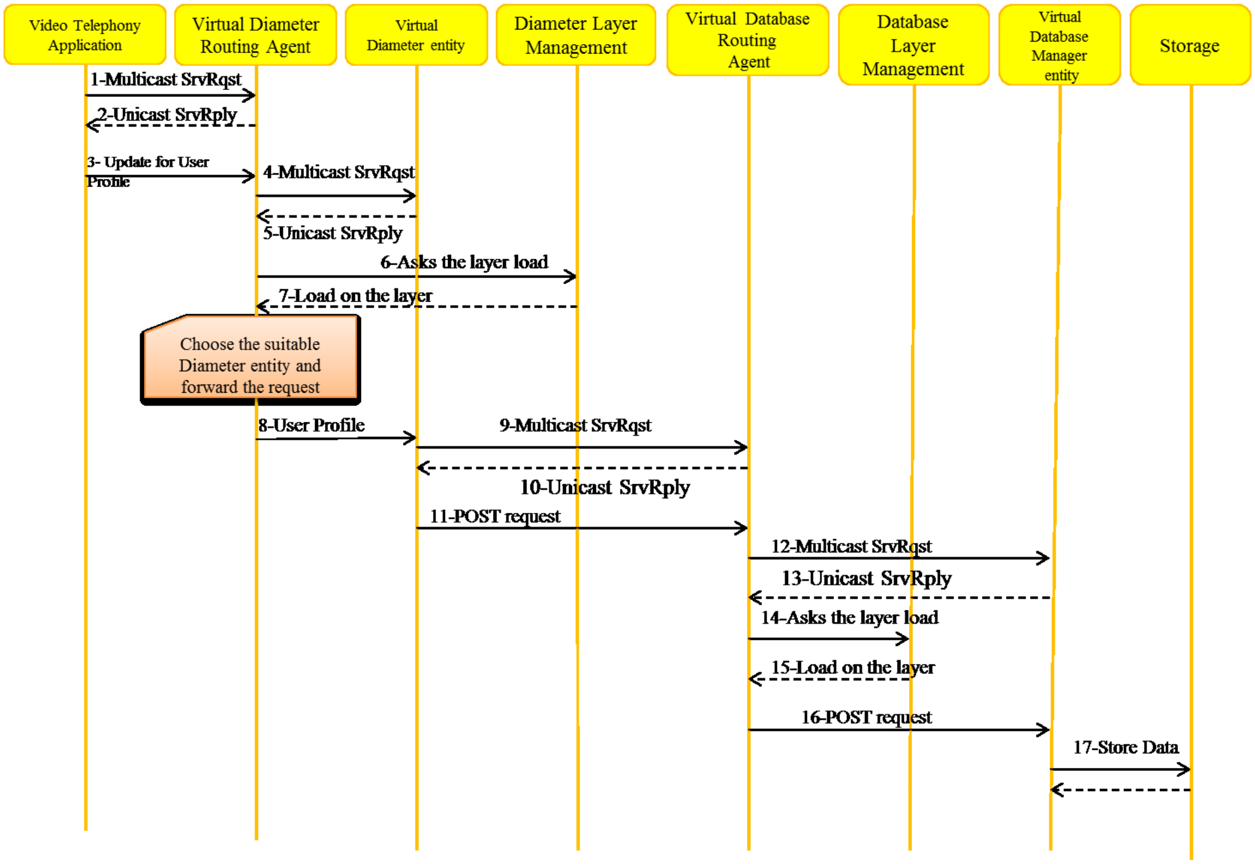

Figure 4.10: The sequence diagram for VHSS (store the data)

4.6 Chapter Summary

In this chapter, we presented the overall architecture, which included the main functional components such as a Video Telephony Service, the Network Policy Rules, VHSS Enabler, and the QoS Enabler. The communication interfaces between the system entities are categorized as: REST interfaces, Diameter interfaces and a Signaling interface. We then discussed two proposed architectures: Video Telephony Service architecture and VHSS Enabler. We also concluded that the architecture satisfies all the requirements described in the previous chapter. Finally, we demonstrated the interaction between the entities by presenting an illustrative scenario.

Chapter 5

5 Validation: Prototype and Evaluation

In the previous chapter we discussed our proposed architecture for QoS Enabled Video Telephony with a VHSS on EPC. This chapter focuses on the software architecture for the system entities. In addition, we present the implemented prototype and the measurements made in order to validate the proposed architecture. We start this chapter by presenting the software architecture of the main functional entities and present the operational procedures that show how the entities interact. Subsequently, we present a prototype architecture and we will then introduce the proof of concept prototype that we have implemented. In the last part, we discuss some performance measurements in order to validate our architecture.

5.1 Software Architecture

In this section, we describe the proposed software architecture for the main entities in our architecture. Then, we discuss the software's operational procedures.

5.1.1 The Architecture of System Components

Our proposed system is divided into three main building blocks: Video Telephony Service, Differentiated QoS Service Delivery Platform, and VHSS. Each of these blocks includes important system components that will be discussed in the following subsections.

5.1.1.1 The Video Telephony Service

The Video Telephony Service is made up by the Video Telephony Application and the Application Policy Rules.

The Video Telephony Application:

Figure 5.1 shows the proposed architecture of the SIP-based Video Telephony Application. It consists of four entities: SIP Servlet, Diameter Client, REST Client and Session Manager. The SIP Servlet entity is responsible for handling the received SIP messages (e.g. SIP INVITE) and for creating and sending the appropriate responses. The Diameter Client implements the Sh Diameter interface, which is used to communicate with the Virtualized HSS. The REST Client is for communicating with the REST-based entities of the overall architecture. These REST-based entities are the VHSS Enabler, the QoS Enabler and the Application Policy Rules. The REST Client is used to send a decision to request the Application Policy Rules and communicate with the QoS Enabler. The Session Manager constitutes the core component of the application. It executes the entire logic of the application related to session management. This includes session establishment and updates, the selection of the ongoing sessions to be modified, and the management of the data stored in the session information repository.

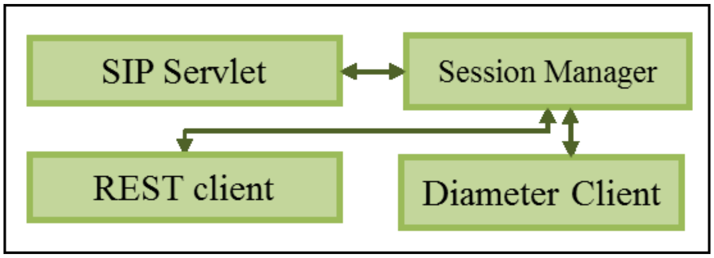

Figure 5.1: The Video Telephony Application Architecture

The Application Policy Rules:

Figure 5.2 shows the proposed architecture of the Application Policy Rules. It consists of two entities: Policy Manager and RESTful API. The Policy Manager is responsible for applying the service provider policies in order to decide whether the sessions are to be downgraded/terminated. A RESTful API module exposes the entity functionalities via a RESTful interface.

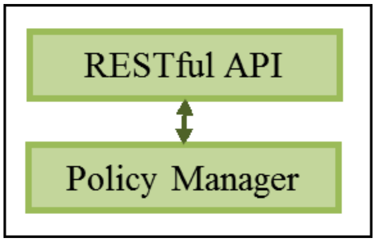

Figure 5.2: The Application Policy Rules Architecture

5.1.1.2 *The Differentiated QoS Service Delivery Platform*

The Differentiated QoS Service Delivery Platform consists of the QoS Enabler and The Network Policy Rules.

The QoS Enabler:

The QoS Enabler in [26] is the basis of our QoS Enabler from which we reused one component and changed it based on our requirements. The QoS Enabler is made up of four entities (Figure 5.3): REST Client, Diameter Client, QoS Manager and RESTful API. The Diameter Client implements the Rx Diameter interface; which is used to communicate with the PCRF. The REST Client is used to send REST requests to the Network Policy Rules to free up some resources. The RESTful API exposes the QoS Enabler's capabilities to the Video Telephony Service. The QoS Manager receives the application's requests to create, modify and delete sessions.

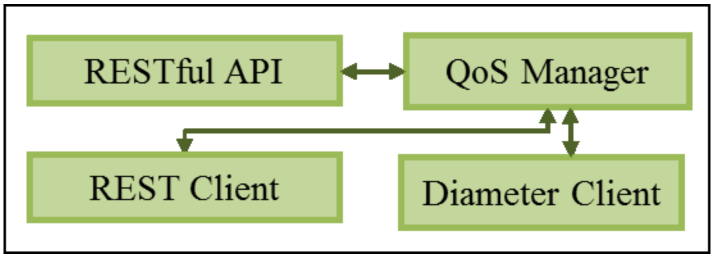

Figure 5.3: The QoS Enabler

The Network Policy Rules:

Figure 5.4 shows the proposed architecture of the Application Policy Rules. It consists of two entities: Policy Manager and RESTful API. The Policy Manager is responsible for prioritizing the service providers based on their defined policies and service level agreements. The Policy Manager can then choose the service provider with the lowest priority to free up resources. A RESTful API module is used to expose the entity functionalities via a RESTful interface.

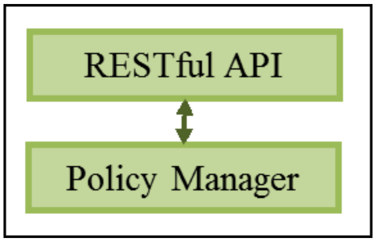

Figure 5.4: Network Policy Rules Architecture

5.1.1.3 VHSS

As we mentioned before, VHSS contains three layers: Diameter Layer, Database Management Layer and Storage Layer. We did not provide software architecture for the Virtual Database Manager entity, the Database Layer Management entity and the Storage entity because we have decided to reuse existing products in their place The existing products reused are shown in a red box clearly in Figure 5.5.

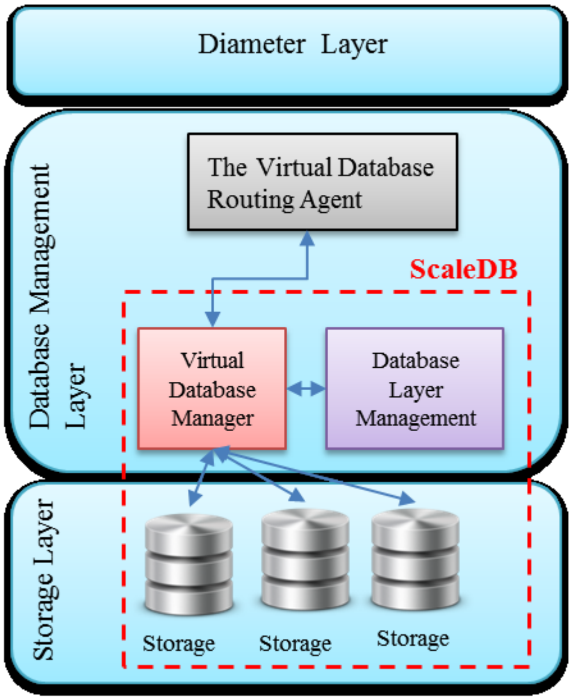

Figure 5.5: The existing product reused

The Diameter Layer consists of the following three components: Virtual Diameter Routing Agent, Virtual Diameter entity and Diameter Layer Management entity. The software architecture of these entities is explained below.

The Virtual Diameter Routing Agent:

The Virtual Diameter Routing Agent is a Proxy Agent which can be used to forward Diameter messages. The proposed architecture of the Virtual Diameter Routing Agent is shown in Figure 5.6. It includes four entities: Sh API, SNMP Client, User Agent, Load Manager and Diameter Client. The Sh API exposes the entity functionalities via Sh interfaces and it receives the requests from the Video Telephony Application and the Application Policy Rules. The User Agent is responsible for discovering available Virtual Diameter entities and sending a list of available entities to the Load Manager. The Load Manager then communicates with the Diameter Layer Management entity via the SNMP Client and requests load information. After receiving this load information as well as the list of available entities, the Load Manager runs a load balance algorithm in order to choose the proper Virtual Diameter entity for forwarding the request. The Diameter Client implements the Sh Diameter interface which it uses to communicate with the Virtual Diameter entities.

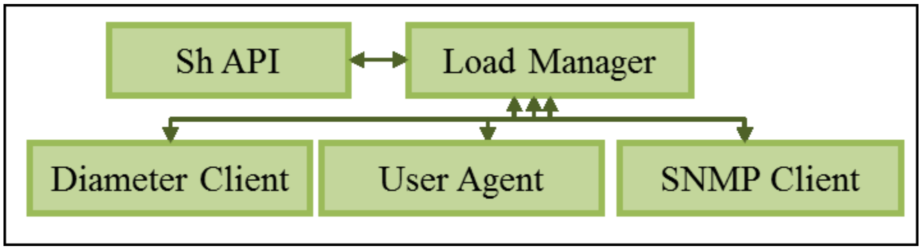

Figure 5.6: The Virtual Diameter Routing Agent Architecture

The Virtual Diameter entity:

The Virtual Diameter entity is a Diameter Server which is responsible for serving the Diameter request. Figure 5.7 shows the software architecture of the Virtual Diameter entity which includes five entities: Sh API, Service Agent, Diameter Manager, User Agent, SNMP Client and REST Client. The Sh API exposes the entity functionalities via Sh interfaces and it receives the request. The Service Agent is responsible for enabling the component to be discovered by the Virtual Diameter Routing Agent. The Diameter Manager is responsible for processing the request and forwarding it to the lower layer with the help of the REST Client. However, to forward the request the Diameter Manager needs to know the list of available Virtual Database Routing Agents. To do this, it uses the Discovery Agent to discover the first entry point of the lower layer. Another responsibility of the Virtual Diameter entity is to communicate with the Diameter Layer Management entity via the SNMP Client entity to inform the Diameter Layer Management entity of the load.

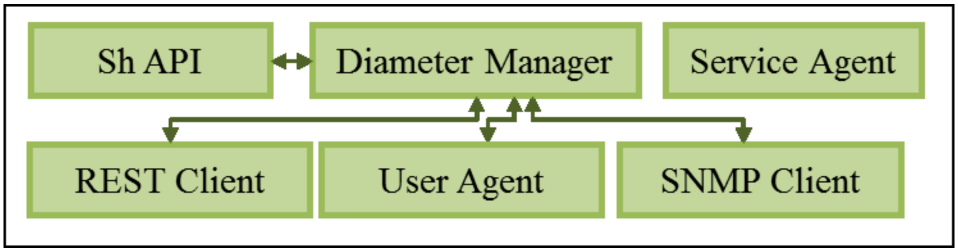

Figure 5.7: The Virtual Diameter entity architecture

Diameter Layer Management:

The Diameter Layer Management consists of two entities (Figure 5.8): SNMP API and Monitor Engine. The SNMP API exposes the entity functionalities via the SNMP interface as well as receiving the requests. The role of the Monitor Engine is to track the load on the layer's entities and to add or remove entities if required.

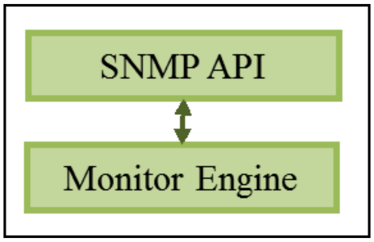

Figure 5.8: The Diameter Layer Management Architecture

Virtual Database Routing Agent:

The Virtual Database Routing Agent is an intermediary component which consists of seven entities (Figure 5.9): RESTful API, Service Agent, Database Protocol Manager, Request Manager, User Agent, SNMP Client and Database Client. The RESTful API exposes the entity functionalities to Virtual Diameter entities via a RESTful interface. The Service Agent is responsible for enabling the components to be discovered by the Virtual Diameter entities. The Request Manager is a core entity that is responsible for processing the request and needs to know the list of available Virtual Database Manager entities. To do this, it uses the User Agent to discover them. The Request Manager then communicates with the Database Layer Management entity via the SNMP Client and requests load information. After receiving this load information as well as the list of available entities, the Request Manager runs a load balance algorithm in order to choose the proper Virtual Database Manager entity for forwarding the request. After this procedure, the Request Manager asks the Database Protocol Convertor to map the REST request to an access mechanism supported by the Database Manager entities in the Database

Management Layer and uses the Database Client to communicate with the Database Manager entities.

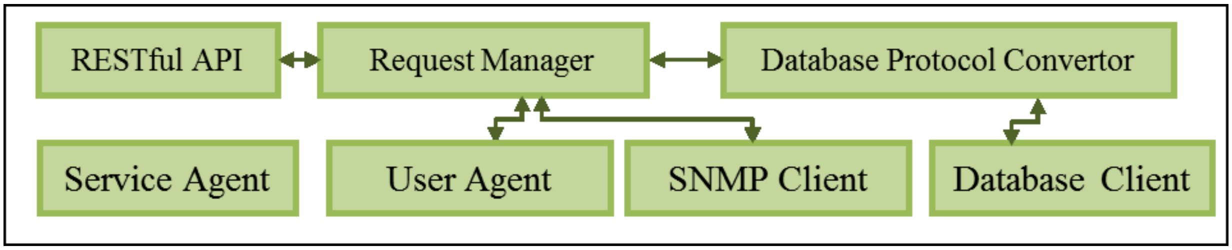

Figure 5.9: The Virtual Database Routing Agent Architecture

5.1.2 Software Operational Procedures

This sub-section studies a scenario to show how QoS Enabled Video Telephony with a VHSS can be can be realized using our implemented architecture. We will show the operational procedures of session creation.

Session Creation: Figure 5.10 shows the first part of the steps in order to create a session. The end-user uses the Video Telephony Application's SIP Servlet to send an INVITE request to initiate a new session. The SIP Servlet forwards the request to the Session Manager who is the core entity. The Session Manager needs to retrieve the user profiles from the VHSS and check if the user is authorized to request the class of service. The Session Manager, with the help of the Diameter Client, sends the UDR to the Virtual Diameter Routing Agent which is the first entry point of the VHSS. The UDR is forwarded to the Load Manager by the Sh API. The Load Manager then asks the User Agent to broadcast the SrvReq to find the list of available Virtual Diameter entities. Following this, the Load Manager communicates with the Diameter Layer Management entity via the SNMP Client to ask for the load information . After these steps have been completed, the Load Manager makes a decision on which nodes are appropriate and should be selected. The Load Manager forwards the UDR request via the Diameter Client. The Diameter entity receives the request by Sh API and sends it to the Diameter Manager for processing (Figure 5.11). In the next step the Diameter Manager needs to know the location of the next hop.

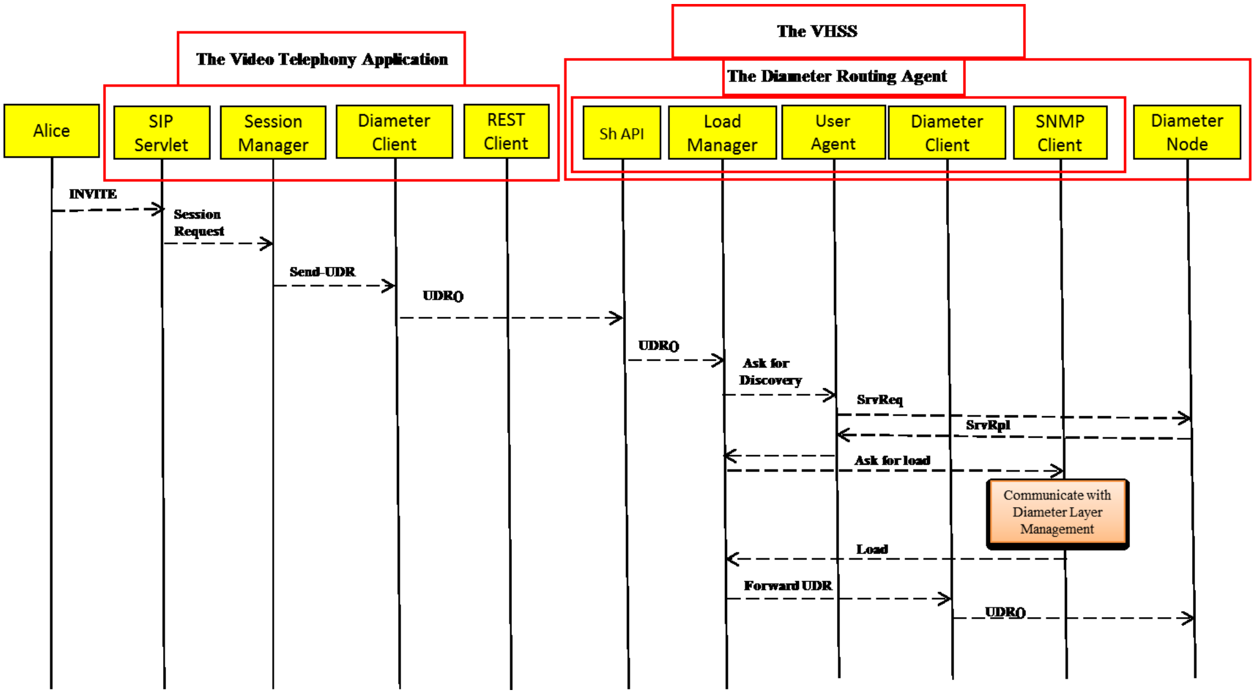

Figure 5.10: Session Creation Steps (Part 1)

For this the User Agent sends the SrvReg to find the available Virtual Database Routing Agents. After that the Diameter Manager creates a GET request and sends it to one of the available Virtual Database Routing Agents.

The RESTful API receives the REST request and forwards it to the Request Manager entity which is responsible for finding the next step and process request. The request is forwarded to the Database Protocol Convertor entity to convert the GET request to the MySQL query (we assume the Database Layer uses MySQL technology). It then forwards the query to the Virtual Database Manager entity which has to retrieve the data from the Storage and then send it back to the upper entities.

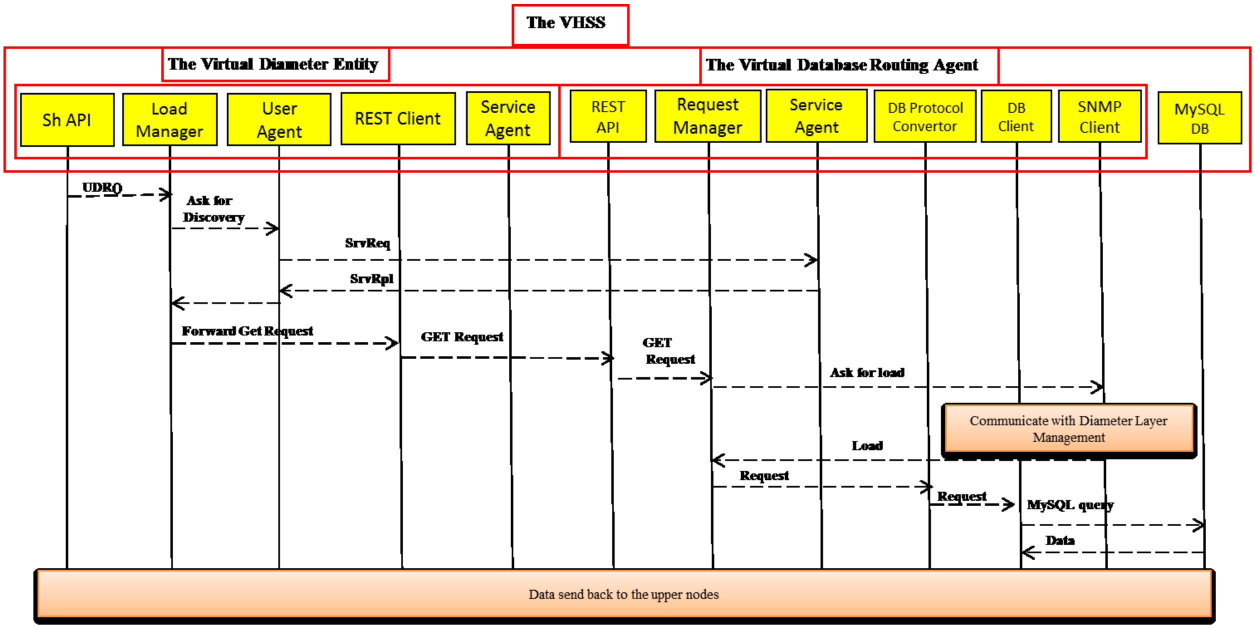

Figure 5.11: Session Creation Steps (Part 2)

The Session Manager extracts the response information from the UDA message and uses the QoS Enabler's REST API to send the POST request for initiating a new session.

When the POST request is received from the REST API (Figure 5.12), the request will be processed by the QoS Manager. The request contains the session information and the desired class of service. The QoS Enabler sends AAR to the PCRF via Rx Diameter Client. The PCRF evaluates the request and responds with the Diameter message AAA. The QoS Manager extracts the AAA message and builds the session request result for the Video Telephony Application. Finally, the Video Telephony Application sends 200-OK to the end-user and the video/audio session is created.

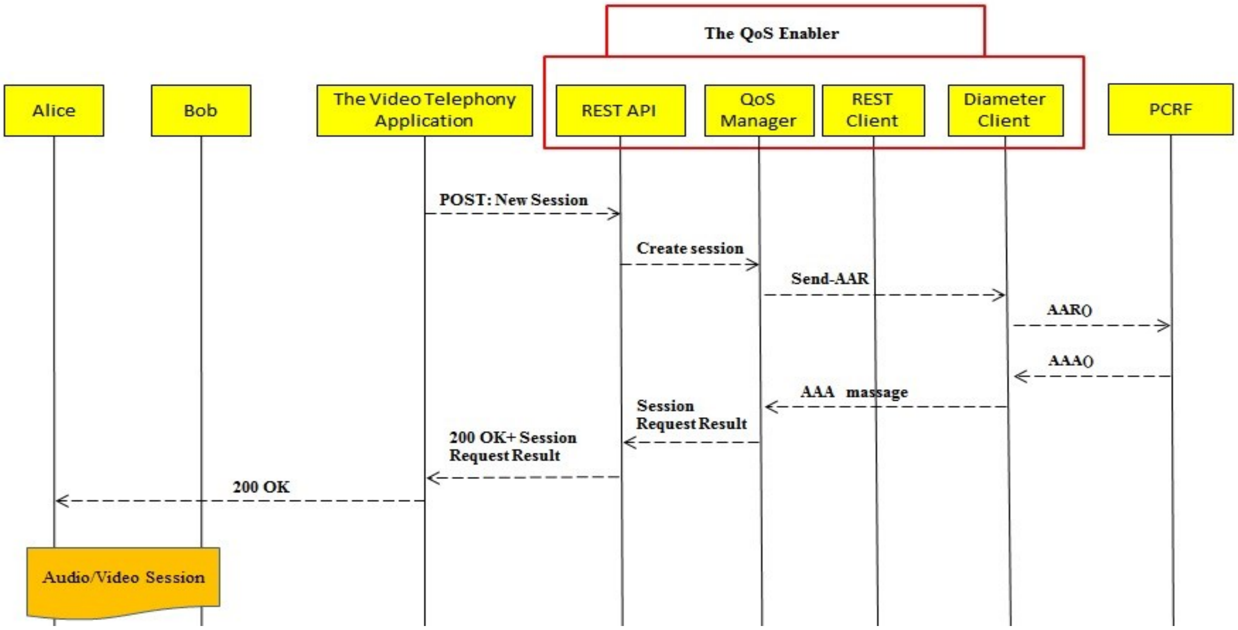

Figure 5.12: Session Creation Steps (Part 3)

5.2 Prototype Design and Implementation

To evaluate our architecture, we implemented a proof of concept prototype for QoS Enabled Video Telephony with a VHSS on EPC. In this section, we discuss the implemented components. Then we present the prototype architecture. After that, we discuss the software tools that are used to implement our prototype. Finally, we discuss the environmental settings.

5.2.1 The implemented components

All components presented in the Video Telephony Service are implemented. As concerns the QoS Service Delivery Platform, only the QoS Enabler is implemented and not the Network Policy Rules. This is because in our scenario presented in Chapter 4 we assumed we had four users with different service providers. However, in order to implement we have to assume that the four users have the same service provider and that this service provider is the one with the lowest priority. This simplifies the prototype implementation since it eliminates the need to implement the Network Policy Rules entity.

Indeed, the output of the Network Policy Rules' execution is known, since it will return the ID of the service provider of the end-users involved.

In the Diameter Layer of the VHSS subsection we implemented the Virtual Diameter Routing Agent and the Virtual Diameter entity. We did not implement the discovery and publication mechanism as we assume it is done. In the Virtual Diameter Routing Agent and the Virtual Diameter entity, we implemented a minor part of the Sh interface (UDR and UDA messages), not all HSS interfaces.

In the Database Management Layer and Storage Layer, we implemented the The Virtual Database Routing Agent and for the rest of the components we used the ScaleDB components as shown in the Figure 5.5. All of the Database Routing Agent resources presented in Table 4-2 were implemented.

5.2.2 Prototype Architecture

Figure 5.13 depicts the prototype architecture. The EPC has a VHSS Enabler to create the VHSS while the QoS Enabler is responsible for enabling differentiated QoS for Video Telephony Service.

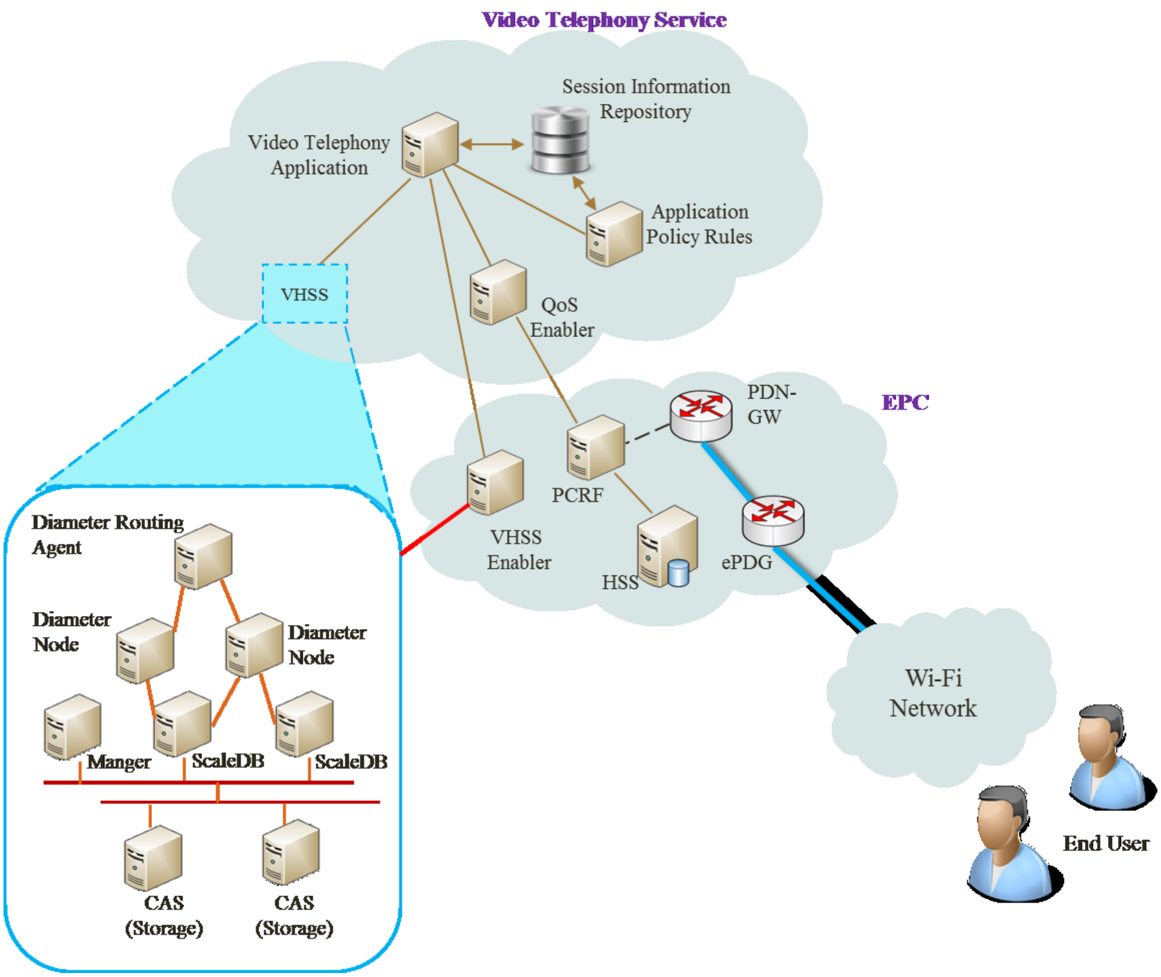

Figure 5.13: The Prototype Architecture

5.2.3 Software Tools

In the following subsections, we discuss the software tools that are used to implement the Video Telephony Service and VHSS prototypes.

5.2.3.1 Fraunhofer Fokus OpenEPC Release 2

We used Fraunhofer Fokus OpenEPC Release 2 [39] as the 3GPP 4G EPC. It is a prototype implementation of the 3GPP EPC which includes a set of software components which conform to 3GPP specifications of interfaces and functionality. It allows academic and industry researchers to have a practical look and feel of the capabilities of the Evolved Packet Core.

5.2.3.2 ScaleDB

We used ScaleDB [32] as the Virtualized Database. It provides a clustered, shared-data storage engine. ScaleDB is a plugable storage engine for MySQL that ensures dynamic scaling of MySQL applications in a public cloud and a private cloud. It consists of at least four virtual machines: ScaleDB Node, Cluster Manager, Mirror Storage and Primary Storage. Each virtual machine runs the Ubuntu operating system. ScaleDB supports dynamic computing and storage scalability, high availability, load balancing and lower total cost ownership. ScaleDB also supports ACID compliance, high-speed indexing and automatic recovery of failed entities. Another advantage of using ScaleDB is its Multi-Table Index that provides the functionality of materialized views without the cost of maintaining the views.

We chose ScaleDB because it covers all of our requirements for Database Virtualization.

5.2.3.3 Additional Software tools

The JAVA programming language is the programming language that is used to implement our prototype. We have used Eclipse IDE [42] for implementing the prototype. The REST interfaces of the QoS Enabler, Video Telephony Service, Application Policy Rules and VHSS Enbaler are implemented with the Restlet framework [43]. The Rx and Sh Diameter Client interface of the QoS Enabler are implemented using the JavaDiameterPeer library [44].

The SIP Servlet in the Video Telephony Application implemented uses the Sailfin [45]. SailFin is an open-source SIP Servlets technology which uses the JSR 289 [46] standard for the SIP-based signaling APIs. SailFin extends the GlassFish application server. As end-users, we use YATE (Yet Another Telephony Engine) [47], an open source SIP client application.

5.2.4 Environment settings

For Video Telephony Service on EPC, we have seven machines, running on a VMware workstation 8 [48]. The first machine runs the Video Telephony Service entities (i.e. Video Telephony Application, Application Policy Rules entity, and Session Information Repository) along with the QoS Enabler. The ePDG, the PDNGw and the four end-users are each running on a separate machine. The first machine is equipped with 2GB of RAM, whereas the other six machines have 512MB of RAM. All of the machines use Ubuntu Release 10.10 as their operating system.

For implementing the VHSS, we have six virtual machines, running on a Xen Server [49] the details of which are shown in table 5.1. All virtual machines have 15GB hard disk space and 1GB of RAM and one VCPU. All of the virtual machines use Ubuntu Release 12.10 as their operating system. The six virtual machines are the Virtual Diameter Routing Agent, the Virtual Diameter entity, the ScaleDB Node (Virtual Database Routing Agent and Virtual Database Manager entity), the Cluster Manager (Database Layer Management entity) and Mirror and Primary CAS (Storage).

The VMware and the VHSS Enabler both are running on an Intel Xeon (R) E5646 2.40 GH Desktop with 8GB of RAM and Windows 7. The VHSS Enabler connects to the Xen Server and controls the virtual machines.

Xen Server Version	5.6 Feature Pack 1
Edition	Citrix XenServer
CPU 0 – 11	Intel(R) Xeon(R) CPUSpeed: 2532 MHz
Interfaces NIC 0 – 3	Speed 1000 Mbit/s
Server Memory	Speed 1000 Mbit/s
Local Storage	600 GB

Table 5-1: The Xen Server Details

5.3 Preliminary Validation and Performance Evaluation

5.3.1 Testing Scenario

As a testing scenario, we have four users: Alice, Bob, Emma and Charlie. Alice’s and Bob’s accounts are configured to use the “Telephony Silver” class of service while Emma’s and Charlie’s accounts are configured to use the “Telephony Gold” class of service.

We assume that the four users have the same service provider and that this service provider is the one with the lowest priority. We initiate the call session between Alice and Bob. During the session Emma calls Charlie but we assume that we do not have enough resources and the application sends BYE to Alice and Bob to terminate the session and retries to create a session for Emma and Bob. At this time we have enough resources and a session is created. We test this scenario in two conditions when the Video Telephony Service uses the real HSS and when it uses the VHSS. Figure 5.14 shows the implementation sequence diagram .

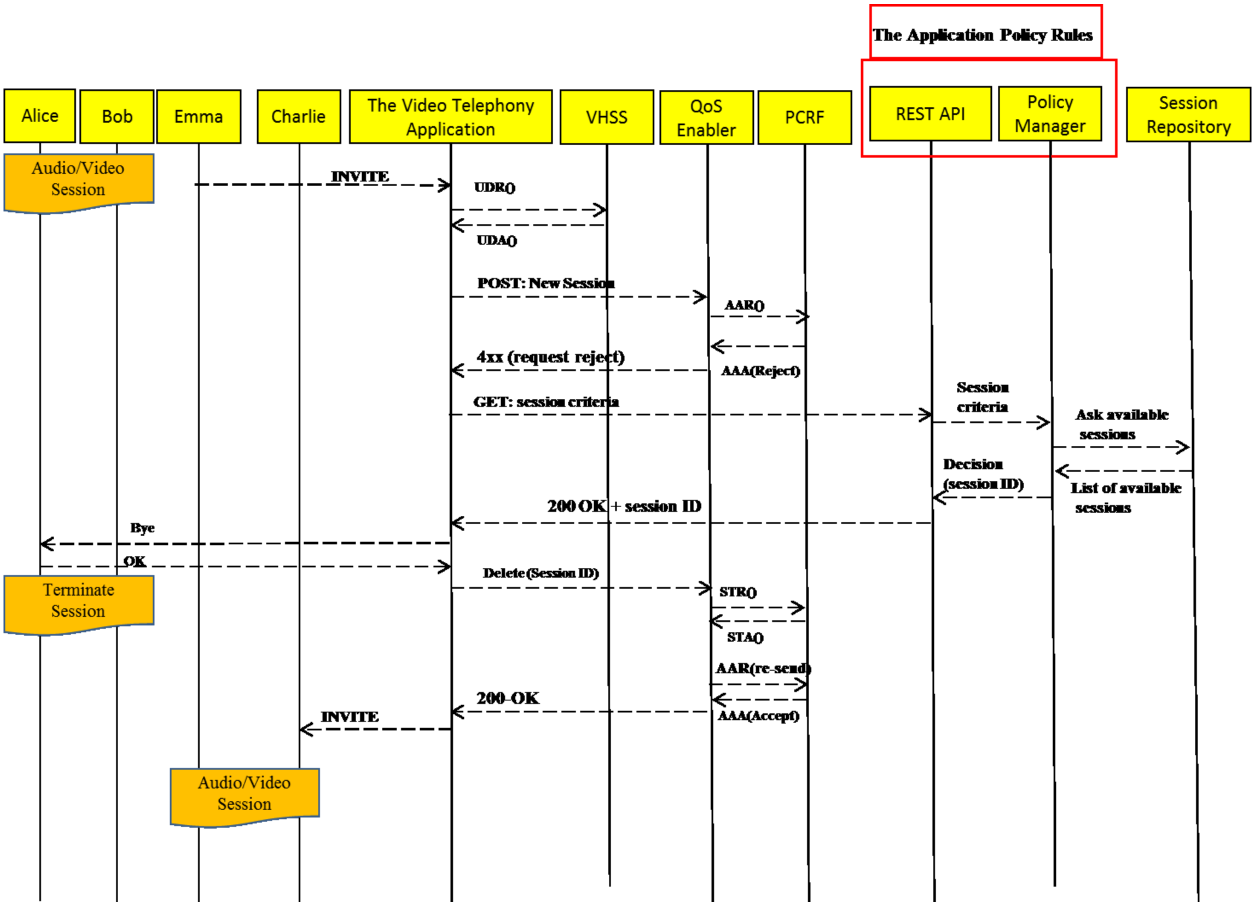

Figure 5.14: The implementation sequence diagram

5.3.2 Measurements and analysis

This subsection presents the performance metrics used to evaluate the prototype and analyzes the collected measurements.

5.3.2.1 Performance metrics

The performance of the prototype is assessed in terms of session establishment. The session establishment delay is the end-to-end time delay needed to establish a new session with a specific QoS profile. Essentially it is the difference between the time when an end-user sends a SIP INVITE to the Video Telephony Application and the time when the session is established with the callee. This includes the communication with the PCRF, the selection of the appropriate sessions to be modified, and the modification of such sessions. We also compare the session establishment delays between the cases where the resources are available and when the resource release is needed. Additionally, we will compare the session establishment delays between the two different cases:

- *Case 1*: The video telephony providers use the non-virtualized HSS
- *Case 2*: The video telephony providers use the VHSS

5.3.2.2 Performance measurements and analysis

Figure 5.15 and Figure 5.16 illustrate the measured results. Corresponding to each of these measured results, Table 5.2 and Table 5.3 list the recorded time in milliseconds. Each result is calculated as an average of 10 measurements. Figure 5.15 displays the session establishment delay when the Video Telephony Application Service uses the real HSS (Case 1).

Figure 5.16 displays the session establishment delay when the video telephony application service uses the VHSS (Case 2).

The delays for the session establishment when the resources are available are measured for the establishment of the first session between the Silver end-users. These delays have an average of 221.6 ms in case-1 and 485.6 in case-2 . They are barely observable by the end-users.

As shown in the figure, the session establishment delays when resource release is needed (i.e. for the second session between the Gold end-users) are higher than when the resources are available. These longer delays occur because of the extra delay for session modification (i.e. Session termination in the context of the prototype).

The comparison of the two cases shows the time has doubled. However, in reality it will be much more than double because we did not implement some parts of the architecture which could eventually add overheads. We did not implement the Publication Discovery mechanism, the Load Balancing algorithm and the Diameter Layer Management entity .

The doubling of delay time is obvious because in Case 1 we have just a single node as HSS but in Case 2 we have three layers and distributed nodes which are responsible for undertaking the job of the single HSS. For Case 2, the delay incurred for the session establishment is 485.6 ms on average. Approximately 50% of this delay is due to the REST messages exchanged between the two layers of the VHSS and communication between the Diameter Routing Agent and the Diameter node.

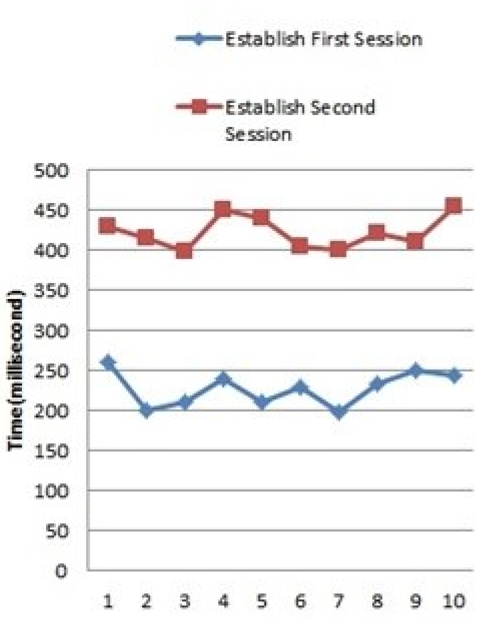

Figure 5.15: Performance measurement 1

	First Session (Silver Session)	Second Session (Gold Session)
1	335	518
2	284	519
3	311	544
4	300	489
5	290	510
6	320	520
7	285	517
8	308	500
9	327	508
10	297	516

Table 5-2: Performance measurement

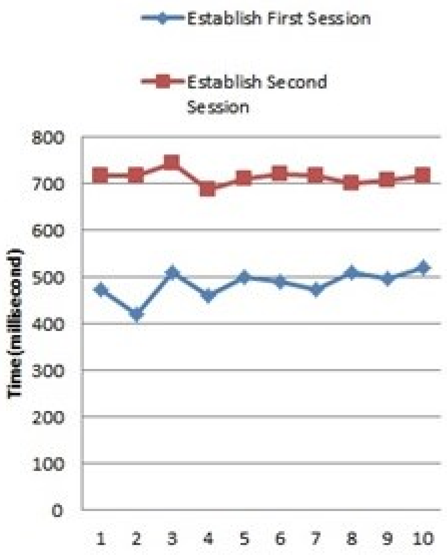

Figure 5.16: Performance measurement 2

	First Session (Silver Session)	Second Session (Gold Session)
1	473	718
2	420	719
3	510	744
4	460	689
5	500	710
6	490	720
7	473	717
8	511	700
9	499	708
10	520	716

Table 5-3: Performance measurement

5.4 Chapter Summary

In this chapter, we have introduced the software architecture for all functional entities in our proposed architecture as well as the operational procedures. Then we have presented the proof of concept prototype. We have built the prototype with the Fraunhofer Fokus OpenEPC as 3GPP 4G EPC infrastructure and ScaleDB as a Database Virtualization. Some measurements have been collected to validate our proposed architecture. We have also presented the experimental scenario that is used to take these measurements. The measurements show that the proposed architecture is feasible and although the delay has been approximately doubled, there is an opportunity in the future to work on optimizing the delay time. In the next chapter, we will summarize the contribution of the thesis and propose some additional future works.

Chapter 6

6 Conclusion and Future Work

In this chapter, we will first summarize the contributions of the thesis and then we will give some ideas for future work.

6.1 Summary of the Contributions

Video Telephony is the real time exchange of voice and video between end-users. It has become ubiquitous and it is the basis of a wide range of applications (e.g. multiparty games, distance learning). EPC is the new core network for 3GPP 4G networks. Home Subscriber Server (HSS) is the standardized master database of 3GPP next generation networks including video telephony networks and EPC.

As one of the contributions of this thesis, the requirements we have identified fall into three categories: general requirements, requirements for the QoS Enabled Video Telephony Application and the requirements of HSS Virtualization. Moreover, we have reviewed the most relevant related works. We have divided these related works into two categories: the guaranteed and differentiated QoS in Video Telephony Applications and HSS Virtualization.

Afterwards, we have evaluated these related works based on our requirements; we have observed that none of them meet all of our requirements. Then, we proposed a novel architecture for QoS Enabled Video Telephony with a VHSS. Our architecture fulfills the requirements described before and supports guaranteed and differentiated QoS and HSS Virtualization. Furthermore, the proposed architecture contains a new Differentiated QoS Service Delivery Platform that enables guaranteed and differentiated QoS which includes the QoS Enabler and the Network Policy Rules. The QoS Enabler provides the ability to support differentiated QoS. Moreover, the proposed architecture also contains a proposed mechanism for a scalable and elastic HSS. We propose a new preliminary architecture for VHSS which covers our requirements.

A proof of concept prototype has been implemented using the Fraunhofer Fokus OpenEPC [41] as 3GPP 4G EPC infrastructure and ScaleDB [32] as Virtualized Database. The prototype includes a SIP client that is using YATE [47]. This client allows end-users to communicate with Video Telephony Service for having video sessions.

Lastly, to validate our prototype, a preliminary performance evaluation of the overall architecture has been taken. Based on these results, we conclude that our architecture is a valid and promising approach for QoS Enabled Video Telephony Applications with VHSS over EPC.

6.2 Future Work

This thesis has solved most of the issues related to the QoS Enabled Video Telephony Application. However, it has left the several key issues related to the VHSS open to further studies. These issues are outlined below.

The first involves the optimization of the architecture so that performance can be improved. One way of improving performance could be bypassing the discovery procedure by pre-configuring the routing agents for some applications that require faster access.

For a more comprehensive performance analysis, full implementation is required. In our preliminary proposed architecture several parts have not been implemented. Therefore, future work to improve performance analysis could include implementing the parts below which were not implemented in our preliminary proposed architecture.

Discovery and publication functions which are needed for finding available nodes or agents in each layer could be implemented. As we have mentioned, we have two ways of knowing the available nodes or agents. Either configure manually or discover the nodes automatically.

How to support and implement the SRVLOC publication and discovery mechanism can also be examined. Optimization of the delay time mentioned in the Evaluation Performance section in Chapter 5 can also be worked on in the future.

It is important to note we did not virtualize everything in our proposed VHSS. For example, the interfaces between layers and entities in each layer were not virtualized. In addition, the first layer is the Diameter Layer which includes several virtual nodes that run the Diameter Server. For virtualizing this layer we only ran Diameter in the virtual machine instead of virtualizing in the appropriate way such as decoupling the Diameter server. However, this solution is not optimal as more layers can reduce performance. Additional research is needed to find a solution that can virtualize the Diameter server while not affecting performance negatively.

Furthermore, the proposed architecture for VHSS needs two types of algorithms. First, an algorithm is needed to select the most appropriate node (Diameter Node or Database Node) for a given request. The algorithms are actually a load balancing for choosing the suitable node.

Secondly, the preliminary algorithm for growing or shrinking nodes in each layer mentioned earlier requires further research. Based on our evaluation with some of the existing algorithms, this algorithm needs to be enhanced.

Bibliography

[1] M. Olsson et al. , SAE and Evolved Packet Core: Driving the Mobile Broadband Revolution, Elsevier, Second Edition 2012

[2] 3GPP TS 23.401, "LTE; General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access," version 10.7.0, 2012.

[3] 3GPP TS 23.203, "Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; Policy and charging control architecture," version 10.6.0, 2012.

[4] 3GPP TS 29.336," Home Subscriber Server (HSS) diameter interfaces for interworking with packet data networks and applications" version 12.1.0, 2013.

[5] RFC 3588 Diameter , Based Protocol , September 2003

[6] J. Gozdecki, A.Jajszczyk, and R. Stankiewicz, "Quality of Service Terminology in IP Networks",IEEE Communication Magazine , Volume 41 , Issue 3

[7] Z. Wang, Internet QoS: Architectures and mechanisms for quality of service. Morgan Kaufmann Publishers, 2001.

[8] H. Ekstrom , "QoSControl in the 3GPP Evolved Packet System",IEEE Communication Magazine , 2009

[9] J. Carapinha, J. Jiménez, "Network Virtualization – a View from the Bottom", the 1st ACM workshop on Virtualized infrastructure systems and architectures, 2009

[10] N.M. Chowdhury and R. Boothbay," Network Virtualization: State of the Art and Research Challenges", IEEE Communications Magazine, 2009

[11] R. Buyya, C. Vecchiola and S. T. Selvi ,Chapter 3 –Virtualization , “Mastering Cloud Computing Foundations and Applications Programming ” Mastering Cloud Computing, 2013, Pages 71-109

[12] B. Prakash, R. Tholeti, Chapter 16 –Hypervisor,Virtualization and Networking , Handbook of Fiber Optic Data Communication (Fourth Edition), 2013, Pages 387-416

[13] A. Khan, A. Zugenmaier, D. Jurca and W. Kellerer,” Network Virtualization :A Hypervisor for the Internet?” , Communication Magazine, IEEE , Volume 50 ,Issue 1 , January 2012 , Pages 136 - 143

[14] P. Barham, B. Ragovic, K. Fraser, S. Hand ,” Xen and the Art of Virtualization” , SOSP '03 Proceedings of the nineteenth ACM symposium on Operating systems principles, Pages 164-177

[15] N.M Chowdhury ,R. Boutaba ,” A Survey of Network Virtualization” , Computer Networks: The International Journal of Computer and Telecommunications Networking 2010

[16] Andy Oppel , Databases DeMYSTiFieD , 2th Edition

[17] Elmasri, R. and Navathe, S. 2011. Fundamentals of Database Systems, Addison-Wesley.

[18] S. Lee ,”Shared-Nothing vs. Shared-Disk Cloud database Architecture”, International Journal of Energy, Information and Communications Vol. 2, Issue 4, November, 2011

[19] Z. Parker, S. Poe, S.V. Vrbsky, “ Comparing NoSQL MongoDB to an SQL DB” , Proceedings of the 51st ACM Southeast Conference , April 2013

[20] T. Kiefer, W. Lehner ,”Private Table Database Virtualization for DBaaS” , 4th IEEE International Conference on Utility and Cloud Computing, 2011

[21] A. A. Soror , A. Aboulnaga , K. Salem, “Database Virtualization: A New Frontier for Database Tuning and Physical Design “ , 23rd IEEE Data Engineering Workshop ,2007

[22] S Blake et al., An Architecture for Differentiated Services, IETF, RFC 2475

[23] D. Wang et al,, Video Telephony over Downlink LTE with/without QoS provisioning, Sarnoff Symposium 2011

[24] S. Dutt et al., A Novel Optimized Scheduler to Provide QoS for Video IP Telephony over Wireless Networks, High Performance Computing and Communications (HPCC) 2011

[25] M. El Barachi, R. Glitho, R. Dssouli, Control Level Call Differentiation in IMS-Based 3G Core Networks, IEEE Network, January/February 2011, Vol. 25, No1

[26] M Abu-Lebdeh, F Belqasmi, R Glitho, "A 3GPP 4G Evolved Packet Core Based – Architecture for QoS Enabled Mobile Video Surveillance Application", Third International Conference on the Network of the Future, November 2012, Tunis

[27] R. Ben Ali , S. Pierre, École Polytechnique de Montréal Yves Lemieux, Ericsson Research Canada, "UMTS-to-IP QoS Mapping for Voice and Video Telephony Services" IEEE Network • March/April 2005

[28] Tianpu Yang, Xiangming Wen, Yong Sun, Zhenmin Zhao, Yuedui Wang "A New Architecture of HSS Based on Cloud Computing" **.** Communication Technology (ICCT), 2011 IEEE 13th Int. Conf.

[29] P. Bosch , A. Duminuco, F. Pianese , " Telco Clouds and Virtual Telco: Consolidation , Convergence and Beyond", 6th IFIP/IEEE International Workshop on Broadband Convergence Networks , 2011

[30] X. AN , F. Pianese, I. Widjaja , "DMME: Virtualizing LTE Mobility Managment", 36th Annual IEEE Conference on Local Computer Networks, 2011

[31] S. Das, D. Agrawal, A. El Abbadi ," Elastras: An elastic , Scalable, and Self Managing Transactional Database in the Cloud", UCSB Computer Science Technical Report 2010-04

[32] "Database Virtualization (The Next Wave of Virtualization)" [Online]. Available : http://www.scaledb.com

[33] 3GPP TS 23.335, "User Data Convergence (UDC)" version 11.0.0, 2012

[34] R. Shea, J. Liu," Network Interface Virtualization: Challenges and Solutions", IEEE Network 2012

[35] F. Belqasmi, R. Glitho, C. Fu, "RESTful web services for service provisioning in next-generation networks: a survey", Communications Magazine, IEEE, vol. 49, no. 12, pp. 66–73, 2011.

[36] M. Kulkarni et al., SIP Servlet Specification, version 1.1, JSR 289 Expert Group, Aug. 2008.

[37] 3GPP TS 29.329, "Sh interface based on the Diameter protocol; Protocol details" version 12.1.0, 2013

[38] RFC 3083, "Notification and Subscription for SLP" , March 2001

[39] RFC 1157 ," Simple Network Management Protocol (SNMP)" May 1990

[40] D. Comer, "Ubiquitous B-Tree", Journal ACM Computing Surveys (CSUR), Volume 11 Issue 2, June 1979

[41] Fraunhofer Fokus OpenEPC; available on web at: http://www.openepc.net/index.html

[42] "Eclipse IDE." [Online]. Available: http://www.eclipse.org/

[43] "RestLet Features." [Online]. Available: http://wiki.restlet.org/docs_2.1/13-restlet/21-restlet/22-restlet.html.

[44] "JavaDiameterPeer Library." [Online]. Available: http://www.openimscore.org/project/jdp.

[45] "Sailfin" [Online]. Available: https://sailfin.java.net/

[46] M. Kulkarni et al; SIP Servlet Specification, version 1.1; JSR 289 Expert Group ; August 2008.

[47] "YATE" [Online]. Available: http://yate.null.ro/pmwiki/

[48] "VMware Workstation" [Online]. Available: http://www.vmware.com/products/workstation/overview.html. [Accessed: 11-May-2012].

[49] "XenSource Inc, Xen" [Online]. Available: http://www.xensource.com/.