Master in Informatics Engineering
Internship
Final Report

Visual Advertising in Voice-over-IP calls

David Joaquim Monteiro Colaço Salvador

djcs@student.dei.uc.pt

DEI Supervisor:

Prof. Dr. Luís Cordeiro

Wit-Software Supervisor:

Eng. Rafael Maia

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Abstract

Nowadays, with the spread of Smartphones and Internet worldwide, a new way of communicate has arrived. A telephone is no longer an instrument to make calls. It is used for work, play or for fast access to news and other media content.

With the increase speed of the Internet in mobile devices, some companies saw an opportunity to develop successful applications, known as Over-The-Top applications. Some of these OTTs offer the same products as the telecommunications companies at a lower price, or even for free. This lead to two major problems: a decrease in revenue on the Telecommunications market, and a demand for faster access to the internet.

A response to OTTs emerged from the Telecommunications world with the creation of the Rich Communications Services (RCS). This specification provides the user new ways to communicate and share content.

It was also clear to the telecommunications industry that an improvement to the network was needed and the migration from Circuit Switch to all IP networks was essential. From this need, the 3GPP group created the IP Multimedia Subsystem (IMS). This network architecture can deliver services and multimedia content using industry standards and provide compatibility across all operators and legacy networks.

On this project, a new service is added to the IMS network, in order to deliver image and video advertisement to RCS clients.

Keywords

"Advertisement", "Image Share", "IMS", "IP Multimedia Subsystem", "Media Server", "MSRP", "RCS", "Rich Communication Services", "Rich Communication Suite", "SIP", "Video Share", "

Table of Content

List of Figures

List of Tables

Acronyms

3GPP	3rd Generation Partnership Project
ACID	Atomicity, consistency, isolation, durability
AS	Application Server
ATM	Asynchronous Transfer Mode
B-ISDN	Broadband Integrated Services Digital Network
CSCF	Call Session Control Function
GSM	Global System for Mobile Communications
GSMA	GSM Association
HSS	Home Subscriber Server
I-CSCF	Interrogating Call Session Control Function
IFC	Initial Filter criteria
IMS	IP Multimedia Subsystem
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISDN	Integrated Service Digital Network
ISP	Internet Service Provider
LTE	Long Term Evolution
MDCP	Media Device Control Protocol
MGCP	Media Gateway Control Protocol
MRFC	Media Resource Function Controller
MRFP	Media Resource Function Processor
MSRP	Message Session Relay Protocol
OO	Object Orientated
OTT	Over-The-Top
P-CSCF	Proxy Call Session Control Function
P2P	Peer-to-Peer
PSTN	Public Switched Telephone Network
RCS	Rich Communication Services

RTP	Real-Time Transport Protocol
S-CSCF	Serving Call Session Control Function
SBB	Service Building Block
SDP	Session Description Protocol
SGCP	Simple Gateway Control Protocol
SIP	Session Initiation Protocol
SIP-AS	SIP Application Server
UE	User Endpoint
UMTS	Universal Mobile Telecommunications System
VAS	Value Added Services
VoIP	Voice Over IP
WiMAX	World Interoperability for Microwave Access
XML	eXtensible Markup Language
XMPP	Extensible Messaging and Presence Protocol

Chapter 1
Introduction

This document describes the work done by the author during the academic year of 2013/2014. The work was developed within the scope of the Masters on Informatics Engineering on the Department of Informatics Engineering of the Faculty of Science and Technology of the University of Coimbra (DEI FCT UC), as part as the Thesis/Internship discipline.

The Internship took place on the company Wit-Software S.A., and has the supervisors:

- Dr. Luís Cordeiro – DEI FCT UC
- Eng. Rafael Maia – WIT-Software S.A.

1.1 Wit-Software S.A.

Wit-Software S.A. was formed in the year of 2001, from a spinoff that started on the University of Coimbra (UC), with the support of the Pedro Nune's Institute (IPN).

Wit-Software S.A. is a software development company specialized in rich and unified communications for Mobile Operators and Mobile Internet companies. It also has departments on the Mobile Applications and IPTV area.

1.2 Document Structure

This document is divided into seven chapters and three Annex documents.

The First chapter is a simple **Introduction** about the internship. It also explains the context and motivation in which the internship is inserted. Then, the main objectives of the project are detailed, and in the final section of this chapter, the original and actual planning is presented.

In the Second chapter, **State of the Art**, the author analyzes the available products on the market, how they threat the Telecommunications Industry, and what was their response. Later in this chapter, a comparison between the available technological solutions that might be used in the context of this project is presented.

The Third chapter, **Technology Background**, is a chapter that introduces all the necessary technologies that the author needed to know before starting the development of the project.

After all the background is presented, the actual system developed is detailed in the fourth chapter, **System Description**. In this chapter, the system is described, the use cases are explained and a high level architecture is introduced. It is also explained what is the software development methodology that was used during the development.

The next chapter is dedicated to a more technical point of view of the entire system. In the fifth chapter, **Development**, the author details the implementation and technologies integration used during the Internship.

In the sixth Chapter, **Requirements Validation**, the author explains which tests were made during and after the development of the Advertisement System. By analysing this battery of tests, the reader can easily understand which requirements (functional and non-functional) were achieved.

In the seventh and last chapter, **Conclusions**, an analyse is made on what are the strong and risk factors of the project, what was achieved and what wasn't achieved during the internship, and what is planned to be done as future work.

Whenever more detailed information about a subject is required, it is detailed in an Annex document.

1.3 Context

Since the beginning of human kind, people have tried to communicate at a distance, mostly using drums, smoke signals and beacons. By the end of the XVIII century, in France, the Chappe brothers invented the semaphore system, which consisted of moveable arms on a pole, whose position denoted letters of the alphabet. However, it was not until 1830s that electrical telecommunication systems started to appear, with the introduction of the Telegraph, and in the 1870s, the Telephone [1]. Soon, multiple telecommunications companies were created, and the way we communicate never stopped evolving. But in the last two decades, with the increasing growth of the Internet and Mobile devices, people started to change the way they interact with each other, and how multimedia content is consumed. Since the access, reliability and the bandwidth of the Internet has been improving over the years, providing new ways for people to communicate, a shift was generated on the paradigm of the telecommunications industry, creating the need to develop new ways of delivering content and services.

In 1995 an association of mobile operators, handset makers, software companies and Internet companies was created with the name GSM Association (GSMA) [2]. The main goal of this group was to support the standardising, deployment and promotion of the Global System for Mobile Communications (GSM) mobile telephone system. In 1998, another group was formed, 3rd Generation Partnership Project (3GPP) [3], with the main focus of developing and standardizing new, faster and reliable ways to deliver content and services over IP, maintaining compatibility across the different operators and previous technologies. It's within this need, that in 2002, in the document 3GPP Release 5 (evolution from 2G to 3G networks) the IP Multimedia Subsystem (IMS) was specified. The specification describes what kinds of components are necessary, their functionality, the interfaces connecting each component, and the flow of communication between them. This network architecture uses Internet Engineering Task Force (IETF) standards whenever possible (e.g. SIP protocol) allowing telecommunication operators to offer new services to their customers using known and established communication protocols. Another key feature of IMS is that it allows applications to be usable across different types of devices (PC, mobile devices, television) and networks (wireless, cable, GSM, SS7).

In the past few years, another thread emerged against Telecommunications operators. Due to the fast evolution of the Internet and the arrival of smartphones, an increasing number of applications and services that provide online content have invaded the telephony, broadband and TV industry. These services are known as over-the-top (OTT) [4]. A fundamental characteristic of OTT's is that they bypass traditional distribution, so, the Internet Service Provider (ISP), or mobile telephony companies does not profit nor are involved in the

distribution of the OTT applications, services and content. The best known examples of OTT services are Skype, Whatsapp, online video games and movies (Netflix, Pandora).

On the attempt of reversing the revenue losses caused by OTT applications, the GSMA group released the Rich Communication Services (RCS) specification (originally known as Rich Communication Suite) [5]. This specifications, defined a new way for integrating a set of features that provides functionalities like image and video share, combine SMS with instant messaging or chat, video calls and file transfer across all devices and networks.

1.4 Objectives

The main goal of this project is to specify the procedures, and describe the implementation of an Image and Video **Advertisement Service**, which acts as a **SIP Application Server (SIP AS)** on the IMS network.

This service will allow any RCS client on the network to receive Image and Video advertisement. There will be 3 different stages in which a user might receive advertisement:

- **Before a call** – When user A calls user B, the call is "answered" by the Advertisement service, which in turn, will start an image or video share with user A. After the media content is transferred, the Advertisement server will transfer the call to user B.
- **During call setup** – User A will receive video or image advertisement during the "Ringing" stage of the call, as happens nowadays with Ringtones.
- **Ongoing call** – User A e B are in a call. If they are not exchanging media content (image/video share) the advertisement service will push media content to both.

More detailed information about the use-cases is presented later in Chapter 4 – System Description.

Although the main focus of the system is to distribute adverts, this service must never interfere with the capabilities available to the user, namely, if users are sharing a video file, no advertisement can be sent to them. It is the responsibility of the AS to detect when and for whom to send advertisement.

Another objective of this project is to develop a simple **BackOffice (BO)** in order to manage advertisement campaigns, as well as provide information to entities that submit ads.

It's noteworthy that this project is a **proof of concept (POC)**, so, some aspects of real user experience, like the use of sound in video ads, or the number of ads in a call, might be ignored.

All technologies used must be free or Open-source.

From a personal point of view, this project will allow the author to acquire important new technical knowledge in new fields such as IMS and RCS. It will also allow improving skills acquired in the past, and to gain experience working with Agile Methodologies in an enterprise environment.

1.5 Planning

1.5.1 Original

The original planning was made by the company supervisor (Eng. Rafael Maia), as reference to what work should be done at a specific time of the internship. It is divided in two phases, corresponding respectively to the first and second semester of the internship.

The main focus of the first phase is for the intern to study and fully understand the technologies that are going to be used. Then the intern should configure the development environment, and do some simple tests. By the end of this phase, the intern will start to implement some of the use cases of the project (image share).

In the second phase, or second semester, the intern will implement the rest of the image share use cases, deploy the media gateway, and implement all of the video sharing use cases. Then the intern will develop a simple back-office for management of the advertisement campaigns. By the end of the second phase, the intern will have some time to write the report and realize all the necessary tests.

First Semester

ID	Task Name	Start	Finish	Duration
1	Project kick-off	02/09/2013	06-09-2013	5d
2	Configuring OpenIMS environment	09-09-2013	20-09-2013	10d
3	Creating accounts, iFCs and configuring clients	23-09-2013	04-10-2013	10d
4	First sample Application Server	07-10-2013	11-10-2013	5d
5	Detailed logging system	14-10-2013	25-10-2013	10d
6	Start using WIT RCS clients and study network traces and logs	28-10-2013	01-11-2013	5d
7	Capability discovery mechanism	04-11-2013	15-11-2013	10d
8	Integrate 3rd party MSRP stack with Application Server	18-11-2013	22-11-2013	5d
9	Use Case 5: Originating user receives 3rd party image share during on-going call	25-11-2013	06-12-2013	10d
10	Use Case 7: Terminating user receives 3rd party image share during on-going call	09-12-2013	13-12-2013	5d
11	Test and debug session	16-12-2013	20-12-2013	5d
12	Miliestone 1: First Advertisement Application Server	30/12/2013	30/12/2013	0d
13	Christmas vacation	23-12-2013	27-12-2013	5d
14	Work on the internship report	30-12-2013	10-01-2014	10d
15	Preparing internship review meetings	13-01-2014	24-01-2014	10d
16	Phase 1: lessons learned	27-01-2014	31-01-2014	5d

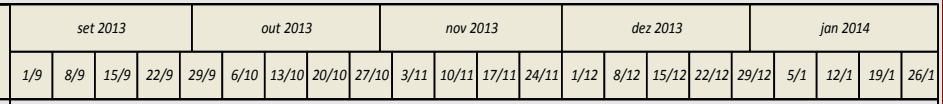

Figure 1 – Original plan first semester

Second Semester

ID	Task Name	Start	Finish	Duration
1	Use Case 3: Originating user received 3rd party image share during call setup	03/02/2014	07/02/2014	5d
2	Use Case 1: Originating user receives 3rd party im. share before call is established	10/02/2014	14/02/2014	5d
3	Back-office to manage image share advertisement campaigns	17/02/2014	28/02/2014	10d
4	Test and debug session	03/03/2014	07/03/2014	5d
5	Milestone 2: Image Share Advertisement Application Server	07/03/2014	07/03/2014	0d
6	Integrate 3rd party MRFP node with OpenIMS environment	10-03-2014	21-03-2014	10d
7	Use Case 6: Originating user receives 3rd party video share during on-going call	24-03-2014	04-04-2014	10d
8	Use Case 8: Terminating user receives 3rd party video share during on-going call	07-04-2014	11-04-2014	5d
9	Easter vacation	14-04-2014	18-04-2014	5d
10	Use Case 4: Originating user receives 3rd party video share during call setup	21-04-2014	25-04-2014	5d
11	Use Case 2: Orig. user receives 3rd party video share before call is established	28-04-2014	02-05-2014	5d
12	Improve back-office to manage video share advertisement campaigns	05-05-2014	16-05-2014	10d
13	Improve back-office with user accounts and login mechanism	19-05-2014	23-05-2014	5d
14	Improve back-office with usage statiscs module	26-05-2014	30-05-2014	5d
15	Test and debug session	02-06-2014	06-06-2014	5d
16	Milestone 3: Final product release	06-06-2014	06/06/2014	0d
17	Work on the internship report	09-06-2014	20-06-2014	10d
18	Prepare internship presentation	23-06-2014	27-06-2014	5d
19	Preparing internship review meetings	30-06-2014	11-07-2014	10d
20	Phase 2: lessons learned	14-07-2014	18-07-2014	5d
21	Internship conclusion	18-07-2014	18/07/2014	0d

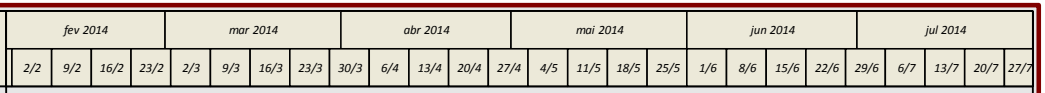

Figure 2 - Original plan Second Semester

1.5.2 Actual

First Semester

During the first semester, due to some external factors, which are detailed in Chapter 7 Conclusions, it was not possible to follow the original planning. However, both the intern and the Company Supervisor (Eng. Rafael Maia) think that some time was recovered, and with some extra work, most of the project objectives for the first semester were achieved.

Also, the author realizes that the technical difficulties faced during the first semester were extremely important because they allowed a more profound knowledge of some of the technologies that were used, like IMS network architecture and the Company RCS Client. This knowledge was a big advantage in the second semester.

Figure 3 illustrates the actual plan that was realized in the first semester of the internship. Some objectives that were in the original plan were achieved, like the technological study, and the installation and configuration of the environment. Then some tests were realized in order for the author to fully understand the concepts and acquire the technical knowledge necessary to develop the use cases.

Second Semester

In the second part of the Internship, the main focus was development of the Advertisement Service. During the Development, the author faced several unexpected difficulties that delayed the schedule that was initially planned.

As soon as the author started the development of the Image Share, he realized that the library available did not hold a key feature for this process, Chunking. This feature had to be developed by the author. This reason delayed the following tasks.

When the author finished the development of the Image Share Use cases, he faced another unexpected delay when installing the Media Server. This process should be trivial; however, due to the fact that this media server was not developed to work with JBoss, it took the author some time to do it.

Nevertheless, the author realizes that these unexpected challenges are part of the work of a Software Developer, and were important for him to learn how to work under pressure and with deadlines.

First Semester

ID	Task Name	Start	Finish	Duration
1	Project Presentation	16-09-2013	20-09-2013	5d
2	Technology study	23-09-2013	11-10-2013	15d
3	SIP Traces study	14-10-2013	18-10-2013	5d
4	Company entrance and integration	21-10-2013	23-10-2013	3d
5	Configure OpenIMS environment	24-10-2013	29-10-2013	4d
6	Sample Application Server and Tests	30-10-2013	05-11-2013	5d
7	Capability discovery mechanism	06-11-2013	22-11-2013	13d
8	Logging System	25-11-2013	29-11-2013	5d
9	Report: State of the Art	02-12-2013	06-12-2013	5d
10	Integrate 3rd Party MSRP Stack	09-12-2013	13-12-2013	5d
11	WIT RCS client (joyn) tests and config	16-12-2013	20-12-2013	5d
12	AS tests and debug	23-12-2013	24-12-2013	2d
13	Report: State of the Art	26-12-2013	03-01-2014	7d
14	RCS client integration with OpenIMS	06-01-2014	17-01-2014	10d
15	Work on Internship Report	20-01-2014	28-01-2014	7d
16	Internship Presentation	29-01-2014	04-02-2014	5d
17	Miliestone 1: Internship Intermediate Presentation	05-02-2014	05-02-2014	0d

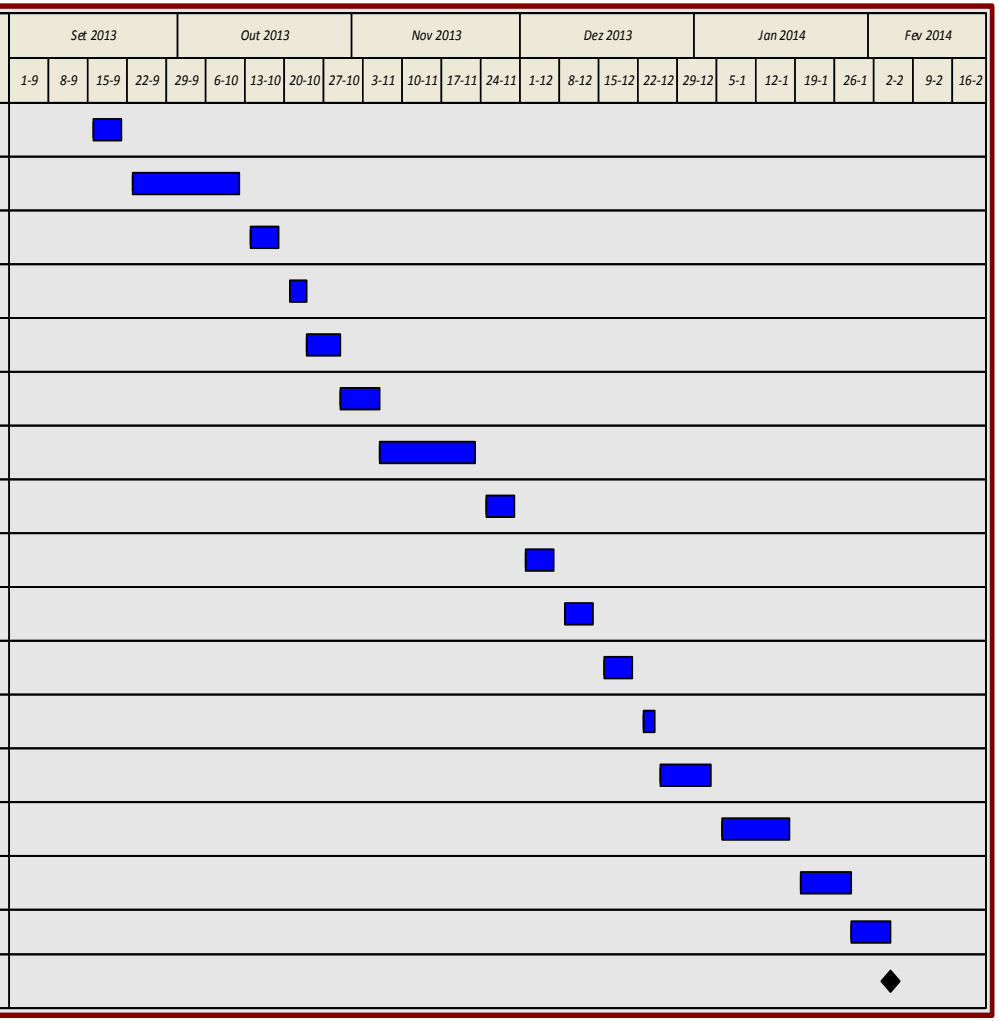

Figure 3 - Actual plan First Semester

Second Semester

ID	Task Name	Start	Finish	Duration
1	MSRP Study & Tests	03-02-2014	14-02-2014	10d
2	MSRP Chunking	17-02-2014	28-02-2014	10d
3	Use Case 1 – Image Share During Call	03-03-2014	07-03-2014	5d
4	Use Case 2 – Image Share Before Call Setup	10-03-2014	14-03-2014	5d
5	Refer/Re-Invite	17-03-2014	20-03-2014	4d
6	Use Case 2 – Image Share Before Call Setup	20-03-2014	21-03-2014	2d
7	Use Case 3 – Image Share During Ringing	24-03-2014	28-03-2014	5d
8	Image Share Use Cases Testing	31-03-2014	04-04-2014	5d
9	Kurento Media Server Install & Configuration	07-04-2014	10-04-2014	4d
10	Kurento Media Server API Integration	11-04-2014	17-04-2014	5d
11	Video Share Tests	14-04-2014	18-04-2014	5d
12	Advertisement Server Architecture Refactor	21-04-2014	30-04-2014	8d
13	Use Case 4 – Video Share During call	01-05-2014	09-05-2014	7d
14	Use Case 5 – Video Share Before Call Setup	12-05-2014	16-05-2014	5d
15	Use Case 6 – Video Share During Ringing	19-05-2014	20-05-2014	2d
16	Database	20-05-2014	23-05-2014	4d
17	Backoffice & Database	26-05-2014	06-06-2014	10d
18	Testing & Debug	09-06-2014	20-06-2014	10d
19	Internship Report	13-06-2014	01-07-2014	13d
20	Presentation	01-07-2014	04-07-2014	4d
21	Internship conclusion	07-07-2014	07-07-2014	0d

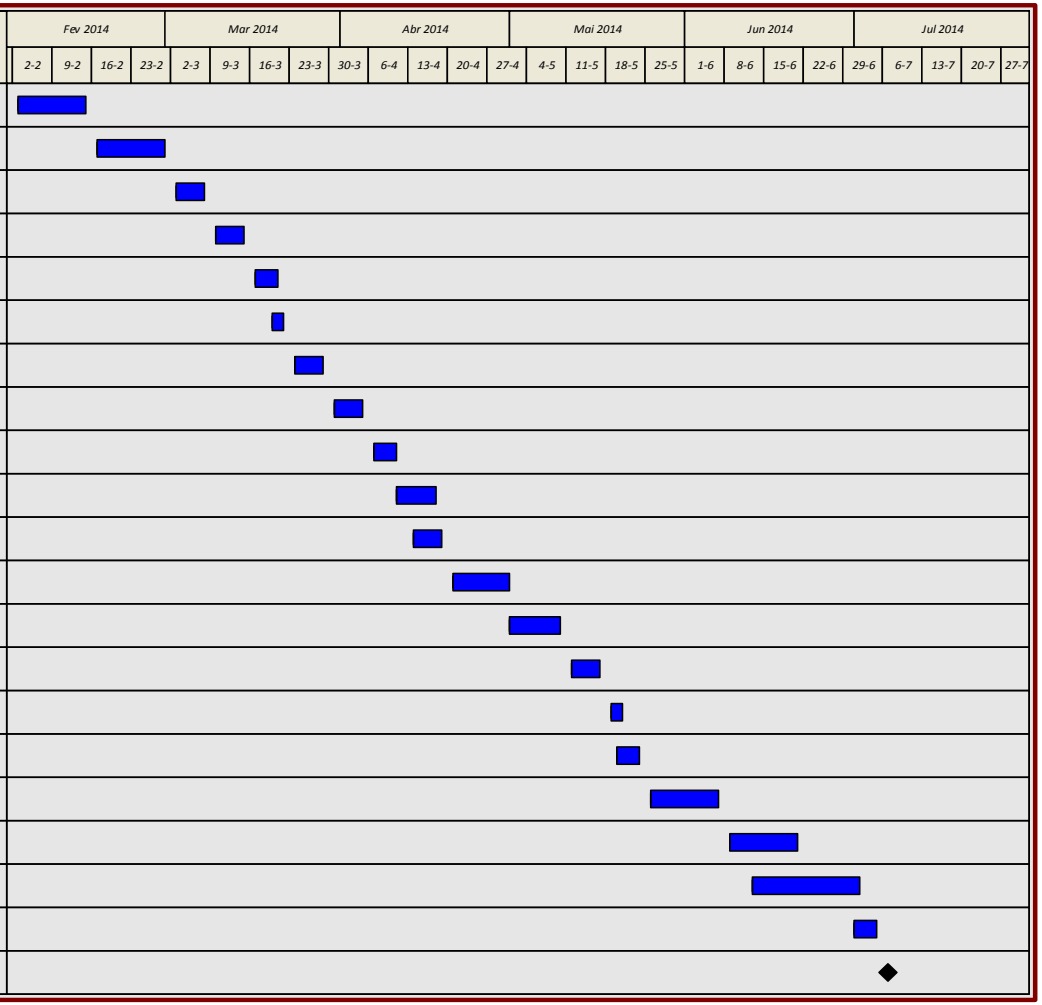

Figure 4 - Actual plan Second Semester

Chapter 2
State of the Art

2.1. Introduction

In the last 30 years the telecommunication industry as evolved from landline to mobile. But in the last decade, with the advent of Smartphones, and the improvement of Broadband Internet, a shift has occurred in industry. New actors, like OTT, are now offering competing products; most of the time at a lower price, or even for free. In this chapter, a brief explanation of OTTs is given, and how they thrive and threat the telecommunication industry. Also an analysis is made in order for the reader to understand how Telecommunication companies are responding to this threat, and how this answer is integrated with the scope of this internship. Some explanation is also given about some technologies/architectures chosen by the author.

2.2 Telecommunications Network Evolution

When people use the word "telecommunications", most think of the classical analog telephone. That is telephony. Telephony is focused on voice communications, while Telecommunications has evolved into much more. Telecommunications can be described as the science of communicating over a long distance using a telephone or radio technology. Nowadays telecommunication not only involves voice, but also video communications and data transmission.

The first commercial telephone network was born in the waning years of 19th century, and by this time there was no concept of switching (the process of setting up a temporary connection between two parties for the duration of the call). In this first iteration, a dedicated circuit between telephones was necessary. The number of connections in this network topology grows exponentially. For example, in order to fully connect a city of 35000 people, it would take around 600 million connections.

Soon companies realise this was untenable, and a new hubbed model was adapted. Instead of connecting every customer permanently to other customers, the circuits were instead connected to a central "hub" (today's central office). In this central, human operators would establish temporary connections between callers on-demand. Operators, then, were the world's first telephone switches.

In 1891, Almon Strowger invented the Strowger Switch, which gave customers the ability to dial the person they wanted to talk to directly. That switch became the mainstay of the telephone network for over 50 years. This was the beginning of the public switched telephone network (PSTN), an aggregate of the world's circuit-switched telephone networks [6].

In circuit-switched networks, electronic signals pass through several switches before a connection is established, and during a call, no other network traffic can use those switches. Telecommunications industry understood this was a limitation, and in 1970, began implementing packet switched networks using X.25 protocol. Those networks used much of the end-to-end equipment which was already in use in the PSTN.

In packet-switched networks, the message is broken in small data packets, which seek out the most efficient route as circuits become available. That means that packets from the same

message might use different routes. Messages are then re-assembled in the destination's endpoint. Formerly, packet-switched digital networks would connect to circuit-switched ports to gain access to computer networks in different locations, but nowadays, with the improvement of bandwidth and reliability of the Internet, remote dial-up access to corporate computers is usually over the Internet.

2.2.1 Mobile Telecommunications

In the last 30 years, telecommunications evolve from fixed technologies to mobile ones, so, in order for the reader to fully understand the future of telecommunications, it is essential to understand the evolution of Mobile Telecommunications. In the past, enterprise adoption was a key driver of technology commoditization; however, recent progress in mobile telecommunication industry has been largely shaped by consumer demand. Table 1 summarizes the evolution of mobile networks.

	First Generation 1G	Second Generation 2G	Second + Generation 2.5G	Third Generation 3G	Fourth Generation 4G
Date	1970-1984	1980-1991	1985-1999	1990-2002	2000-
Technology	Analog	Digital	Digital	Broad Bandwidth	Unified IP and seamless combination of broadband
Standard	AMPS, TACS, NMT	GSM, TDMA, CDMA	GPRS, HSCSD	WCDMA, CDMA-2000	Single Unified Standard
Bandwidth	1.9kbps	14.4kbps	14.4kbps	2mbps	200mbps
Network	PSTN	PSTN	PSTN & Packet network	Packet network	Internet
Switching	Circuit	Circuit	Circuit & packet	Packet except for air interface	All packet
Services	Voice mono-service Person-to-Person	Voice, SMS, Media Person-to-Person	Higher capacity, packetized data	High quality audio, video and data	Dynamic information access, wearable devices

Table 1 - Comparison of 1G, 2G, 2.5G, 3G and 4G [7]

2.2.1.1 Early Services – 0G

In the year of 1946, AT&T introduced the first wireless telephone, the Mobile Telephone Service, but because of the weight of the equipment (36kg), and the fact that only three radio channels were available (only three customers in any city could make mobile calls at one time), it was not very popular. In 1960s AT&T made another try by introducing the IMTS (Improved Mobile Telephone Service), which had more radio channels available and reduced size and weight. Despite the improvements, a wait of 30 minutes to place a call was common. Other mobile services were released all over the world; West Germany had a network called A-Netz, and Norway a system called OLT, but none of them had significant success. These first attempts of deploying a mobile network are known as 0G.

2.2.1.2 Analog Cellular Networks – 1G

The first analog cellular system widely deployed was the Advanced Mobile Phone System (AMPS) in the 1980s. It was a pioneering technology that helped drive mass market usage of cellular technology. Despite the success, it had several issues like unencrypted data and was easily vulnerable.

2.2.1.3 Digital cellular networks – 2G

The second generation of mobile phones emerged in the 1990s. Two systems compete for the global market, the European GSM (Global System for Mobile Communications) and in the U.S. the CDMA (Code Division Multiple access). These networks differ from 1G because they used digital transmission instead of analog, and also for using phone-to-network signalling. The rise of mobile phone usage as a result of 2G was explosive, and in this decade, a huge improvement in mobile phones technology was made.

In this generation, GMS introduced a new variant of communication called SMS or text messaging. SMS soon became the method of choice amongst the young. 2G also introduced the ability to access media content on mobile phones, mainly ring tones.

Naturally, companies saw an opportunity in mobile devices, and the first advertising service appeared in Finland in 2000. This service was a free daily SMS news headline, sponsored by advertising.

2.2.1.4 Mobile broadband data – 3G

As mobile devices became more widespread and people became more dependent of their phone, it was clear for mobile industries that the demand for data was growing. From the experience of Internet Service Providers, companies also realize the never ending demand for greater data speed. It's within this scope that the 3G technology is developed. The main difference that distinguishes 3G networks from 2G is the use of packet switching rather than circuit switching for data transmission, and of course, data speed.

During the development of 3G systems, other systems like CDMA2000 and GPRS were developed as extensions to existing 2G networks also known as 2.5G systems. Although they provide some features of 3G, they not fulfil the high data rates or full range of multimedia services.

2.2.1.5 Native IP Networks - 4G

Due to the increasing number of Smartphone users, it was clear that 3G networks would be overwhelmed by the growth of bandwidth-intensive applications like streaming media. The industry started to develop data-optimized networks, which could support speed

improvements up to 10-fold over 3G technologies. The two main 4G networks available are WiMAX (World Interoperability for Microwave Access) and LTE (Long Term Evolution).

The main difference between 4G and 3G networks (other than speed), is that 4G completely eliminates the use of circuit switching. Thus 4G treats voice calls like any other stream of data, utilizing packet switching over internet, LAN or WAN networks via VoIP (Voice over IP).

2.3 Over-the-Top Applications

As mentioned, Over-the-Top Applications (OTT) can be described as any application or service that is provided over the Internet that is not provided directly by Internet Service Providers (ISP) or Telecommunication Providers. The critical point about OTT is that ISPs or Telecommunication Providers do not earn any money from those services. When talking about OTTs, ISPs become merely providers of IP connectivity. The big thread for telecom companies and ISPs is that most of OTT offer competing products at lower price, or in some cases, for free. The arrival of smartphones, their declining price and the update of access networks are contributing to OTT growth in the communication market. For example, Messaging OTTs like "WhatsApp" and "iMessage" declined SMS revenue by Telcos at 13.9 Billion dollars in 2011 [8] and Voice/Video calls OTTs like Skype, Google Hangouts or Facetime are becoming more popular, costing more than 36 billion dollars a year to the telecom industry. Paradoxically, the increasing popularity of this services generate a growing flow of traffic and an increasing demand for broadband, which translates into a need for network investment by ISPs/Telecommunication providers.

In the next sections, some examples of OTTs that threat telecom companies' revenue are described, and an explanation about how the Telecommunication industry is responding to these threats is given.

2.3.1 Solutions Available

In this section of the chapter, some of the most used OTT's that threat the Telecommunication Industry are analysed, however, it is noteworthy that several others exist on the market, but most of them offer the same features, so it is unnecessary to describe them. Table 2 has a feature list of the most used OTTs.

	Skype	Facetime	Google Hangout	WhatsApp	Viber	Kakao Talk
Voice Calls	✔	✔	✔	✖	✔	✔
Video Calls	✔	✔	✔	✖	✔	✖
Messaging	✔	✖	✔	✔	✔	✔
PSTN integration	✔	✖	✔	✖	✖	✖
Free	✔	✔	✔	✔	✔	✔
Advertisement	✔	✖	✖	✖	✖	✔

Advertisement on Call	✔	✖	✖	✖	✖	✖
Number of Users	+300M	-	-	+200M	+200M	+100M

Table 2 - OTTs Features

Skype and Google Hangout have a yellow "check" on the "Free" field because those services charge for calls to regular phones. Facetime is free for newer iPhones, but cost about a dollar in the Apple Store for older devices.

2.3.2 Advertisement in OTT Applications

As mentioned, the first advertising service in telecommunications was developed by a Finnish company in 2000. The service sent a free SMS everyday with the main news. Since then, companies have unsuccessfully tried to build a business model in telecommunications advertising.

With the advent of Smartphones, and the "fusion" between telecommunications and the Internet, many of the major Internet companies like Google and Skype are trying to bring their expertise in Internet advertisement to the mobile telecommunications world. Despite there is still no active on-call advertising service on the telecommunication market, there are some interesting products, especially in VoIP services.

The Pudding [9]

The Pudding was a service that provided VoIP, PC-based calls to anywhere in the US or Canada. The interesting fact about this product was that it monitored and analyzes every word of conversations so they could serve up advertisement. Users would access the website, entered a phone number, and after the call is established, advertisements tailored to the conversation will begin to appear on screen (Figure 5). The product was never released (only a beta version), and the company closed in 2008.

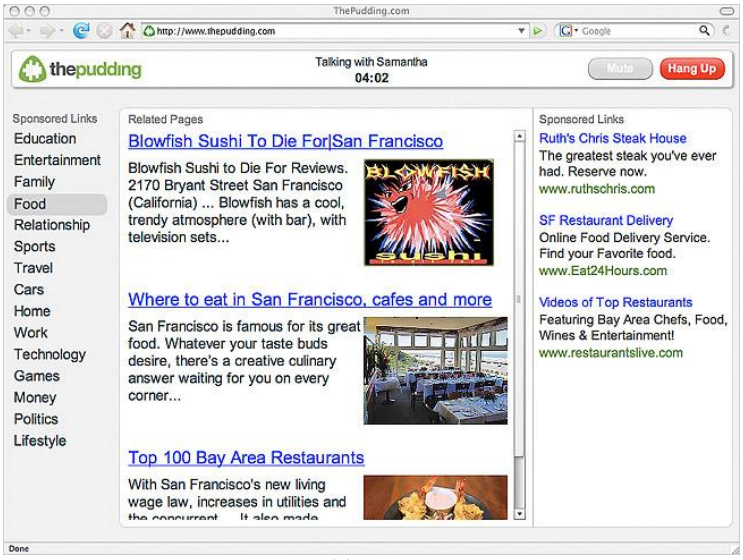

Figure 5 - The Pudding [10]

Jangl and Jajah [11]

In 2008, Jangle, a Silicon Valley Internet phone company announced they were testing "in-call advertising". While customers waited for Jangl to connect their calls, the originating party would hear audio advertisement. Almost at the same time, jajah, also an Internet-based communications company announced they were ready to launch in-call ads. In this case, ads were optional, and in return, the user would earn credits. Jangl closed in the end of 2008, while jajah announced they were closing their services in the beginning of 2014.

Microsoft Skype Conversation Ads [12]

In June of 2012, Microsoft and Skype launched the advertising platform Conversation Ads. This platform displayed ad units within the calling window of Skype. These ads appear to users who do not have a paid subscription to Skype. These ads can be video, image or games. It's noteworthy that this platform uses clients' information for targeting ads (e.g. location, gender or age). Figure 6 illustrates a Skype call with Conversation Ads.

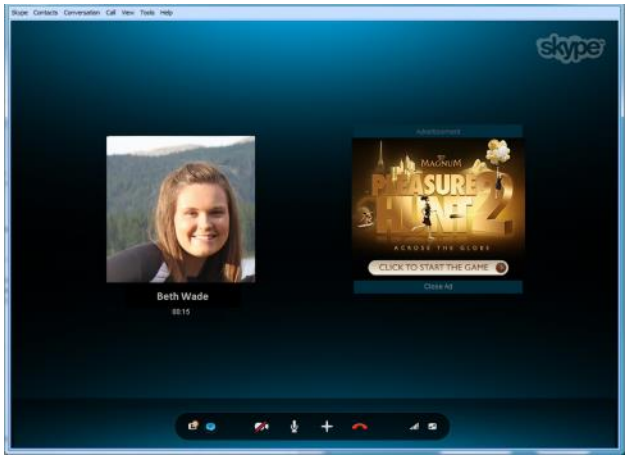

Figure 6 - Skype Conversation Ads

Google [13]

At the time of writing, Google does not include advertising in its Voice service, however, in March 20012, Google registered a patent for targeted advertising in Phone Calls. It's noteworthy that this patent only covers audio ads in phone calls before a call is being answered or when a call is on hold.

2.4 Operators' Solution – Rich Communication Services

RCS is an initiative run by the GSMA. It brings together the main actors in the telecommunications industry from network vendors, telecom operators, handset manufactures and software and Internet companies for the creation of an "interoperable, convergent, access-technology-independent rich communication experience to end-users". At the time of writing, there are nearly 200 companies participating in the RCS initiative within GSMA.

RCS is not a group that defines new standards, instead, they bring together the already defined services into profiles based on global standards (e.g. IMS, presence, content sharing), and define requirements for service features. The main goal of RCS is to facilitate the introduction of commercial, IMS-based communication services for 3G, LTE, CDMA and fixed networks. The necessity for RCS lies with two main points:

- The necessity for global interoperability between social driven services
- Subscriber-centric social networking, rather than platform-centric social networking.

For the purpose of this project, two main features of the RCS 5.1 [5] (current version) are discussed:

- Capability and new user discovery mechanisms (3.2.1 Capability Discovery).
- Content Sharing (3.2.2 Content Sharing).

2.4.1 IP Multimedia Subsystem

Introduction

In the past decades, the communications industry has undergone profound changes driven by similar forces across mobile, fixed, and IT/Computing networks. Equipment vendors and operator have pursued lower cost technologies, most often based on IP technology. It was within this atmosphere that the IP Multimedia Subsystem (IMS) was first designed.

At its simplest, IMS is a set of IP-based technologies, or an architecture framework, that allows for ubiquitous access to multimedia services from any terminal, be it a mobile, landline phone, or PC. It is designed to provide global interoperability between all handsets and all operators worldwide. Another key feature of IMS is that it was designed to handle universal service access, in other words, wherever the user is roaming in the world, his handset should provide him the same set of services. This happens automatically, without additional development work, or more important, the user needing to do anything.

From the business point of view, IMS covers two fundamental market needs: first, what all users expect in terms of standardized services, where we can reach anyone without having to worry what operator the person in the other end has chosen, or what device he/she uses. Secondly, the operator, together with service partners, can provide innovative and differentiating services that make the operator more attractive to end-users.

History

IMS was initially defined by an industry forum called 3G.IP, formed in 1999, and then was brought to the 3rd Generation Partnership Project (3GPP). It first appeared in Release 5 (evolution from 2G to 3G) and originally was designed to evolve Universal Mobile Telecommunications System (UMTS) networks to deliver Internet Protocol multimedia to mobile users. Support for the older GSM and GPRS networks was also provided.

IMS specification was later refined by subsequent Releases, those changes and add-ons are described in the table below (Table 3).

Release	Change /Add-on
6	Added interworking with WLAN, inter-operability between IMS, multiple registration and forking, presence, speech recognition and speech-enable services (Push to talk).
7	Added support for fixed networks, interworking with non-IMS networks, policy and charging control (PCC), emergency sessions.
8	Added support for Long Term Evolution (LTE) and IMS centralized services.
9	Added support for IMS emergency calls over GPRS and EPS, enhancements to services centralization and continuity.
10	Added support for inter device transfer, enhancements to single radio voice call and IMS emergency sessions.
11	Added Unstructured Supplementary Service Data (USSD) simulation service and network-provided location information.

Table 3- IMS Evolution [14]

Release 12 is planned to be launched in the 2nd quarter of 2014.

2.4.2 IMS Alternatives

Although IMS is the standard architectural framework for delivering IP multimedia services, and for that reason it is the chosen framework for this project, some alternatives exists on the market.

B-ISDN and ATM

Broadband Integrated Services Digital Network (B-ISDN) [15] is a broadband communication network developed by International Telegraph and Telephone Consultative Committee (CCITT) that enables the transmission of text, voice, video and other multimedia services in one network. This network is able to increase transmission rate, up to 155Mbits/s on a switching basis.

This network is the evolution of the Integrated Service Digital Network (ISDN) that provided transmission rate of only 64kbits/s, and it is based on a digital packet switching technique known as Asynchronous Transfer Mode (ATM) [16]. In ATM technology, packet switching is done purely on hardware level, without the need for software interface.

The main reason this network was not adapted as a standard in the Telecommunication Industry is the investment it required. Because of ATM hardware switching, and in order to fulfil the high transmission rated required in B-ISDN, most of the infrastructure of a Telecommunication provider need to be changed. It was declined in favour of Next-Generation-Networks (all IP networks like IMS).

Peer-to-Peer SIP

This implementation uses a peer-to-peer (P2P) architecture, in which there is no need for a centralized server. In this approach, users communicate and negotiate sessions directly with each other. P2P SIP is less complex than IMS or B-ISDN, and offers scalability. However, there is no standardization in policy control, and integration with legacy networks is complex and forces the use of a centralized system.

Nevertheless, there are some products in the market that uses this approach (or slight variations), being the most successful Skype [17].

2.4.3 IMS Solutions Available

At the time of writing, the most notable available open-source/free solutions for the implementation of a functional IMS network are the following.

Imszone

htttp://www.imszone.org/

This solution provides all the components that form the core of an IMS network, however, development and support has stopped a few years ago.

LittleIMS

https://cipango.atlassian.net/wiki/display/LITTLEIMS/

Solution with core elements of an IMS network, however, not all the interfaces are available. Development and support have stopped by the end of 2011.

OpenIMS

http://www.openimscore.org/

This is the most used open-source solution of an IMS network. Besides the core elements of IMS, it provides extra components and interfaces. No update has been launched since December 2012; however, end of development has not been announced. One of the main reasons for the success of OpenIMS is its active community.

Project Clearwater

http://www.projectclearwater.org/

This product is the first open-source solution specific for cloud deployment. Although it is an open-source solution, its design for deployment on Amazon Web Services, so, some cost should be taken into account. This solution offers the core elements of IMS network, and some additional components like Media Relay. Development and support are still active, but it has a small community.

Kamailio

http://www.kamailio.org/

Former known as OpenSER, it is a merge of different projects (OpenSER + SER + SIP Router). Development and support are still active, with a new released launched in December 2013. Although it can be used as an IMS network, it is basically an SIP Server. It provides all CSCF entities, but not a HSS.

Chosen solution

In order to choose the best solution for this project, several aspects were taken into account. Next, a table of comparison between all the solutions is presented.

	IMS Core Entities	Interfaces	Development and Support	Extra components	Active Community
Imszone	✔	✔	✖	✖	✖
LittleIMS	✔	✔	✖	✖	✖
OpenIMS	✔	✔	✔	✔	✔
Clearwater	✔	✔	✔	✔	✖
Kamailio	✔	✔	✔	✔	✔

Table 4 - IMS Solutions Comparison

After evaluating all solutions, OpenIMS was chosen for setup of an IMS network. This choice was driven to the fact that this solution offer all the necessary entities, some extra components, and because it has a huge active community behind. Also, a lot of the company collaborators have experience with this solution, and could provide some guidance if any difficulty appeared while setting up the network.

2.5. Media Server / Media Gateway

Media Servers or Media Gateways are components responsible for handling multimedia streams (video and audio) like playing announcements, streaming video, voice recognition, etc... It's noteworthy that all this functions are controlled by Application Servers deployed on the IMS network, via Media control Protocols like MGCP or H.248, protocols that are explained in the next chapter. It can be distributed as only software or software and hardware, and in the context of the IMS architecture, it can be seen as the Media Resource Function Processor (MRFP) as seen on Figure 7.

2.5.1 Solutions Available

There are several free/open source solutions on the market for this component. The requirements for choosing the media servers are:

- Support for most common codecs:
    1. Audio Codecs:
        - G.711 (must-have)
        - GSM (must-have)
        - G.729 (nice-to-have)
        - Speex (optional)
        - Opus (optional)
    2. Video Codecs:
        - H.261 (nice-to-have)
        - H.263 (must-have)
        - H.264 (must-have)
        - VP7 (optional)
        - VP8 (optional)
- Standard control protocol
    1. MGCP (must-have)
    2. H.248 (nice-to-have)

Further information about codecs can be found on Annex A.3.

The comparison of all the media servers/media gateways found is presented on Table 5.

		Imszone	Kurento	Mediactrl	Mobicents MS	MCU Medooze	Sip Express	Lynckia
Audio Codecs	GSM	✔	✔	✔	✔	✔	✖	✖
	G.711	✔	✔	✖	✔	✔	✖	✖
	G.729	✔	✖	✖	✔	✖	✔	✖
	Speex	✖	✔	✖	✔	✔	✔	✖
	Opus	✖	✖	✖	✖	✖	✖	✔
Video Codecs	H.261	✖	✔	✖	✔	✔	✖	✖
	H.263	✖	✔	✔	✖	✔	✖	✔
	H.264	✖	✔	✔	✖	✔	✖	✔
	VP7	✖	✖	✖	✖	✖	✖	✖
	VP8	✖	✖	✖	✖	✖	✖	✔
Control Protocol	MGCP	✖	✔	✖	✔	✖	✖	✖
	H.248	✖	✖	✖	✖	✖	✖	✖
	Other	✔	✔	✔	✖	✔	✔	✔

Table 5 - Media Server Comparison

From the analysis of the entire available media server, only Kurento media server [18] offers the desired functionalities, and for that reason, it is the chosen media server for this project.

Chapter 3
Technical Background

3.1 Voice over IP (VoIP)

Voice over IP (VoIP), also known as IP Telephony, can be described as the real-time transmission of voice signals using the Internet Protocol (IP) over the Internet or a private data network [19].

The main difference between a VoIP and a regular call is that the first uses a packet switching network instead of traditional circuit switch. As referred before, circuit switching establishes a dedicated path between the source and the destination of the call. However, the capacity of the circuit is not shared by other users, thereby hindering the systems' overall capability. In contrast, a packet switching network (like the Internet), switches data through a network by splitting data into packets. For this reason, network resources can be shared by multiple calls. However, for this same reason, VoIP calls might have a poorer quality than normal PSTN calls. Table 6 presents the advantages and disadvantages of both technologies.

	Advantages	**Disadvantages**
Circuit Switching	• Circuit dedicated to the call. No interference • Full Bandwidth guaranteed. • Quality of Service (QoS)	• Inefficient – if no data is being sent, the line still remains open. • Long time to set up the circuit • Unstable or Unavailable during crisis or disaster.
Packet Switching	• Security • Bandwidth used to full potential • Different speed devices can communicate • Redirects signal (not affected by line failure) • Availability • During a crisis, when public telephone network might not work, emails and texts can still be sent	• There can be a delay under heavy use • Packets can get lost or be corrupted • Protocols are needed for reliable transfers

Table 6 - Advantages/Disadvantages Packet and Circuit switching [20]

VoIP was introduced in February of 1995, by a small company in Israel called Vocaltec. Their product was a softphone (computer program), that allowed users to call each other via their computers, using a microphone and a set of speakers. Additionally, this application only worked if both users used the same software. This had a massive success in huge companies, because calls within those companies became virtually free. However, driven by poorer quality then circuit switch calls, it had little success in private industry.

Advertisement reached VoIP calls by 1998, in the U.S., when some entrepreneurs started to market PC-to-phone and phone-to-phone VoIP solutions. Those calls were marketed as "Free", and the caller had to listen to advertisements before the call was connected. These first services are called First Generation Providers, and by the year of 2000, only 3% of all calls were VoIP calls.

Second generation providers, like Skype, changed the business model of VoIP calls, building closed networks for private user bases, offering the benefit of free calls, while charging for access to other communication networks (Freemium), such as PSTN. By the end of 2003, almost 25% of all calls were VoIP calls [21].

VoIP can use several protocols like SIP, H.323, RTP, SDP, or proprietary like Skype or Teamspeak. Some of these protocols are later described.

3.2 Rich Communication Services (RCS)

3.2.1 Capability Discovery

Described in section 2.6 of the RCS 5.1 specifications [5], the capability discovery is a process that enables endpoints to understand what subset of RCS services (e.g. video share, file transfer, video call) are available in another endpoint (in rare cases, by servers), at a certain point in time.

RCS 5.1 gives two alternatives for capability discovery:

- **SIP OPTIONS exchange**, in this method an endpoint queries the capabilities of another through using a SIP message (SIP OPTIONS). This message already contains the capabilities of the originating endpoint. The response to this request will have the capabilities of the target contact. Using this method, both users get updated information in a single transaction.
- **Presence**. In this case, the capabilities are queried against a server deployed on the network, and not directly to end-user. This server must use standard Open Mobile Alliance (OMA) SIMPLE Presence procedures. It's noteworthy that in order for the Presence server to use OMA standard, an XDMS (XML Document Management) must also be deployed in the network.

RCS 5.1 also specifies that the default method for capability discovery is SIP OPTIONS exchange, so, even if a network uses a Presence server, it must also support SIP OPTIONS exchange. For this reason, and because the main purpose of this project is to develop an Application Server to be deployed in the network, SIP OPTIONS is the method chosen for capability discovery.

3.2.2 Content Sharing

Content sharing in the scope of RCS can be defined as the ability to share different types of content like chat, files, images and video in near real-time. This functionality can be used both in connection to a voice call, or in a standalone manner, when there is not an ongoing call. For the purpose of this project, focus is given to the first option. Content sharing is specified in chapter 3.6 of RCS 5.1, and 3 different sources for the shared content are presented:

- The front camera of a device
- The rear camera
- File (video streaming or stored image).

Maximum sizes of files/images and video are not specified by RCS, and are configurable by Service Providers.

The content share services during a call are connected to that call, which means that when the call ends, all shared content is automatically terminated. Before any sharing is made, the originating user must recognize that the terminating user supports that type of content sharing service (video and image). This can be achieved by capability discovery (chapter 3.2.1 Capability Discovery).

Both Video and Image share sessions are unidirectional, but 2 sessions can be setup at the same time in different directions. This means that once a user A is sharing a video with user B, user B can also start a video share with user A. If capabilities of any user change while having an active session, and it is no longer supported, this user must terminate the sharing session.

3.2.3 RCS Client

In order to implement and test all capabilities of Rich Communication Services, a RCS client is needed. There are several RCS clients available, most of them are free, but are not available for Portugal, or, are not configurable in order to connect to a custom network.

WIT Software S.A. was selected by GSMA as an official joyn distributor (GSMA RCS client). Joyn supports and follow all RCS 5.1 specifications, and for that reason, it is the RCS client that will be used in this project.

3.3 IP Multimedia Subsystem

3.3.1 Architecture

This section introduces the IMS network architecture [22]. The next figure (Figure 7) shows the architecture as defined in 3GPP technical specifications.

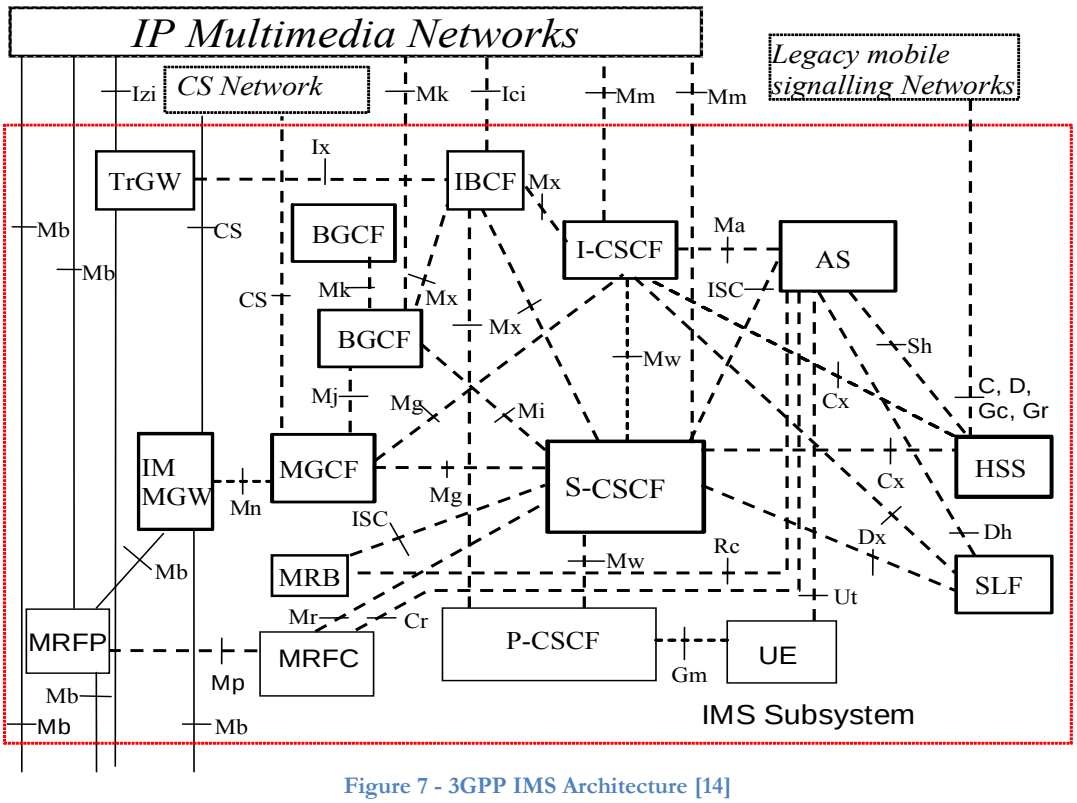

Figure 7 - 3GPP IMS Architecture [14]

The objective of this section is to explain this architecture in a simpler way.

First, we can break IMS architecture into three big layers (Figure 8):

- Transport and Endpoint Layer
- Session and Control Layer
- Applications and Services Layer

The **Transport and Endpoint Layer** unifies transports and media from analog, digital, or broadband formats to Real-time Transport Protocol (RTP) and Session Initiation Protocol (SIP), protocols that are explained later in this chapter. This unification is accomplished by media and signalling gateways. This layer also includes media servers that allow for announcements, in-band signalling, and conferencing. These media servers are shared across all applications, maximizing statistical use of the equipment and creating a common base of media services without "hard-coding" these services into the applications.

Session and Control Layer orchestrates logical connections between various other network elements. It provides registration of end-points, routing of SIP messages, and overall coordination of media and signalling resources. It's within this layer that the IMS Core (detailed in next session) is located.

The **Application Services Layer** contains, like the name suggests, multiple Application Servers (AS). Each of these servers is responsible for performing functions on subscriber sessions, maintaining the state of the call. This layer is detailed in further sections.

So, after separating the architecture above into layers, we end up with a more user-friendly and easy to understand architecture.

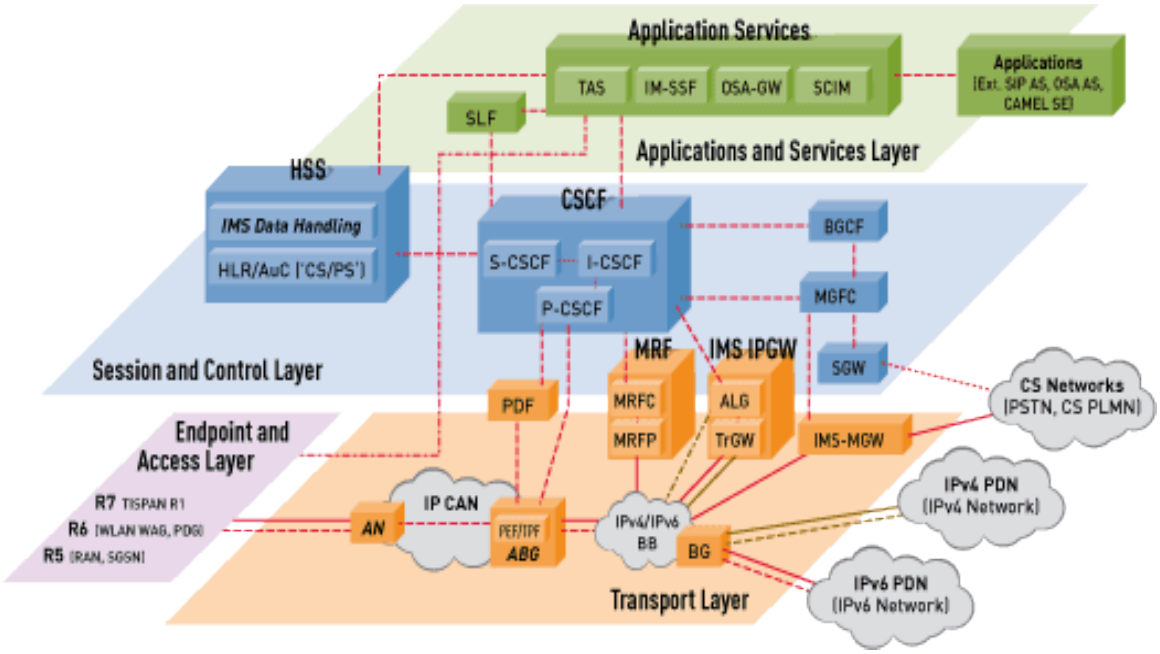

Figure 8 - IMS Layer Architecture [23]

3.3.2 IMS Core

In this section we take a closer look to the session and control layer, particularly to the IMS Core. This core consists in a limited number of functional entities, being the more important:

- **Home Subscriber Server** (HSS) – Database that stores all necessary information about users and servers.
- **Proxy Call Session Control Function** (P-CSCF) – entry and exit point of IMS Core. It makes the connection with User's Endpoints.
- **Interrogating Call Session Control Function** (I-CSCF) – Queries the HSS to find what S-CSCF is associated with some user, or register the user in one.
- **Serving Call Session Control Function** (S-CSCF) – connects the IMS Core with Application Servers.

The specific function of each of the functional entities can be found in Annex A.1. Figure 9 describes how IMS core is specified, and the possible communications between entities.

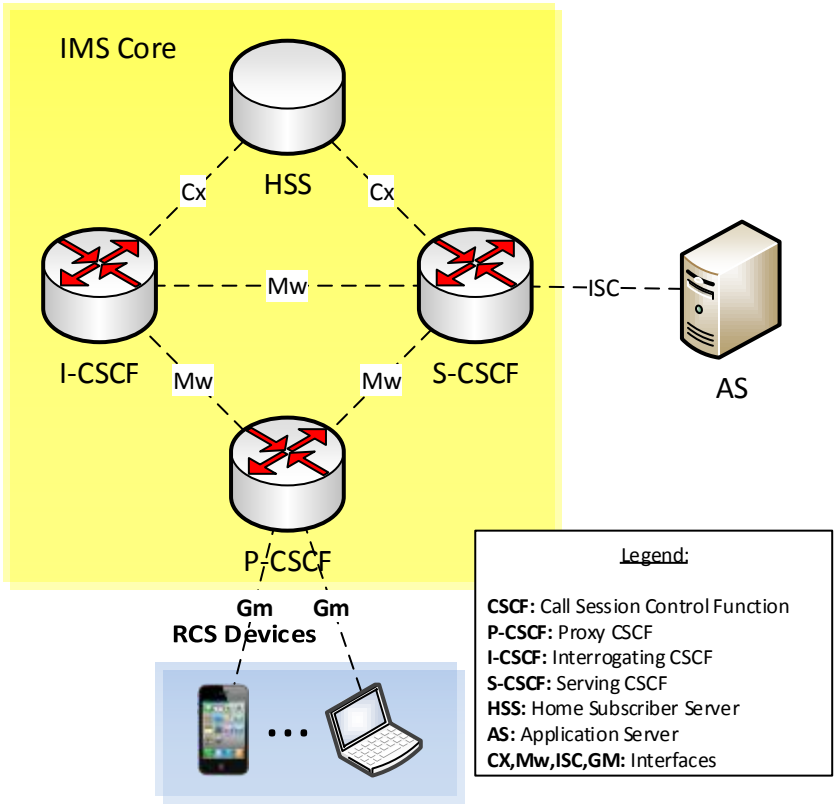

Figure 9 - IMS Core Overview

Interfaces

As the reader might have noticed in Figure 7 and Figure 9, the communication between IMS components is made through the use of interfaces, as defined by 3GPP in IMS specification. Although most of the IMS solutions and Application Servers Development Technologies implement these interfaces automatically, it is important to understand that are used different interfaces and protocols between components. The two main protocols used on the IMS entities are **SIP** and **Diameter** (Annex A.2.4). SIP is mainly used for message relay among entities, and Diameter for session control. Figure 10 and Table 7 shows the different interfaces, and the protocol they use.

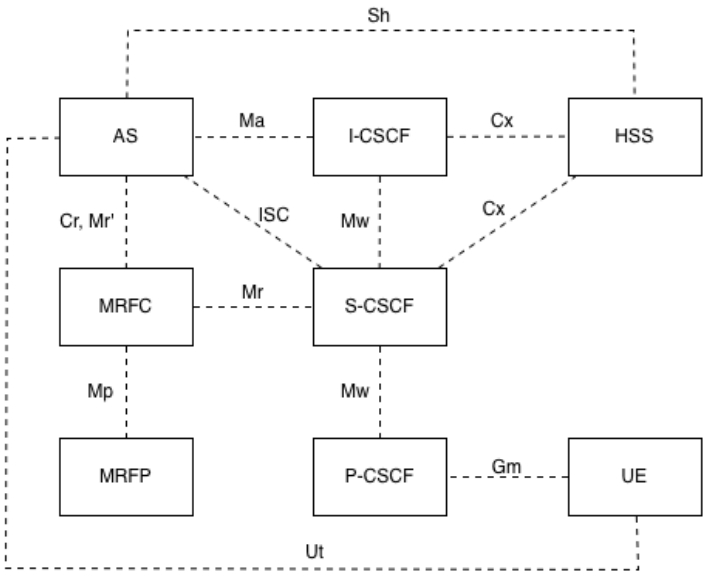

Figure 10 - IMS Interfaces

Name	Description (Reference Point between...)	Protocol
Cx	a CSCF (I-CSCF or S-CSCF) and an HSS.	Diameter
Mw	CSCF's.	SIP
Ma	an AS and an I-CSCF.	SIP
Mr	a CSCF and an MRFC.	SIP
ISC	a S-CSCF and an AS.	SIP
Sh	an AS and a HSS.	Diameter
Gm	a UE (User Endpoint) and a P-CSCF.	SIP
Mp	a MRFC and a MRFP.	Megaco/H.248
Cr	an AS and a MRFC (media control).	SIP
Mr'	an AS and a MRFC (session control).	SIP
Ut	UE and an AS.	HTTP,XCAP

Table 7 - IMS Interfaces Description and Protocol

It is noteworthy that other interfaces exist in an IMS network, but for the sake of simplicity, only the interfaces relevant to the scope of this internship are described.

3.3.3 Application Servers

Application servers (AS) have a specific and prominent role in IMS networks. ASs are the main difference between common Circuit Switch network and IMS. CS networks itself are fully geared for offering a telephony service, and in addition, they contain a large set of supplementary services attached to it. One of the main features of IMS is that it was developed in order to simplify the development and deployment of powerful value-added services (VAS). VASs are implemented on IMS network though ASs, also known as SIP application servers (SIP-AS).

It's noteworthy that AS might themselves be connected to third party services (e.g. external databases), or other Application Servers on the network (Figure 11). This feature creates the opportunity for external actors (e.g. Software development companies), and not only the network vendors, to develop services and business models supported by an existing IMS network.

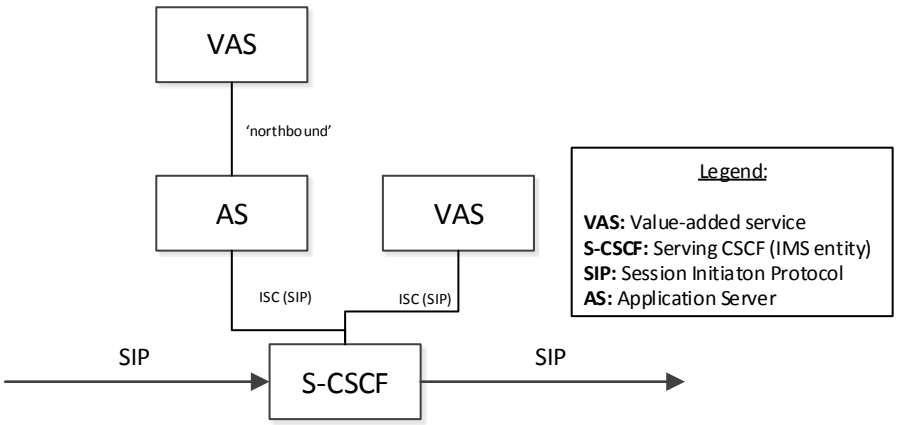

Figure 11 - IMS Value-Added Services [22]

3.3.4. Application Server Development Technologies

At the time of writing, there are two main technologies for AS development, both for Java programming language, JAIN SLEE and Sip Servlets.

JAIN SLEE

https://developer.opencloud.com/devportal/display/OCDEV/JAIN+SLEE

This is a Java Programming Language API specially design for the development of communications applications. It allows developers to easily produce highly available and scalable applications, with low latency and high throughput. It was first specified by a collaboration of different telecommunication operators and software vendors as an event driven application server, and integrates the ACID (atomicity, consistency, isolation, durability) properties. It is designed to automatically handle transactions; therefore, the main focus of the developer should be on handling events. Current version is 1.1.

SIP Servlets

https://java.net/projects/sipservlet-spec/pages/Home

Sip Servlet is a Java programming language server-side component that performs SIP signalling. This technology is built off generic HTTP Servlets and is used for developing applications that are deployable across different containers, as long as they support SIP Servlets, with minor changes. It also allows integration with other JEE Components. Current version is 2.0.

Technology chosen

Although JAIN SLEE has a slower learning curve than SIP Servlets, it was the chosen technology to be used in the development of the Application Server. The main reason for this choice was the fact that JAIN SLEE is a technology specifically design for the development of communications applications. It has fewer restrictions than SIP Servlets, like for example, the support for Diameter protocol. It is also, in terms of software engineering, a more correct way to develop applications, because it is design to promote re-usable components, and it is OO (Object Orientated) component model, as well as offers concurrency and failure control. Table 8 shows a comparison between the two technologies.

	SIP Servlets	JAIN SLEE	
SIP supported	✔	✔	IMS Requirements
Supports more than one SIP API	✖	✔	
Common SIP Components (e.g. Proxy)	✔	✔	
Service Interaction & Coordination	✔	✔	
Other IMS protocols	✖	✔	
Standard Tracing and Alarms	✖	✔	O&M and Extensibility to IMS
Support for non IMS nodes & protocols	✖	✔	
Promotes good software architecture	✖	✔	Quality, Robustness, cost, maintenance
Includes Failure model	✖	✔	
Promotes Re-use	✖	✔	

Table 8 - SIP Servlets vs. JAIN SLEE [24]

3.4 Protocols

In this part of the chapter, some of the protocols relevant to the internship are explained. Complementary details are also provided in Annex A.2.

3.4.1 VoIP Protocols

Several different protocols, with different functions, might be used in Voice over IP. **SIP** is the Internet Engineering Task Force (IETF) standard for signalling, and **RTP** is for real-time transportation of data. However, some alternatives exist on the market.

3.4.2. Session Initiation Protocol (SIP)

As mentioned, SIP is a signalling protocol, which means that it is not actually responsible for transmitting or transporting voice data, rather its purpose is to initiate, negotiate, coordinate and tear down a communication between two endpoints. It is extensively used in the IMS. SIP can also be used for delivering messages, or to initiate an instant messaging session.

This protocol is based on HTTP, functioning like a request-response protocol, independent of the transport protocol. It was first defined on RFC 2543 [25], having suffered several updates and extends on other RFC's. By the time of writing, the latest specification of SIP is on RFC 3261 [26]. There are some concepts associated with SIP that are important to know, like transaction and dialog, concepts that are further explained on Annex A.2.1.

Like HTTP, SIP is a text-based protocol, and messages are composed by an initial line, describing the type of operation, a set of headers, and an optional body. Headers and body are separated by a blank line. An example of a SIP message is presented on Figure 12.

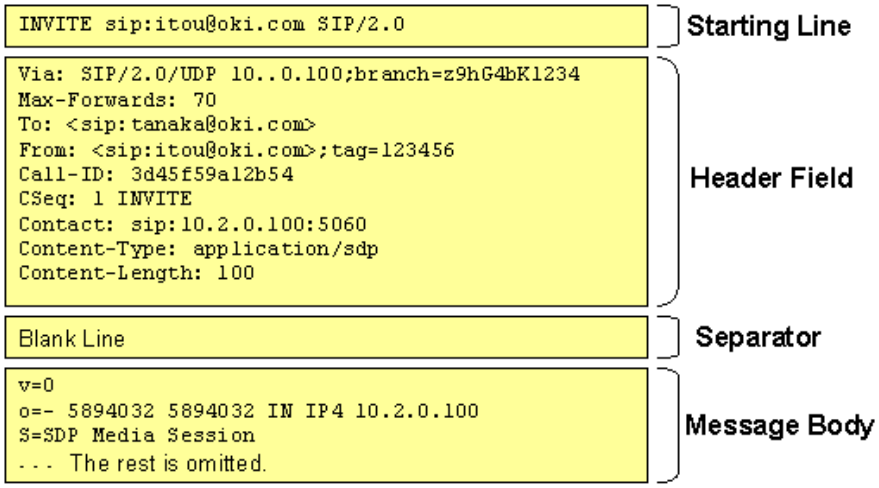

Figure 12 - SIP Message

As mentioned, SIP messages can be Requests or Responses. It supports an extent list of request methods, each with its own functionality. Some are presented on Table 9 - SIP Requests, and a full list of methods is described in Annex A.2.1.

Method	Description
INVITE	Allows a user to invite another to a session. It might be a voice call, video call, and instant message session. The description of these is embedded on the message body using SDP
BYE	Terminates an ongoing session between two users.
OPTIONS	Informs about supported capabilities.
ACK	Informs the other peer that the response was acknowledged.

Table 9 - SIP Requests

There are also a number of possible responses for each request. Those are divided into 6 categories (Table 10), similar to what happens in HTTP Responses.

Response Code	Description
1xx – Provisional	Optional response, informs that the request was received, and it's being processed.
2xx – Success	Request was successfully received and processed.
3xx – Redirection	Server returned possible locations. Client should retry the request to another server.

4xx – Client Error	Request failed due to an error by the client.
5xx – Server Error	Request failed due to an error by the server. If possible, the user should retry to another server.
6xx – Global Failure	The request has failed. It should not be retried to any server.

Table 10 - SIP Responses

In order for the reader to better understand the SIP Request-Response method, Figure 13 illustrates a simple call flow using SIP.

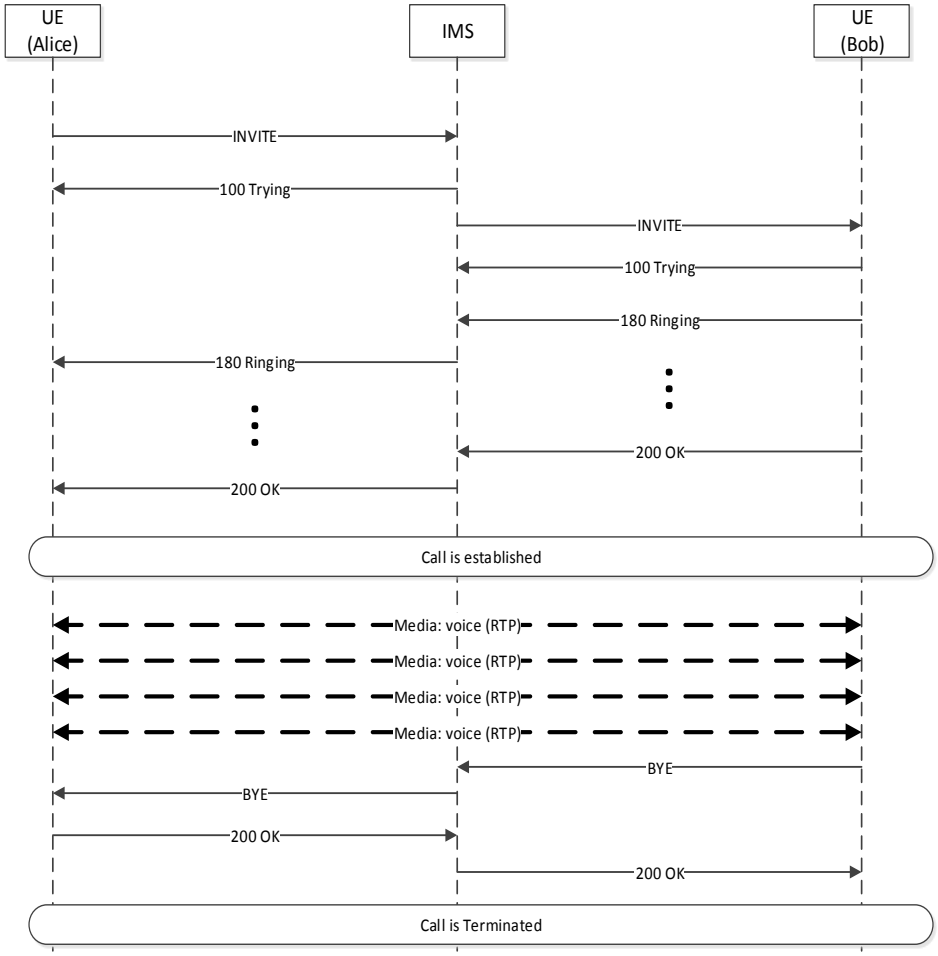

Figure 13 - SIP Invite Flow

It's noteworthy that media traffic (RTP) does not go via the IMS Core.

3.4.3 Alternatives to SIP

Although Session Initiation Protocol is the standard protocol for signalling on VoIP calls, there are some alternatives on the market.

H.323

This is another VoIP signalling protocol standardised by the International Telecommunications Union (ITU-T). It was widely used in the 1990s in Internet applications like "Microsoft NetMeeting" or "GnomeMeeting".

H.323 provides roughly the same kind of functionalities as SIP, but nowadays it is nearly obsolete, as Internet and Telecommunications communities adopted SIP as the standard protocol for signalling. This protocol also uses RTP for data transmission.

Jabber (XMPP)

Specified on RFC's 3920 and 3921, Jabber is an XML (eXtensible Markup Language) based protocol developed in 1998. Originally its scope was only instant messaging, but an extension (Jingle) was developed by Google in order for Jabber to support VoIP calls. Jabber is currently also standardised within IETF, and it is now known as XMPP (Extensible Messaging and Presence Protocol).

XMPP was adopted by Google for use in its Google Talk service, lending the protocol serious credibility. This is the main challenger to SIP, and is considered by some telecommunications engineers as the VoIP protocol of the future. This protocol also uses RTP for data transmission.

Proprietary Protocols

There are also some proprietary protocols on the market, like Skinny Client Control Protocol (SCCP) by Cisco, Skype protocol or Inter-Asterisk Exchange Protocol (IAX2) by Asterisk. An interesting property of the last is that it combines both signalling and voice data into the same stream, transmitted over UDP.

3.4.4. Real-Time Transport (RTP)

Standardised by the IETF, in RFC 3550 [27], RTP is used for real-time transportation of data. In VoIP is mainly used for transmit voice, but it can also transmit any data, such as video. As described before, there are several alternatives to SIP protocol; however, most of them use RTP for data transmission protocol. It is also used in many multimedia streaming solutions.

Due to its many usage scenarios and low-level protocol, RTP is quite a complex protocol. For the reader, it is sufficient to think of it as a data transmission protocol.

3.4.5. Session Description Protocol (SDP)

So far, it is described how to use SIP protocol (or any other signalling protocol) to establish a session with another endpoint, and how media can be exchanged (using RTP protocol). However, it has not been discussed how endpoint exchange user-plane details. Simplifying, both endpoints must exchange a set of information that is needed for sending and receiving media (e.g. ports, audio codecs, etc...). This is the main purpose of Session Description Protocol (SDP). SDP, specified by RFC 4566 [28], is a text-based protocol that is transported in the body of SIP messages, and for that reason follows the offer-answer model. This model ensures that all negotiation is made during the establishment of an SIP session. It's noteworthy that multiple SDP messages can be exchanged until both endpoints reach an agreement (Figure 14).

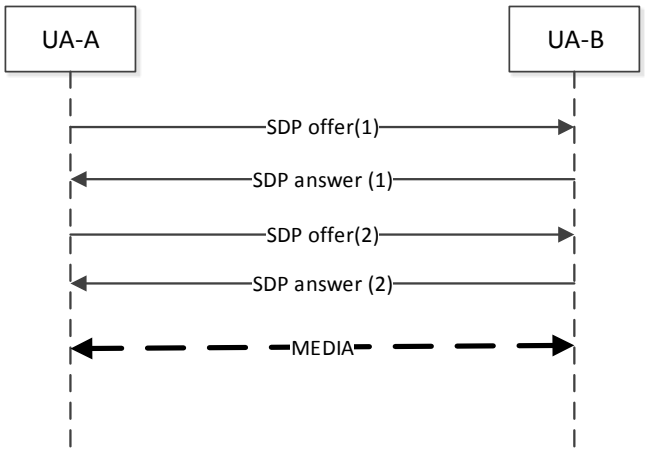

Figure 14 - SDP offer-answer model

Further explanation about SDP protocol and the meaning of all its parameters is given in Annex A.2.2. The next figure (Figure **15**) shows an example of a SDP message.

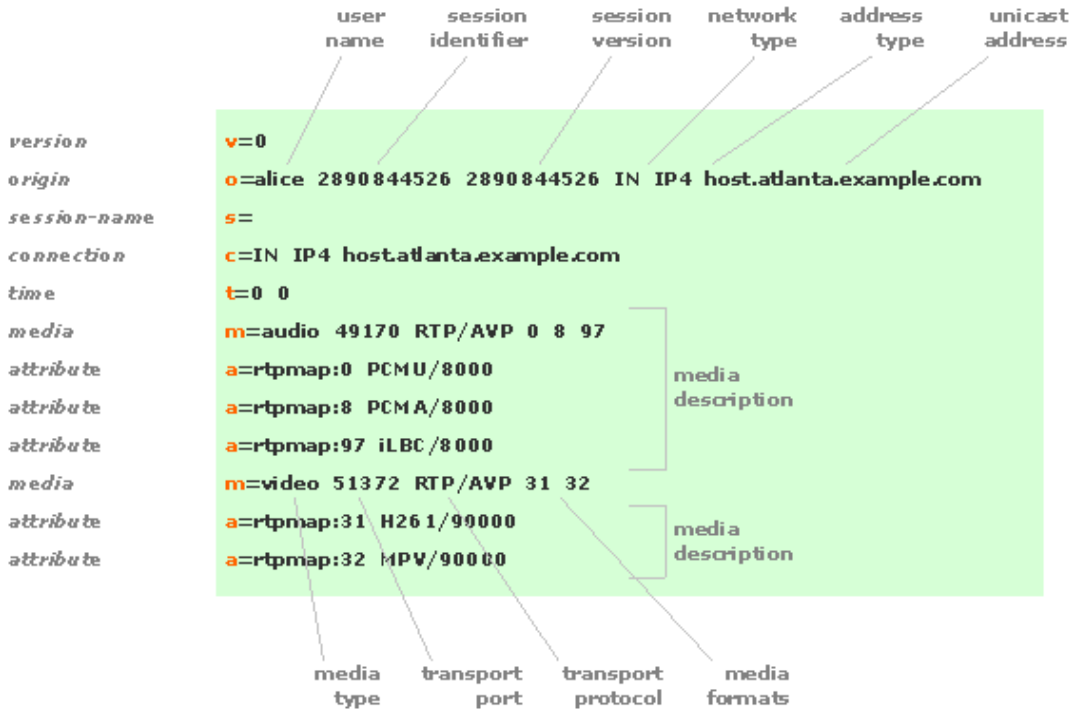

Figure 15 - SDP example [29]

3.4.6 VoIP in IMS

As the reader should have understood by now, in a VoIP call over IMS, SIP is used as the signalling protocol and RTP for data transmission. It is worth to mention that SIP and RTP use different channels in a communication (Figure 16).

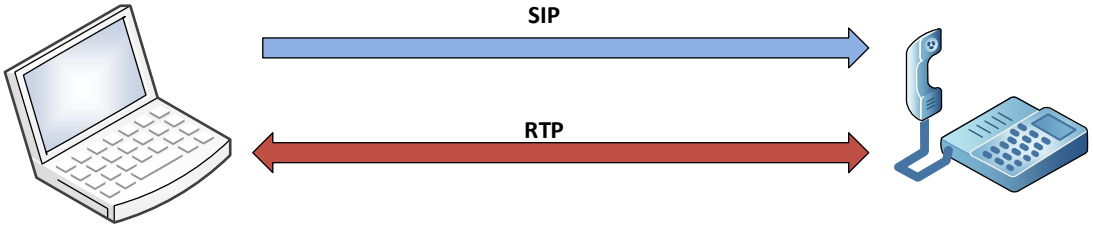

Figure 16 - VoIP over IMS

3.4.7. Message Session Relay Protocol (MSRP)

This protocol was developed in the IETF and it is documented in RFCs 4975 [30] and 4976 [31]. The main function of MSRP is transmitting a series of related instant messages in the context of a communication session.

SIP protocol also has a built in mechanism for sending instant messages: the MESSAGE method, however, in this method, the content of an instant message is carried as the payload in a SIP message, much like SMS. With SIP MESSAGE, there's no connection between one message and another, providing a "pager" like user experience. Also like in SMS, this method has some strict limitations on size and content in a single message.

The main difference between MSRP and SIP MESSAGE is that in MSRP a media session is established between endpoints, similar to how RTP is used for video and audio. This session is negotiated with SIP protocol (using INVITE method), and SDP offer/answer model. Unlike SMS or SIP MESSAGE, MSRP provides a "chat" like user experience.

This protocol is inherently multi-media. It can carry any arbitrary type of content, and has no built-in size limitations, although must of operators specify a limit for MSRP messages size. Even though it was originally designed with messaging in mind, it's really a generic content transfer mechanism.

In RCS context, MSRP is used for the instant messaging, file transfer, and image share features.

3.4.8. Media Gateway Control Protocol (MGCP)

Media Gateway Control Protocol (MGCP) is a protocol developed by IETF that enables a call agent (e.g. AS) to manipulate media gateways, allowing it to create, delete or modify multimedia communication sessions to endpoints. This protocol uses SDP to negotiate and modify media streams with the gateway. This protocol is specified in [32], and is an evolution of the Simple Gateway Control Protocol (SGCP). Figure 17 illustrate how MGCP will be used in this project.

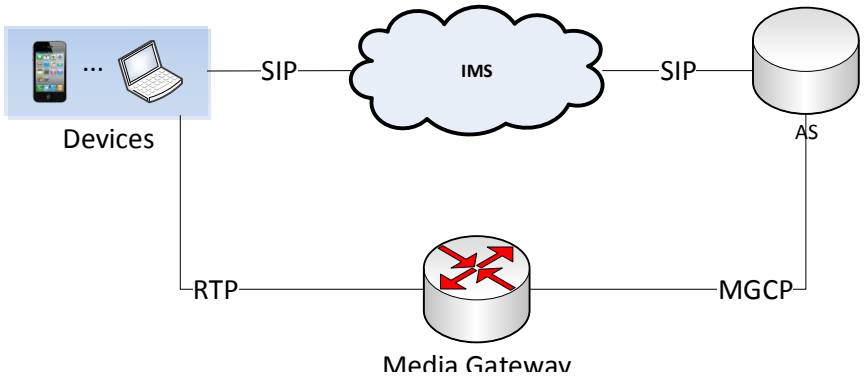

Figure 17 - MGCP example

3.4.9. Alternatives to MGCP

There are several media control protocols available on the marker: MSCML over SIP, VXML, JSR-309, XMLRPC, and others, but the most successful are MGCP and H.248 (both are IETF standards).

H.248, also known as MEGACO, can be seen as an evolution of MGCP and Media Device Control Protocol (MDCP) as seen in Figure 18 and specified in RFC3525 [33]. Although, H.248 is much more complex than MGCP, many providers of Media Servers are reluctant in adopting it as the main control protocol, and many companies use their own variation of H.248 or MGCP.

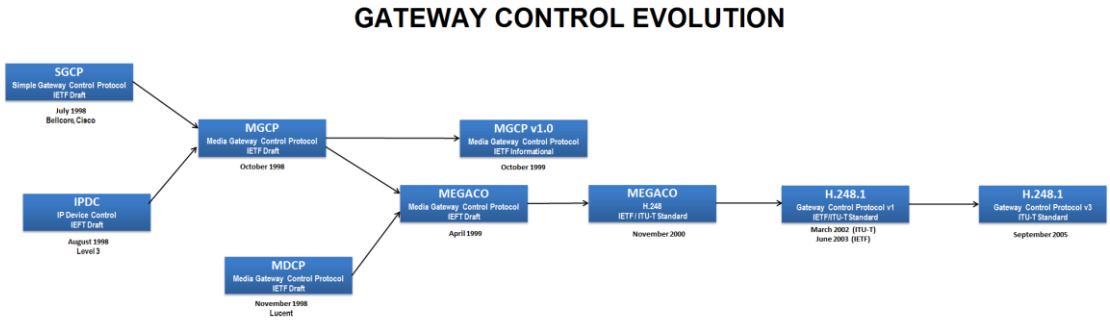

Figure 18 - Gateway Control Evolution [34]

Chapter 4
System Description

4.1 General Objectives

As previously explained, the main goal of this project is the development of an Advertisement Service. This service broadcast Image and Video ads before and during calls (use cases specified later). It is noteworthy that the service must not change any of the users' capabilities, like video and image share. Also, this service must follow RCS specifications [5], for capabilities discovery and image/video sharing. All software used in this project should be based on open-source or free solutions. The developed solution must be deployed on an IMS network as an application server. In the next sections, further detail is given about the project, as well as the technical requirements.

4.2 Requirements Analysis

During the development of this service, there are some functional and non-functional requirements that should be achieved. They serve as a guideline for the internship, and can also measure the success or failure of the internship. Requirements can also be divided into four categories, Configuration, Application Server, BackOffice and Codecs.

Configuration

Description	Semester	Achieved	Priority
Configure OpenIMS	1st	✔	Must-Have
Configure RCS client (joyn)	1st	✔	Must-Have
Create and configure accounts on HSS	1st	✔	Must-Have
Configure IFC for Options Request	1st	✔	Must-Have
Configure IFC for Invite Request	1st	✔	Must-Have
Integrate 3rd party MRFP (Media server) node with OpenIMS	2nd	✔	Must-Have

Table 11 - Configuration Requirements

As the reader should have noticed, all the requirements of configuration are "must-have". This is due to the fact that these requirements must be achieved before starting the development of the Application Service (except with the media server integration, which will only be necessary in the video share).

Application Server

Description	Semester	Achieved	Priority
Develop logging system	1st	✔	Nice-to-Have
Capability Discovery System (via Options Request)	1st	✔	Must-Have
Integrate 3rd party MSRP stack in AS	1st	✔	Must-Have
Image Share via MSRP	1st	✔	Must-Have
Use case: Originating user receives 3rd party image share during on-going call	1st	✔	Must-Have
Use case: Terminating user receives 3rd party image share during on-going call	1st	✔	Must-Have
Use case: Originating user receives 3rd party image share during call setup	2nd	✔	Must-Have
Use case: Originating user receives 3rd party image share during before call	2nd	✔	Nice-to-Have
Video Share via MRFP (Media Server)	2nd	✔	Must-Have
Use case: Originating user receives 3rd party video share during on-going call	2nd	✔	Must-Have
Use case: Terminating user receives 3rd party video share during on-going call	2nd	✔	Must-Have
Use case: Originating user receives 3rd party video share during call setup	2nd	✔	Must-Have
Use case: Originating user receives 3rd party video share before call	2nd	✔	Nice-to-Have
Retrieve image campaigns from "Back-Office"	2nd	✔	Nice-to-Have
Retrieve video campaigns from "Back-Office"	2nd	✔	Nice-to-Have
Statistical Information BO	2nd	✖	Nice-to-Have

Table 12 - Application Service Requirements

Not all of the use cases are classified as "must-have" requirements due to the fact that they are variants of "image share" or "video share", and it is more important to have two or three use cases of each working, than to have all image share use cases working, and none of the video share.

Back Office

Description	Semester	Achieved	Priority
Image Share campaigns	2nd	✔	Nice-to-Have
Video Share campaigns	2nd	✔	Nice-to-Have
Login mechanism	2nd	✔	Nice-to-Have
Usage statistics	2nd	✖	Nice-to-Have

All of back office requirements are considered "nice-to-have" due to the fact that in the scope of this internship is more important to have a fully functional advertisement service, than to have a nice Back-Office mechanism.

Codecs

There are several codecs available in the market, like detailed in Annex A.3. However, in the telecommunications industry, there are some codecs more used than others, and are considered as standard codecs for endpoint manufactures. For this reason, this codecs are considered as Must-Have in the media server (chapter 2.5.1 Solutions Available).

Those codecs are:

Audio:

- G.711
- GSM

Video:

- H.263
- H.264

Non-Functional Requirements

There are also some non-functional requirements that should be achieved. Those requirements are described in Table 13 - Non-Functional Requirements.

Name	Description	Achieved
Standard Protocols	Use standard and well established protocols on the industry	✔
IMS 11	Solution must be compliant with the IMS Release 11.	✔
RCS 5.1	Solution must follow RCS 5.1 specifications in Capability discovery and Image/Video share	✔
High-Throughput	Service should handle a wide number of users.	✖
Free/Open-Source	All software used must be free or Open-source	✔

Table 13 - Non-Functional Requirements

4.3 Methodology

At WIT Software, the default **Agile** methodology used during the development of software products is **Scrum** [35]. However, due to the fact that there was only two persons involved during the internship (the Development Team was only composed by the author, and the role of Product Owner and Scrum Master was taken by the company supervisor, Eng. Rafael Maia.), a **different approach** of Scrum was used. Detailed information about Scrum is given in Annex A.4.

In the methodology approach that was used, **Sprint** is the basic unit of development (same as Scrum), and the duration of each sprint is fixed in advance. Each sprint is preceded by a planning meeting, where tasks are identified and an estimated commitment for the spring goal is made.

Using Scrum, there is a **Product Backlog**, an ordered list of requirements and tasks to be done during the development of the project. This Backlog is ordered by the Project Owner. Each Sprint must also contain a backlog. In the approach taken, there is **no Product Backlog**, the next task to be done is defined by the original Planning (1.5). This approach is similar to what happens in the **Waterfall** model [36].

Like in Scrum Methodology, at the end of each day, a brief meeting takes place between the Development Team (the author), and the Scrum Master (company supervisor) in order for the last one to understand if there is any problem preventing the developer from reaching the spring goal. These meetings were facilitated by the fact that both actors involved in this project worked side by side.

4.4 High Level Architecture

Figure 19 illustrates the required high level architecture.

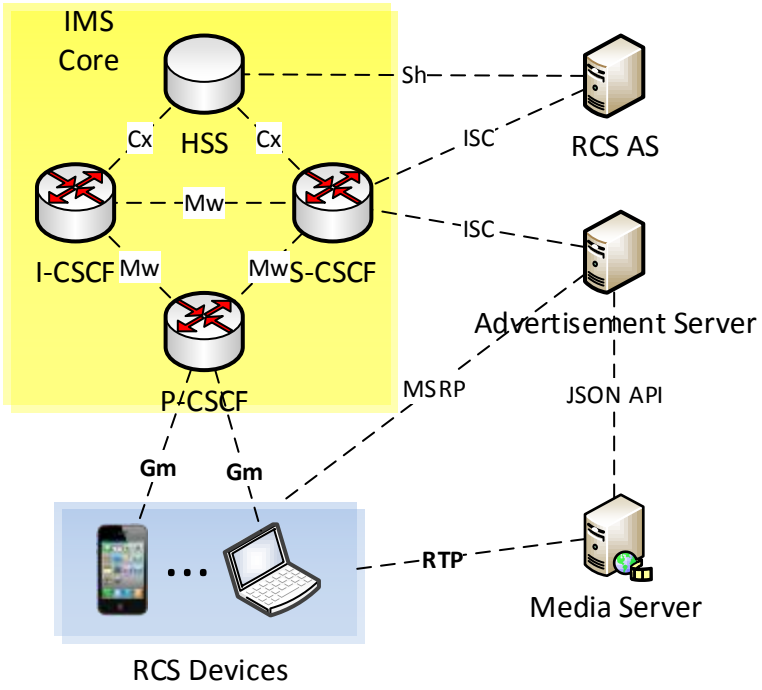

Figure 19 - High Level Architecture

An RCS solution is usually deployed over IMS architecture (IMS Core), with the RCS functionality being offered by one or more application servers deployed in the IMS Service layer. For simplification, in the above figure, all RCS functionalities are assumed to be offered by a single application server. All interfaces presented (Cx, Mw, Gm, Sh, ISC) are IMS reference points and are further detail in 3.3.2 IMS Core. RTP and MGCP are protocols already described in chapter 3.4 Protocols.

4.5 Advertisement Service

Like already explained, the main goal of this project is to develop an Advertisement Service which provides image and video ads during RCS VoIP Calls. This service will be deployed on the IMS Service layer as an application server.

This **Application Server (AS)** needs to be in the signalling path of SIP Options, in order to be aware of capabilities of all endpoints, and other SIP messages (e.g. Invite, Bye, etc...), so it can manage all calls. In the IMS architecture, an application server can be added to the signalling path by configuring a trigger point in the **Initial Filter Criteria** (IFC) of a subscriber (more detail in Annex A.1.1.4).
It is also responsibility of the application server to determine when and to whom it can send ads, in order to do that, the AS has to connect and manage a media server. Then, the media server will connect directly to the endpoint and send the content over **RTP** (3.4.4. Real-Time Transport (RTP)).

A high level architecture of the **Application Server** is presented in Figure 20.

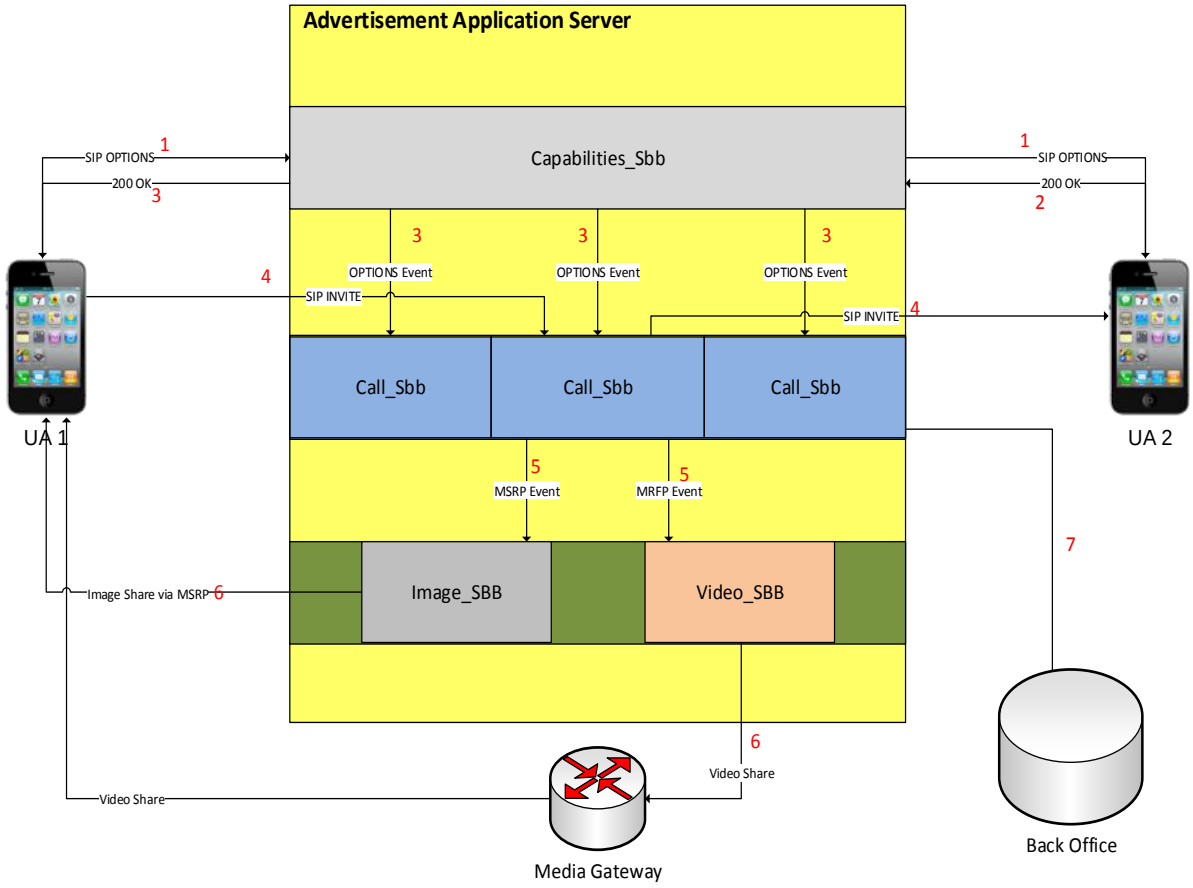

Figure 20 - Advertisement High Level Architecture

The Advertisement Application Server is divided into 3 layers, composed of SBBs - Service Building Blocks (logical representation of a re-usable component). The **capabilities layer** is responsible for the capability discovery (detailed in chapter 4.5.1). The **Call layer** is responsible for the handling call requests and responses, and to decide when to send advertisement. The **Image/Video layer** is responsible for actually send media content to Endpoints. In order for the reader to better understand the messages flow within the AS, a brief explanation is given:

1. Originating user (UA 1) sends a **OPTIONS** Request to Terminating user (UA 2)
2. Because an IFC is configured in the HSS, **OPTIONS** are redirected to the Advertisement AS and handled by the **Options layer**. This layer processes the request, saves the capabilities of UA1, and forwards the request to UA2.
3. When the **200 OK** Response arrives at the Capabilities layer, the SBB responsible for it handles the Response, and saves the capabilities of UA2. Then, it sends an **Event** (Options Event) to all the Call Sbbs (Call layer), with the capability of both users. These SBBs in the Call layer process the event, and check if the users in the event are the ones it is responsible for, if not, it discard the event. The Call Sbb responsible for handling the call between UA1 and UA2 process the Event, and saves (or updates) the capabilities of both users. At this point the SBB is aware if users can receive image or video share.
4. When an **INVITE** Request arrives at the AS (again, because an IFC is configured in the HSS), a new Call_Sbb is created, and becomes responsible for handling that call. If the Request is a Re-Invite, the Call_Sbb already assign to handle the call process it.

5. When the Call_Sbb responsible for a call decides that advertisement can be sent to User Agents, it sends one of two possible events, **MSRP** or **MRFP Event**, respectively for sending image and video. This event has all the information necessary (SDP) for the **Image/Video** Sbb to send advertisement.
6. In image share, the Image_Sbb sends the image share directly to the UA. In video share, the Video_Sbb communicates with the Media Server, and this server sends the video share to the UA.
7. Through all this process, the Call layer might exchange information with the Back-Office, in order to know which advertisement to send, or to add usage statistic information.

The system and the methods described by this invention can be split in two different phases.

4.5.1 Capability Discovery

As previously described, all features in this project must follow RCS 5.1 specifications. This document specifies two mechanisms for discovering the capabilities of endpoints, the SIP OPTIONS exchange, and Presence server. As explained on chapter 3.2.1 Capability Discovery, the method chosen for this project is **SIP OPTIONS exchange**.

In this mechanism, an endpoint A can send a SIP OPTIONS message to an endpoint B in order to find which capabilities are supported by this endpoint. When endpoint B receives the message, it can look at the "**Accept-Contact**" header to find out which capabilities are supported by endpoint A. Bob can then respond with a SIP status error code (if neither of the capabilities are supported, usually if non-RCS client), or with a **200 OK**. In the last case, endpoint B lists the supported capabilities in the "**Accept-Contact**" header. Figure 21 illustrates a SIP OPTIONS message, and Figure 22 describes the flow of the capability discovery process.

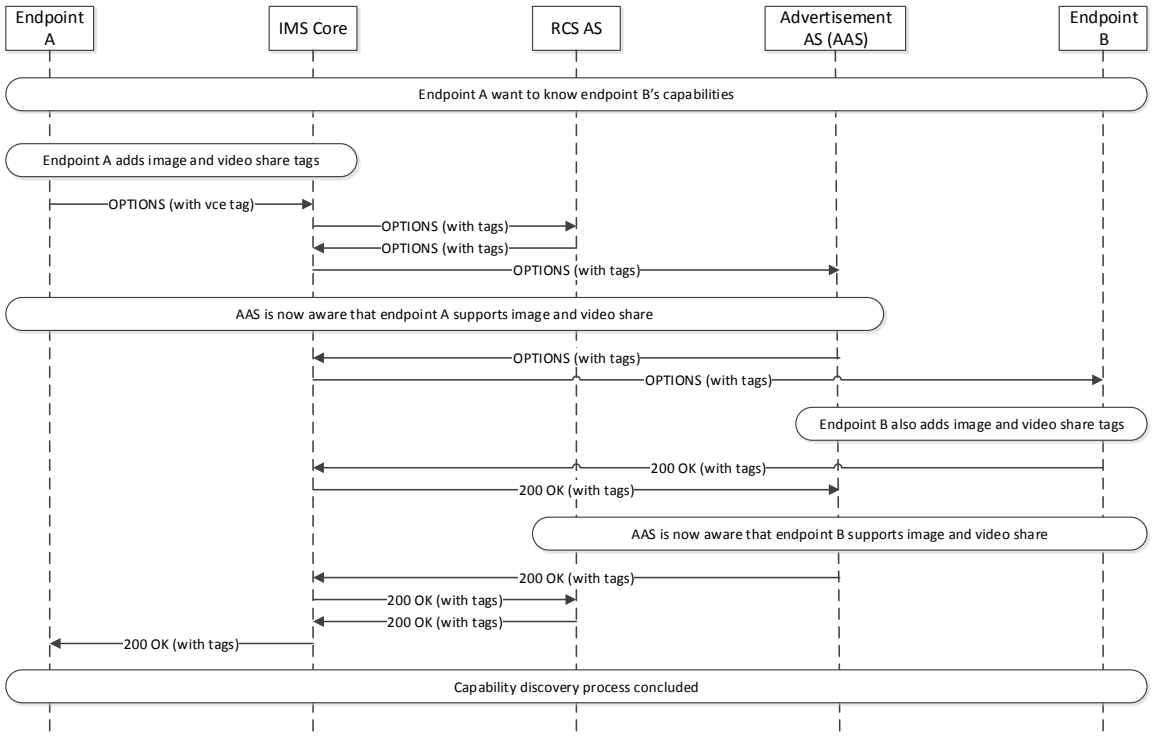

Figure 21 - Capability discover

```
OPTIONS sip:alice@open-ims.test SIP/2.0

From: <sip:alice@open-ims.test>;tag=dZPlDOWwdcoEntTQK
To: <sip:bob@open-ims.test>
Cseq: 23098 OPTIONS
●
●
●
Contact: <sip:alice@open-ims.test>;+g.3gpp.iari-ref="urn%3Aurn-7%3A3gpp-
application.ims.iari.gsma-is";+g.3gpp.cs-voice
Accept-Contact: *;+g.3gpp.iari-ref="urn%3Aurn-7%3A3gpp-application.ims.iari.gsma-
is";+g.3gpp.cs-voice
```

Figure 22 - SIP OPTIONS example

As shown in the figure above, the "**Accept-Contact**" header contains a list of tags. A list of all possible tags is presented in Annex A.4.1, however, for the purpose of this project the most important tags are described in Table 14 - RCS Tags.

RCS Service	Tag
Image Share	+g.3gpp.iari-ref="urn%3Aurn-7%3A3gpp-application.ims.iari.gsma-is"
Video Share during a call	+g.3gpp.cs-voice

Table 14 - RCS Tags

4.5.2 Content Transfer

The 3rd party content transfer can only start once the **capabilities discovery** method is **completed** and the Application Server is aware of endpoints capabilities. As mention before, the AS is aware of those capabilities because it is in the signalling path and monitoring the capability discovery process. After this process is completed, when an endpoint start a voice call, the AS uses the mapping to check if either the peers supports image or video share. If any of those capabilities is supported by one, or by both of the endpoints, and that channel is not in use (i.e. no peer to peer image or video share), then the AS can send content to the endpoints. The following figures illustrates in which cases the AS can or cannot send content.

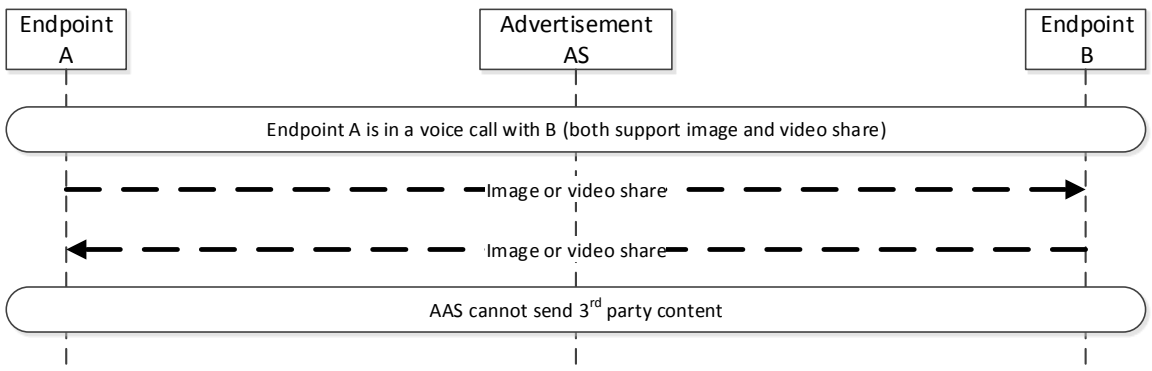

Figure 23 - 3rd Party Content Share 1

In this case, the endpoints are in a voice call, and they are sharing content with each other, so the AS cannot send content to any user.

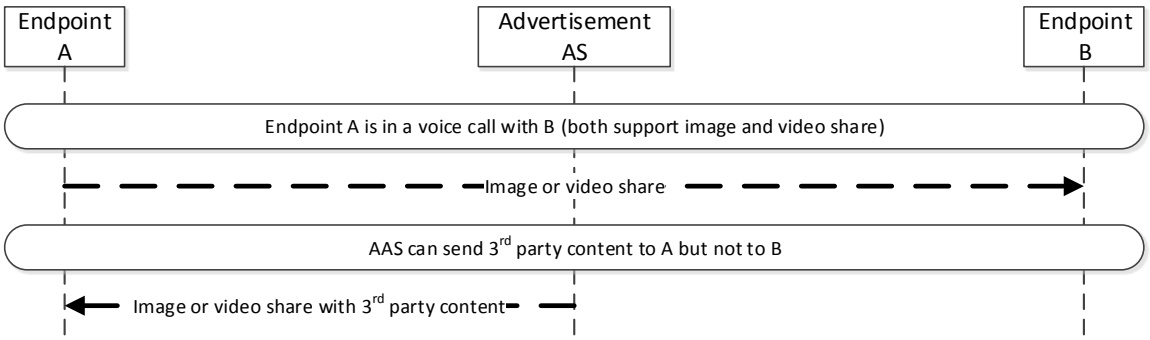

Figure 24 - 3rd Party Content Share 2

As already mentioned, RCS 5.1 [5] specifies that image and video share are bi-directional. This means that even a user is sending content to another, he or she can still receive image/video share from that same user (or in this case, by an AS in the signalling path).

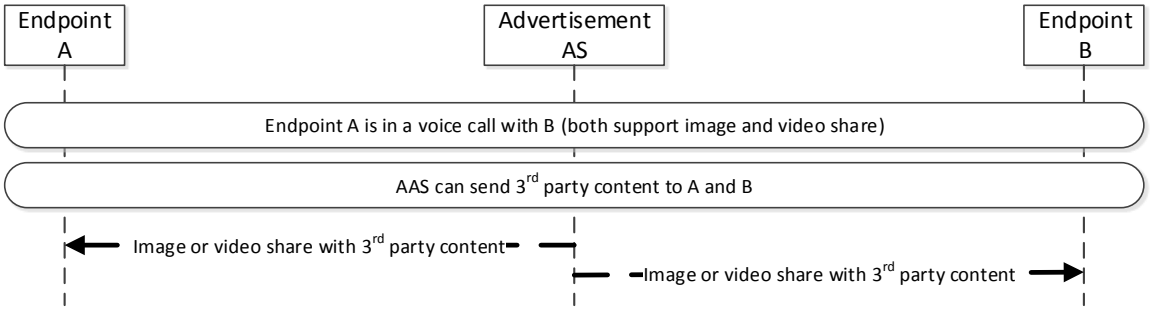

Figure 25 - 3rd Party Content Share 3

In this case, the more common, endpoints are in a voice call, but any of them is using the video/image share capability, however, both of them support it. The AS, in this case, can send content to both endpoints. It is noteworthy that the content can be different to both endpoints.

4.5.3 Use Cases

In this section of the chapter, the required use cases are described and their respective flow is illustrated. There are 3 main use cases:

- Media Sharing **before** a call is established
- Media Sharing **during** call **setup**
- Media Sharing **during** call

It is assumed that both endpoints use RCS clients.

Media sharing before a call is established

In this case, the service must send advertisement to a user before the call is established. This means that the Advertisement AS deployed on the IMS network and in the signalling path of the communications must intercept the INVITE request sent by user A (Alice), answer it on behalf of the destination user (Bob), and then start a media share session with Alice. After the media content is transferred to the Originating user agent (Alice), the service will

start a new call with the destination user of the original invite (Bob). After this call is successfully established, the AS sends a **RE-INVITE** (containing the SDP of Bob) request to Alice; this method informs this user that he/she should change the session information in order to meet the call to another user. After this, Alice and Bob will be in an active call.

The following figure illustrates the message flows between the users and the AS. It is noteworthy that in the case of video sharing, the AS communicates with the Media Gateway, and it is this that sends the video, but in order to simplify, the reader can understand the Advertisement Application Server as the full service (SIP AS + Media Gateway).

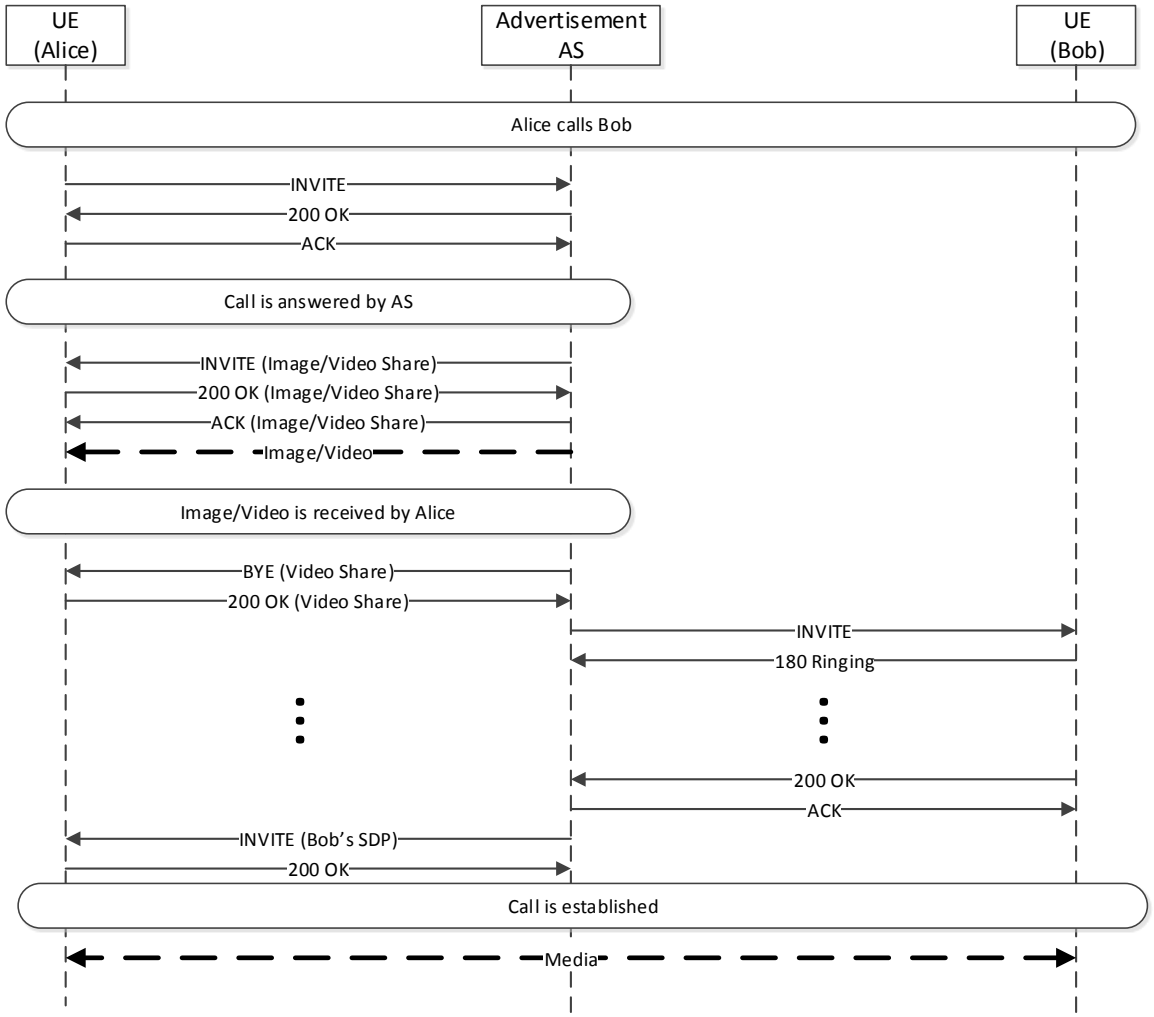

Figure 26 - Use Case 1 - Image share before call to Originating user

Media sharing during call setup

In this use case, the Advertisement Service does not answer the call from the originating User Endpoint (Alice) on behalf of the terminating UE (Bob), but instead, it pushes the media content (Image or video) during the call setup. In other words, it sends the media during the time that the terminating user does not answer the call (Ringing). If the terminating user answer the call before the media content is transferred, it is cancelled, and the communication between users is established. Figure 27 illustrates the message flow between all the actors involved.

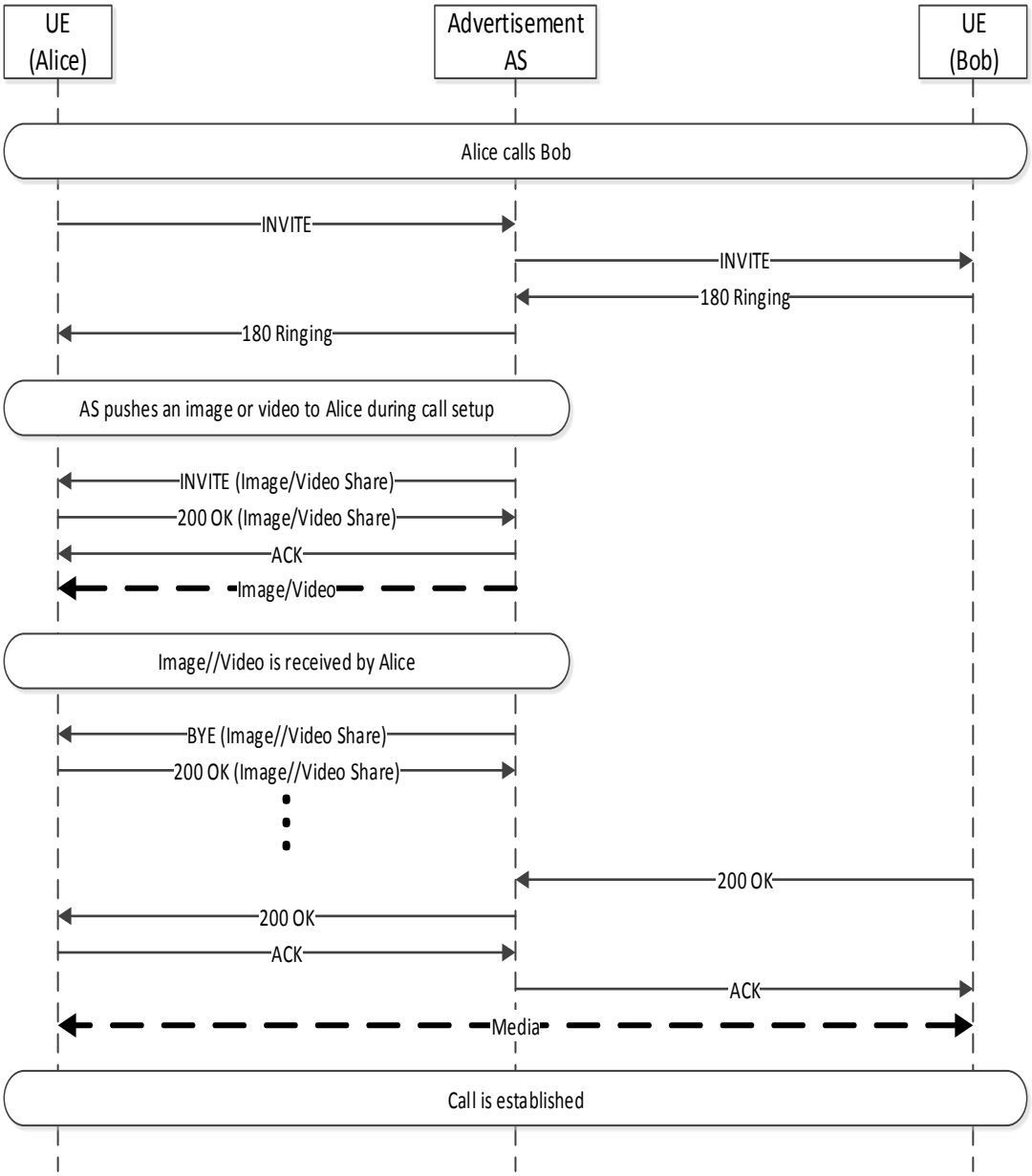

Figure 27 - Use case 2 - Originating user receives Image during call setup

Media sharing during call

In this case, the involved users are already in an active call. The Advertisement service must then realize if, and to whom it can send media content (like described in

4.5.2 Content Transfer). Figure 28 illustrates how the Advertisement Service can send image or video content to users during an active call, in this case to UA Alice, and how the service handles the sharing of media content between users on the call (in this case Bob sends content to Alice).

Like the reader should remember, the Advertisement AS is in the signalling path of the call established between both users, so, it can send (when possible) media content to any of the users involved. In the first part of Figure 28 Alice and Bob are in an active call, but are not exchanging any media content, so, in this case, the AS starts a image or video share with Alice (it could also start a share with Bob) by sending an INVITE request. After the user

accepts this invite, the content is pushed to Alice. It is noteworthy that the audio call between Alice and Bob is not affected by this sharing session, and stays active.

In the second part of the figure, we can see that Bob is trying to share an image/video with Alice, so, the Advertisement AS cannot send any media content to Alice (but it can send to Bob, like explained in 3.2.2 Content Sharing). After the sharing session between both users is concluded, the AS is now free to send media content to Alice. If Bob's INVITE is sent before the sharing session between Alice and the Advertisement service is over, the AS must cancel the first session, in order to maintain all the original sharing capabilities of users.

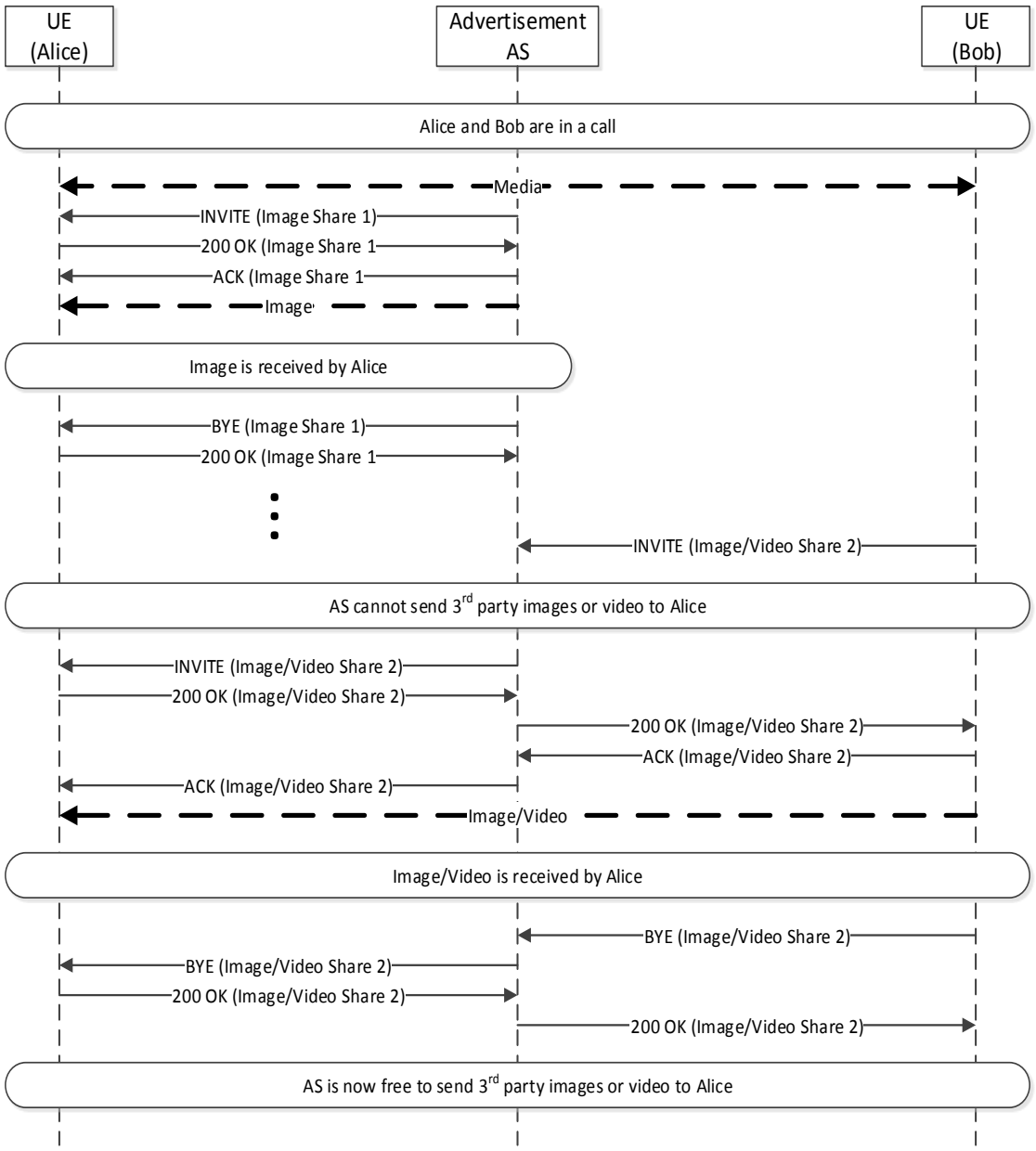

Figure 28 - Originating User receives image during on-going call

4.5.4. Technologies

In this part of the chapter, a summary of all technologies that are going to be used is presented. It is noteworthy that besides this are the chosen technologies, during the development of the project, some changes might occur.

Technologies	
Network Architecture	IP Multimedia Subsystem (IMS)
IMS Core	OpenIMS
IMS Home Service Subscriber	OpenIMS-FHoSS
Application Server Development technology	Mobicents JAIN SLEE
Media Server / Media Gateway	Kurento Media Server
Rich Communication Services Client	WIT Software – joyn Android
Protocols	
Signalling	Session Initiation Protocol (SIP)
Description	Session Description Protocol (SDP)
Transport	Real-Time Transport Protocol (RTP)
Gateway Control	Kurento Media Server API (JSON)
Image Share	Message Session Relay Protocol (MSRP)

Table 15 - Technologies Used

Chapter 5
Development

In this chapter all the components of the Advertisement Service that were developed by the author during the internship are detailed, as well as the integration between them. In order for the reader to fully understand the purpose of each component, it is used a top-down approach, in other words, this chapter starts with a high level architecture of the service, and then focus is given to each of the components involved.

5.1. Virtual Machines / Hardware

Before starting the analysis of all the components developed and integrated into the service, it is important for the reader to understand the hardware, or virtual Machines (VMs), used during the internship.

The architecture was designed so that the full service can be deployed in a single machine (or virtual machine). This option simplifies the deployment and integration with any IMS network, as well as minimizes the usage of network resources.

However, in practice, were used two VMs. This option was taken because the company already have a VM with an IMS network (OpenIMS) installed, but not with enough resources to support all necessary components [37], so instead of spending time installing all the components in a new VM, it was chosen to use two different Virtual Machines for this Prove Of Concept (POC). Although this was the approach taken, the service is prepared to be installed as a whole with minimal changes.

Virtual Machine A

This is the Virtual Machine provided by the company, with **OpenIMS** and **JBoss 5.1.0.GA** installed. Table 16 summarizes the resources available on this machine.

Type	**Resource**
CPU	1 Core (Unknown CPU)
Memory	2 GB DDR3 1333 MHZ
Storage Size	16 GB
Operating System	Ubuntu Linux 10.04 TLS

Table 16 – Virtual Machine A

On this machine, the author had to install the **container** (3.3.4. Application Server Development Technologies) and, in order to direct the SIP Messages to the correct SIP Application Server, configure an **Initial Filter Criteria** on the OpenIMS network.

Virtual Machine B

This new VM was created by the author because the Media Server chosen (2.5. Media Server / Media Gateway) has some prerequisites that the VM-A does not support. Table 17 - Virtual Machine B the resources allocated on this machine.

Type	Resource
CPU	2 Cores (Unknown CPU)
Memory	8 GB DDR3 1333 MHZ
Storage Size	16 GB
Operating System	Ubuntu Linux 13.10

Table 17 - Virtual Machine B

On this machine the author installed the Kurento Media Server (5.7. Media Server) and the Database (5.5. Database).

5.2. High Level Architecture

As already said, in this chapter it is used a Top-down approach, so, the reader starts with the high level architecture (Figure 29 - High Level Architecture) of the service, and then the low-level and details of each component are explained in the next sections.

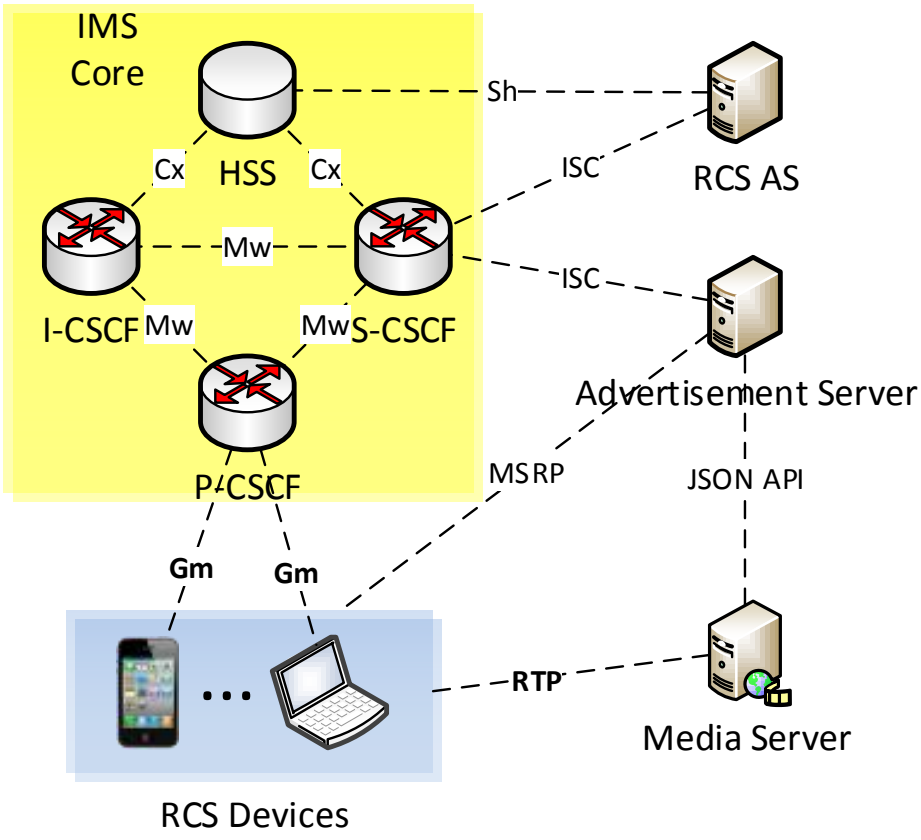

Figure 29 - High Level Architecture

5.3. IMS Core

The function of this component is detailed in chapter 3.3.2 IMS Core and this component was already installed in Virtual Machine (VM-A) that was provided by the company.

In this component, the author only had to configure the **Clients Identities** (Annex A.1.1.1/2) and an **Initial Filter Criteria** (Annex A.1.1.4). Clients Identities and IFCs are stored in the HSS, and are configured in an XML file; however, instead of writing the file, OpenIMS have an Administration BackOffice to simplify this process (Figure 30 - Client Configuration).

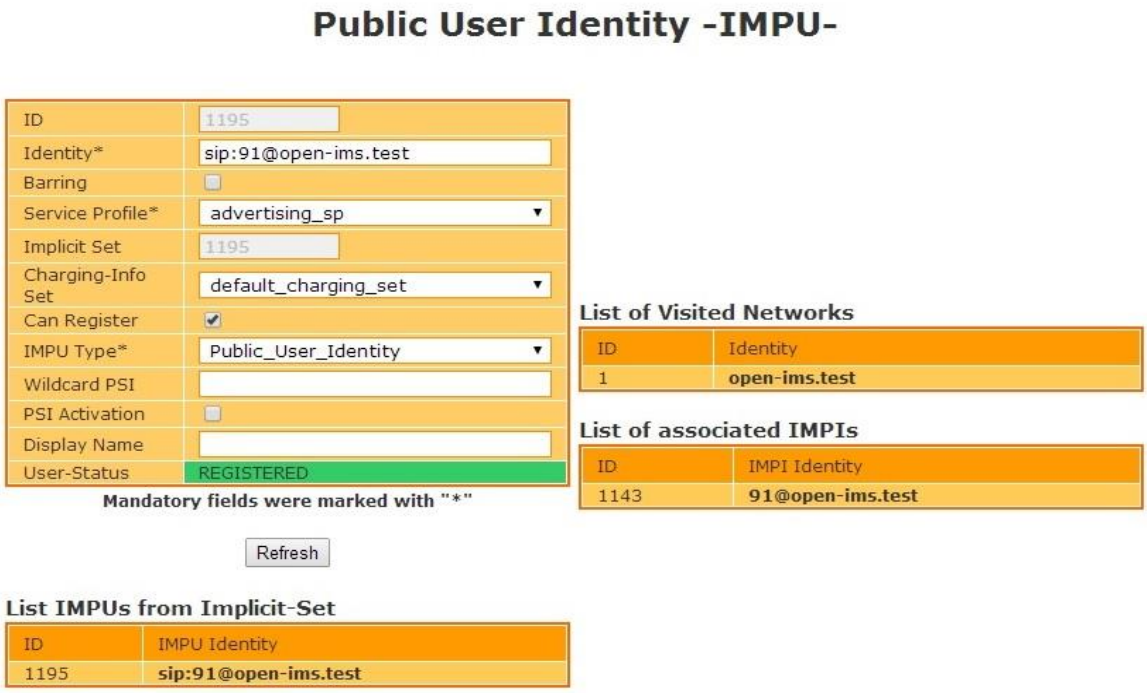

Figure 30 - Client Configuration

5.3.1. Initial Filter Criteria

During the development, the author configured **two** IFCs.

As explained, the Advertisement Service needs to be in the Signalling path of all Calls. In order to achieve it, the author had to configure an IFC that is triggered when any new SIP INVITE arrives to the network - Figure 31.

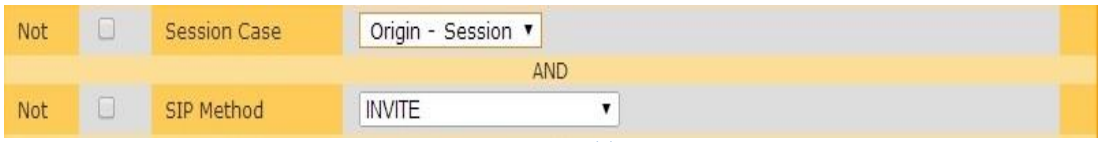

Figure 31 - INVITE IFC

The second IFC is related with the **Capability Discovery Mechanism**. As explained in chapter 3.2.1 Capability Discovery, the service must be aware of Endpoints capabilities. This is achieved by creating a new IFC that is triggered whenever a new SIP OPTION arrives to the SCSF responsible for any of the users configured to this IFC - Figure 32 - OPTIONS IFC.

Figure 32 - OPTIONS IFC

5.4. Endpoints

One of the requirements of the service was that it should follow RCS 5.1 Specifications [5].

It was used the RCS Client developed by the company – **joyn** – configured according to IMS network configuration in 5.3. IMS Core This was achieved by changing a few parameters on the configuration file of the application.

5.5. Database

The Database of the Advertisement Service is used to store the information of all Advertisement Campaigns. BackOffice users can edit their campaigns through the BackOffice Website (5.6. BackOffice).After the campaigns are configured, the Advertisement Server (5.8. Advertisement Server) communicates with the Database in order to know which campaigns are active, and which media should be sent.

5.5.1. Database Management System

The database used does not have any special requirement (procedures, triggers, etc...), so any open-source Database Management System (DBMS) could be used. The solution chosen was **PostgreSQL** [38], and the only criterion for this choice was the previous experience of the author with this DBMS.

The installation of the DBMS was made in **VM-B** following the official documentation [39].

5.5.2. Database Entity-Relationship Model

The database was then configured according to the Entity-Relationship Model (ER) shown in Figure 33 - Database ER. The ER was design using Crow's foot notation [40].

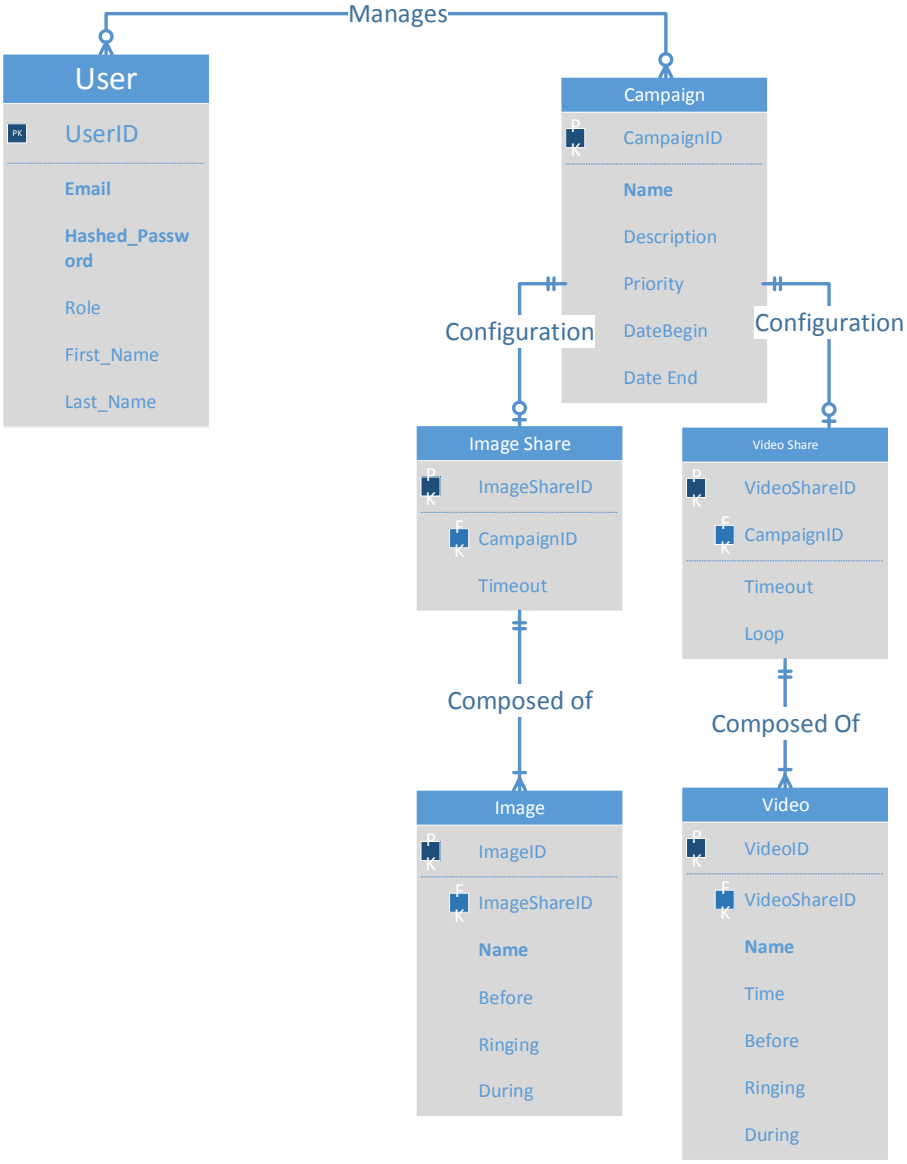

Figure 33 - Database ER

The Database works as following:

- A **User** represents a BackOffice user. This User has access to the BackOffice website and can create/edit/delete Advertisement **Campaigns**. If this user is an Admin, he can edit any campaign and also add/delete new users.
- A **Campaign** has some configuration attributes such as **Priority** (campaigns with high priority have more probability to be chosen), the **Period** when the campaign is active (between **date_begin** and **date_end**) and the **Timeout** (time, in seconds, between advertisements).
- **Image/Video Share –** this entities make the connection between a **Campaign** and its **images/videos**. They also store the information about when (4.5.3 Use Cases) to send the each type of media.
- **Image/Video –** Represent an advert to be sent to the endpoints.

5.6. BackOffice

The service developed during the internship contains a simple **BackOffice** (BO) Website that allows authenticated users to **create** and **manage** Advertisement **Campaigns**. It is noteworthy that the website developed is temporary, and not the final product, this option was made due to the fact that the development of the final product would be very time consuming. The design of the final product was done by the company Designer ***António Lopes*** and is presented in Annex B.

The main objective of the **Demo BackOffice** is to test the connectivity between all components, and use cases. Another important goal was to develop a **Java Class**, to be used in the final product, that contains all the necessary **methods** to **access** and **manage** the Database.

As the user can see in the High level Architecture (Figure 29 - High Level Architecture), the connectivity between the BackOffice and the Database is made using the **Java Database Connectivity API** (JDBC). The BackOffice was deployed in **VM-A**'s JBoss, in order to not install another Application Server.

More information about the Demo BO, and some screenshots are presented in Annex A.5

5.7. Media Server

As previous explained (2.5.1 Solutions Available), the Media Server chosen was the **Kurento Media Server** (KMS) [18].

Although during the beginning of the Development, it was decided to use MGCP (3.4.8. Media Gateway Control Protocol (MGCP)) to control the Media Server, during the month of **April** (2014), the Kurento developers [18] released a **JSON API** [41] that simplifies the control of KMS. The option to **switch to from MGCP to JSON API** was due to the fact that one of the requirements of the Internship is to use Standard protocols (4.2 Requirements Analysis). Switching to this API also allows the future develop of the Advertisement Service to make use of all the new features that Kurento releases in between.

KMS was installed in VM-B following the official documentation [37].

After installation, some configuration files had to be chanced in order for the Media Server to accept connections from the Visual Advertisement Server (VAS). However this is unnecessary if both the Media Server and the VAS are deployed in the same machine.

It was also necessary to configure a **default Session Description Protocol** message (3.4.5. Session Description Protocol (SDP)). This SDP message contains information about the codecs that should be used by the Media server. It will be used by the AS during the negotiation phase and by the Media Server during the transcoding of videos.

The SDP message used was as follows:

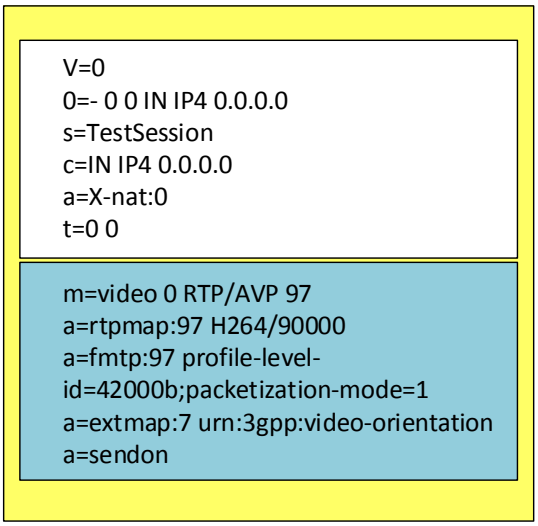

Figure 34 - Kurento SDP

The white box contains session information, Kurento is responsible for changing the *IP4 0.0.0.0* to the correct address. In the blue box, there is information about the codec used in video transcoding, in this case only H.264 is configured. More information about SDP Protocol and the meaning of each attribute is given in Annex A.2.2.

5.8. Advertisement Server

The **Advertisement Server (AS)** is a SIP Application Server and the most important component in the system; it is the core element of the internship. This component has all the business logic of the system, and makes the connection between the service developed during the internship, and the IMS network.

The main responsibilities of this server are:

- Capability Discovery Mechanism
- Call Management
- Communicate with Database
- Image Share
- Control Media Server – Video Share

5.8.1 Communication

The AS is connected to the **Database** through **JDBC**, and it controls the **Media Server** by the **Kurento Media API**, which will be explained later in this section. The connection to the **IMS Network** is made over the **ISC** (3.3.2 IMS Core), this interface, which is defined by the IMS release 11 specifications (3GPP, uses SIP as communication Protocol.

5.8.2 Architecture

Due to the fact that the AS is the core element of the system, and because it interact with a few other elements, this section of the chapter will detail which modules compose the AS, what are they specific functions, and how they communicate with each other. Figure 35 - Advertisement Server Architecture illustrates the architecture design of the Visual Advertisement Server (Bigger picture in Annex A.7.1).

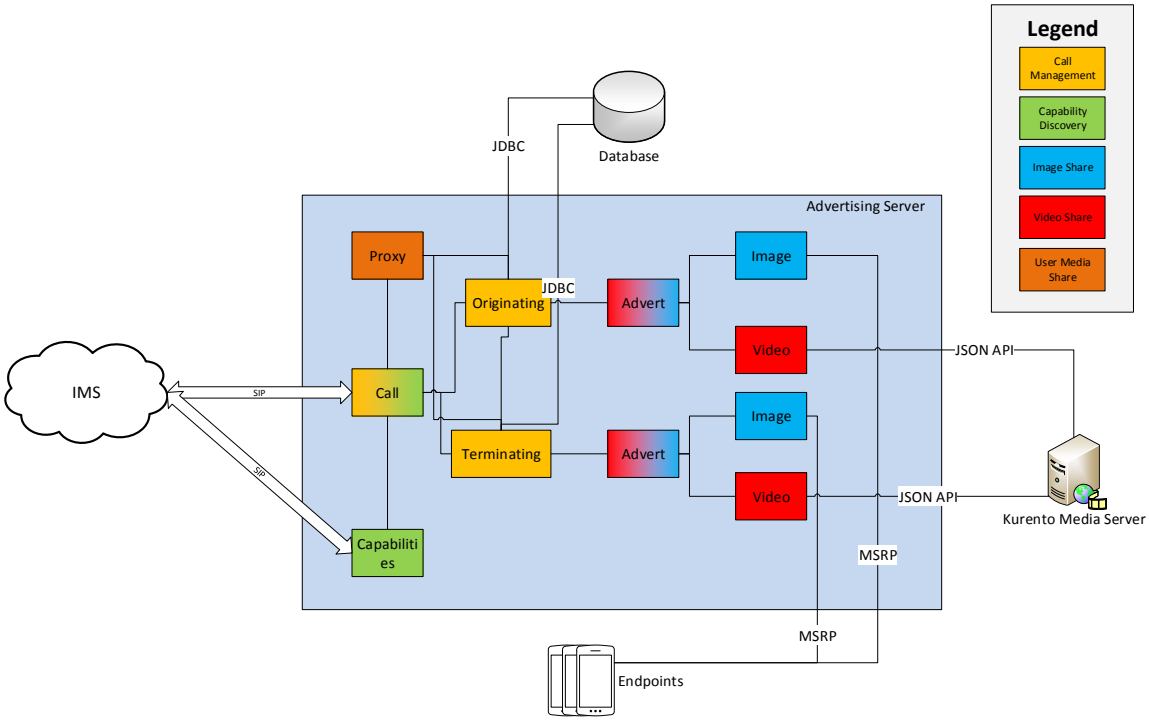

Figure 35 - Advertisement Server Architecture

The AS is composed by eight different modules, which in the context of **JAIN SLEE** are referred as **Service Building Blocks (SBBs)**. These SBBs will be further explained in the next sections.

5.8.3 Capability Discovery

As referred in chapter 3.2.1 Capability Discovery SIP OPTIONS is the method chosen for Endpoints capabilities discovery. Figure 36 describes the flow of messages necessary for the AS to be aware of endpoints capabilities at all time.

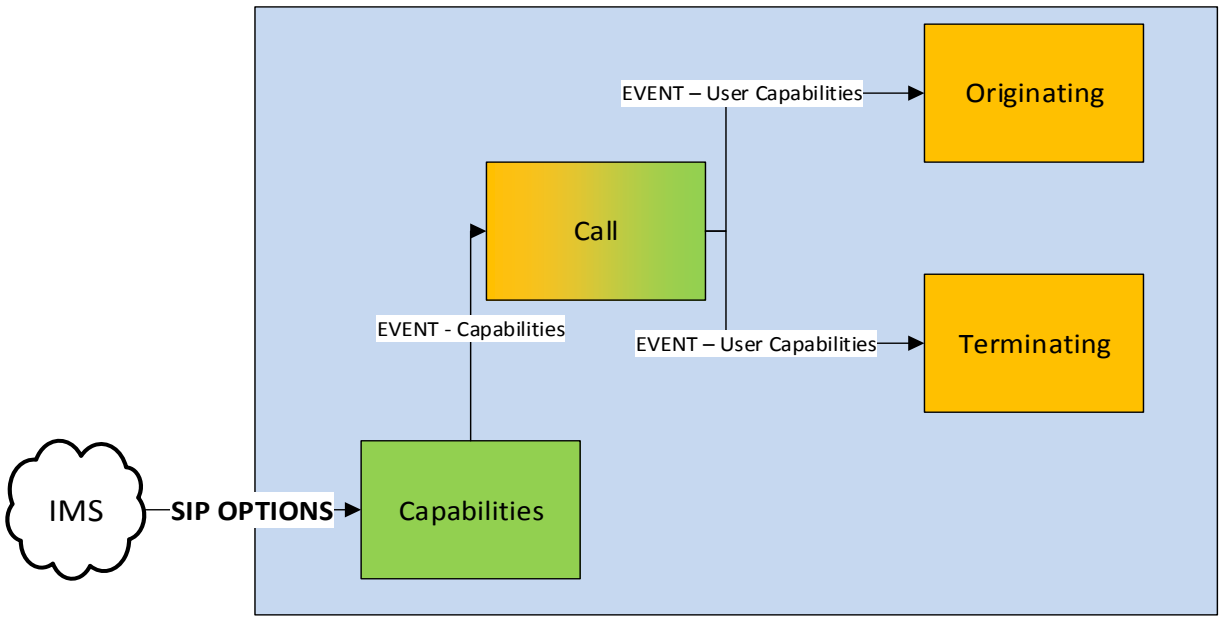

Figure 36 - AS Capabilities Discovery

The **Initial Filter Criteria** (IFC) configured in the IMS (5.3.1. Initial Filter Criteria guarantees that all SIP OPTION messages are redirected to the AS. The module Capabilities Sbb is constantly listening for new options messages.

Once a message arrives, the Capabilities module looks in the **ACCEPT-CONTACT**, and **CONTACT** headers for the image and video share RCS tags (3.2.1 Capability Discovery) of the originating endpoint. It also verifies the **FROM** and **TO** headers, in order to know which are the Endpoints involved in the messages exchange.

After the process above, the module adds its IP address to the **ROUTE** header and returns the OPTIONS message to the IMS network, so it can be forward to the terminating Endpoint (in this use case, the AS works as a SIP Proxy).

Because the IP Address of the VAS is in the **ROUTE** header of the SIP OPTIONS message, the response to this request will also pass in the VAS.

When the response to the initial request arrives at the Capabilities module, the initial process of looking for the image and video tags is repeated, in order for the AS discover the capabilities of the terminating endpoint. After this, the response is also returned to the IMS network.

By this point, the Capabilities module is aware of the image and video share capabilities of both endpoints, however, this is not the module responsible for handling the image and video share process. That is the responsibility of the Originating and Terminating modules. In order to inform these modules of the capabilities of their user's endpoint, the Capabilities module fires an event to the Call Sbb responsible for handling the call between both users (further explained in section 5.8.4. Call Management). The Capabilities Event fired is a **Java Object** that contains the users involved, and their respective capabilities (Figure 37 - Capabilities Event).

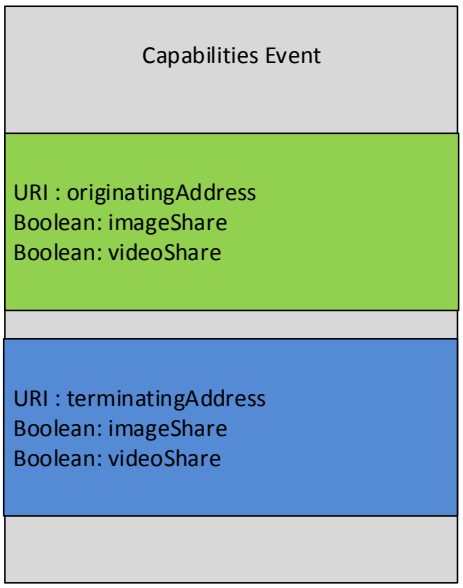

Figure 37 - Capabilities Event

After this event arrives at the respective Call Sbb, it is divided into two, one for each endpoint, and is forwarded to their respective module.

5.8.4. Call Management

The Call Management is the process responsible for handling all SIP Requests/Responses involved in a RCS VoIP Call. It also contains the necessary business logic to decided **when**, to **who**, and **what** kind of advertisement to share with endpoints. All the necessary modules and the message flow involved in the Call Management are presented in Figure 38 - Call Management.

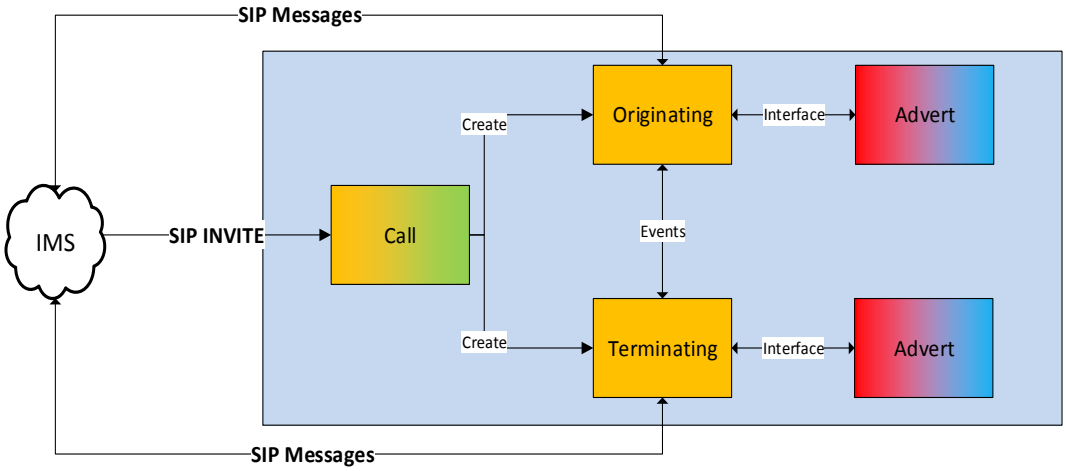

Figure 38 - Call Management

The HSS in the IMS Network contains an IFC that redirects every new SIP INVITE request to the AS.

When a new SIP INVITE request arrives to the AS, a new **Call** SBB is created to handle that new call. Then, the recent created SBB analyse the content of the Request to check if this Invite is a user media share inside an already established call, if that is the case the User Media Share process is established (see 5.8.8. User Media Share).

In case the request is an INVITE for a new RCS VoIP Call, the Call SBB creates two children of type **Originating** SBB and **Terminating** SBB, each responsible for communicate with the corresponding Endpoint. In this process, the VAS behaves as a **Back 2 Back User Agent** (Annex A - B2BUA).

Both **Children** SBBs are responsible for communicating with the **Database** in order to retrieve an **active** Advertisement Campaign (5.8.5. Database Communication), after that, and depending on the call use cases configured in the campaign (4.5.3 Use Cases), this SBBs must decide when it is possible to share an image or video advert.

For sharing media content, the **Originating** or **Terminating** SBB create a children of type **Advert** SBB. This SBB is responsible for handling the processes **of Image and Video share** (5.8.6. Image Share and 5.8.7. Video Share). It is also responsibility of these SBBs to stop and destroy the **Advert** SBB if the call ends during a Content share, or if the corresponding user no longer has the Capabilities to receive the image/video share.

The Originating SBB and Terminating SBB of one call must communicate between each other. This communication is done in a similar way of the Capability Discovery mechanism (5.8.3 Capability Discovery), using **Events**. For example, when the Originating Endpoint ends a call, a SIP BYE Request is sent to the Originating SBB, this Sbb must then **fire** an **Event** to the Terminating SBB, in order for this module to end the call with the Terminating Endpoint.

5.8.5. Database Communication

As mention in previous sections, the communication between the AS and the Database is made using **JDBC**.

In the AS architecture, the only two modules that require communicating with the Database are the **Originating** SBB and the **Terminating** SBB (Figure 39 - JDBC).

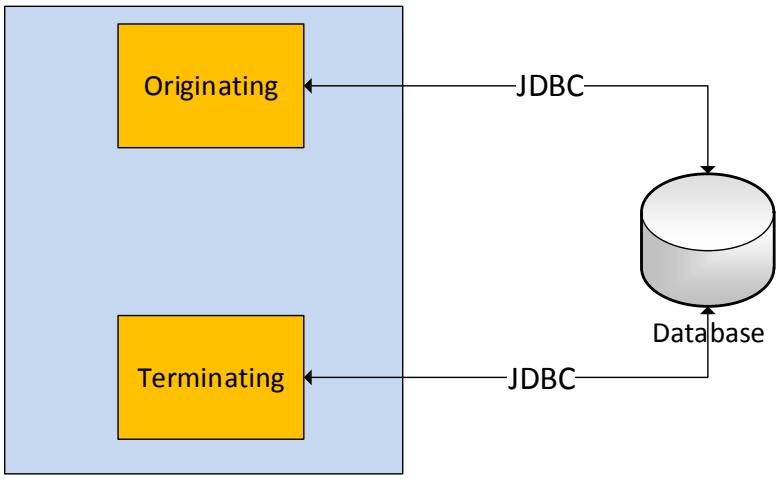

Figure 39 - JDBC

These two modules retrieve from the Database an **active Advertisement Campaign** that defines which image or video should be send to Endpoints. It's noteworthy that the campaign retrieved by the Terminating SBB must also have the "On Call" use case active; this is due to the fact that that is the only use case that the terminating Endpoint can receive advertisement.

After the campaign data is retrieved from the Database, it is responsibility of the Originating and Terminating SBB to decide when to share Media. The processes for sharing Image and Video with Endpoints are further explained in sections 5.8.6. Image Share and 5.8.7. Video Share.

If there are no active campaigns available, the RCS VoIP Call continues its "normal" flow, and no media is shared with Endpoints.

5.8.6. Image Share

In this section it is explained how the AS shares Images with Endpoints. Figure 40 - Image Share illustrates the modules involved in the Image Share process, and how they interact with each other.

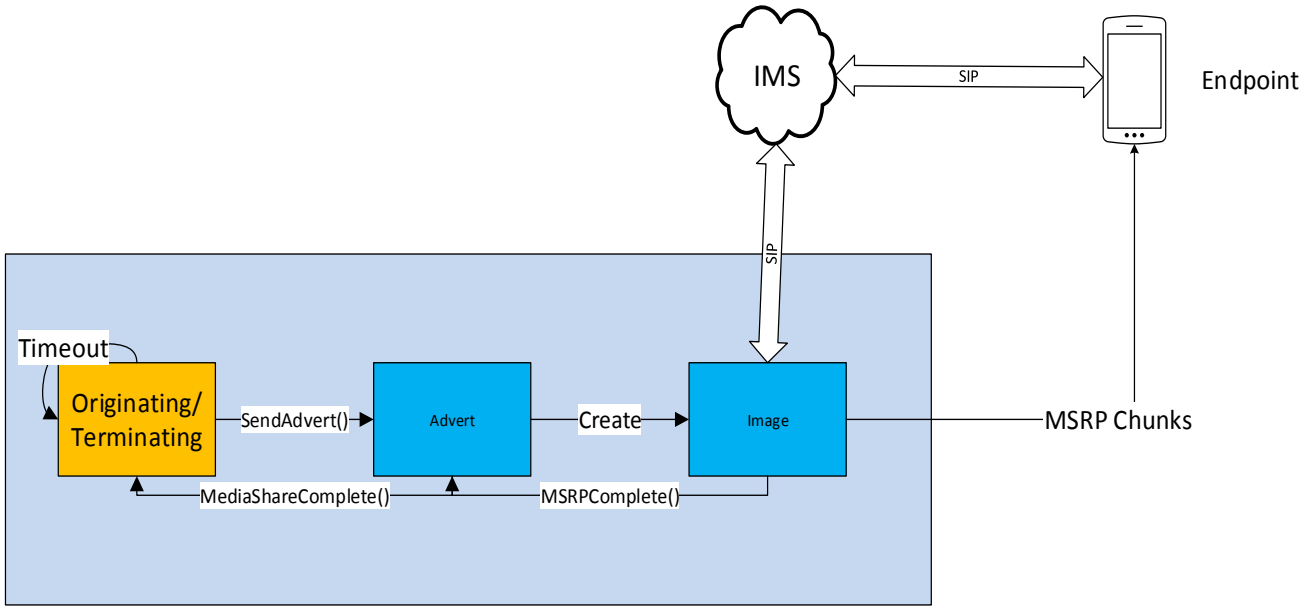

Figure 40 - Image Share

The Image Share process is initiated by either the Originating or Terminating SBB by calling the SendAvert method in the Advert Sbb interface (5.8.4. Call Management).

The **Advert** Sbb is then responsible for understanding if the media to be sent is an Image, and if so, to **create** a new child SBB of type **Image** SBB. The first responsibility of this child SBB is to handle the SIP communication with Endpoint (detailed in 4.5.3 Use Cases), as well as negotiates the SDP to be used (Figure 14 - SDP offer-answer model).

After the AS and the Endpoint have agreed on the SDP exchange, and the "**200 OK**" SIP Response have arrived to the initial Image Share SIP Invite, the application server has now all the necessary information to successfully share an Image file.

The other main function of the Image SBB is to **chunk** the Image file into **several** small parts, and to send these pieces of images as **MSRP** messages, as explained in 3.4.7. Message Session Relay Protocol (MSRP).

When the Image Share is **completed** (when a SIP BYE request arrives at the Image SBB), or if the Endpoint user does not accept the image share, a **notification** of this event is sent to the Parent SBB (**Advert** SBB) using its' interface. The Advert SBB then forwards that notification to its parent (Originating/Terminating SBB). When the Originating or Terminating SBB receives a notification that the image is completed or cancelled, it **starts** a **timer event** (Timeout defined in Campaign properties) in order to restart the entire process.

It is noteworthy that at each cycle, the **current capabilities of the endpoints are checked**, and the process only continues if the image share capability is available. Also, if the

campaign has more than one image configured, a random choice of image is made at each cycle.

Message Session Relay Protocol

During the development of the Image SBB, it became clear to the author that it was not practicable to develop a MSRP library from scratch, because it would be too time consuming. After an analysis of which libraries were available, it was verified that at the time of writing, there is only one java open-source library available [42]. Luckily, the person responsible for the development of this library also has some experience working with Mobicents, and a beta version of a **Resource Adaptor** to be integrated in the Mobicents JAIN SLEE platform was already available [43].

In the context of MSRP, **chunking** is the process of breaking a file into small pieces that can later be aggregated by the endpoint to obtain the original file. As the reader can easily understand, this process ensures that if one message is lost during the transfer, only that specific message needs to be retransmitted.

However, and due to the fact that the library is still in beta version, the Chunking option was still not available. Due to the fact that this feature was essential for the Image Share process, the **author developed this feature,** and at the time of writing, it is planned to be merged with the **MSRP .net library** [42] in the next version.

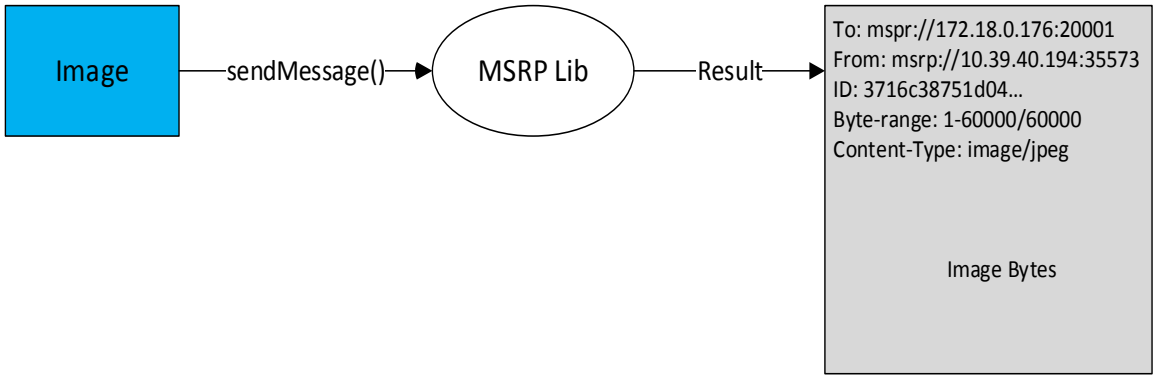

Figure 41 - MSRP Original Library

Figure 41 illustrates an example of how the original library converts an image into a MSRP Message. On the other hand, Figure 42 illustrates the process of chunking implemented by the author.

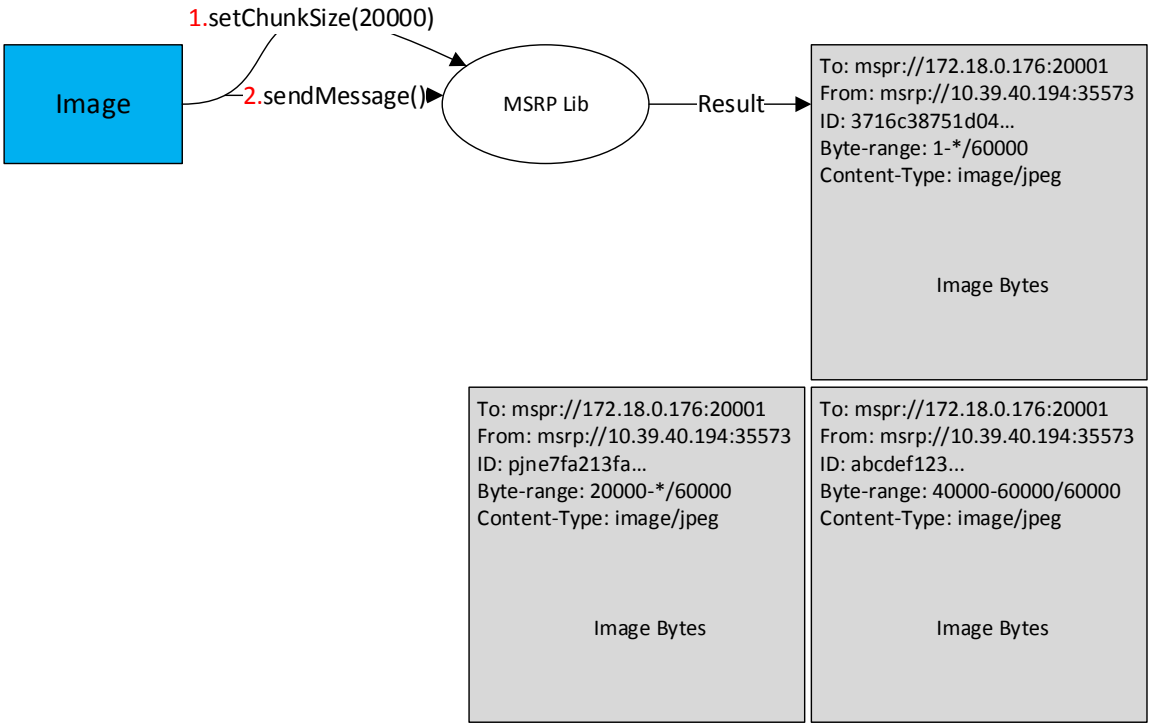

Figure 42 - MSRP Changed Library

In the changed library, the Image SBB starts by defining a chunk size, in other words, the maximum size of one MSRP Message. After this, when the SBB calls the method sendMessage, the MSRP library divides the image into several chunks, and sends each message as an individual message.

5.8.7. Video Share

Contrary to what is done during an Image Share, the content of a **video advert** is **not** sent directly from the AS to the **Endpoint**. That is achieved with the help of the Kurento Media Server (2.5. Media Server / Media Gateway). Figure 43 illustrates the VAS modules involved in the Video Share Process.

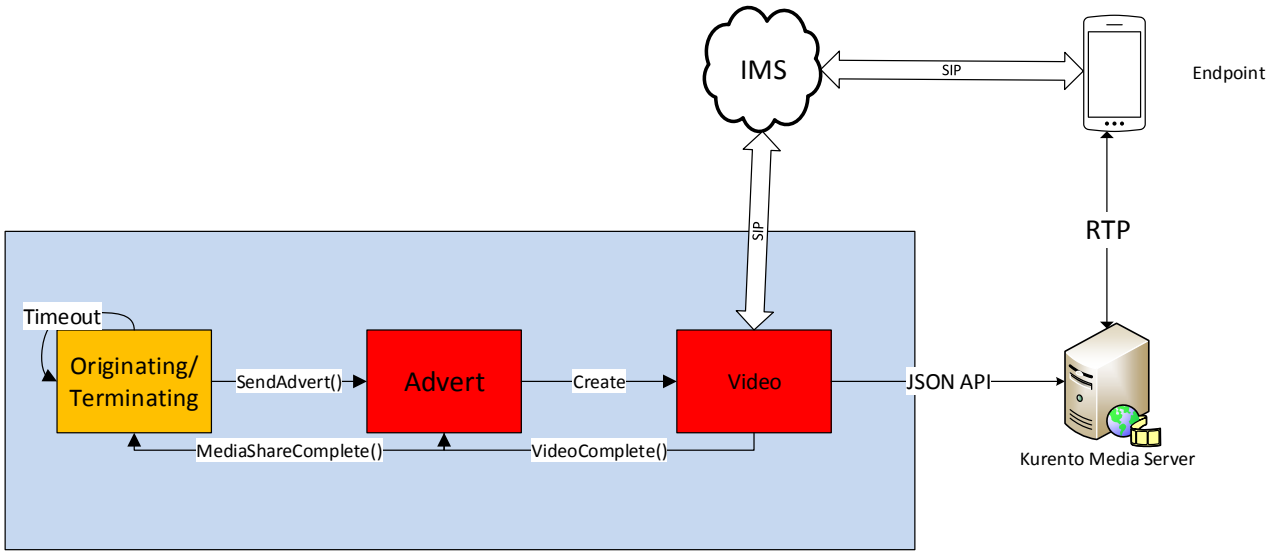

Figure 43 - Video Share

The process of Video Share is the **same** used in Image Share (5.8.6. Image Share), until the Sbb responsible for sending the media content to the Endpoint (Video Sbb) is created. Because the Video Share process must use transcoding and streaming, it is necessary to use a Media Server. During this project, it was used the **Kurento Media Server** [18].

Similar to what is done in the Image Share process, the first responsibility of the Video Sbb is to negotiate the Session with the Endpoint, but unlike in the image share process, the SDP used in the negotiation is the Media Server SDP (Figure 44).

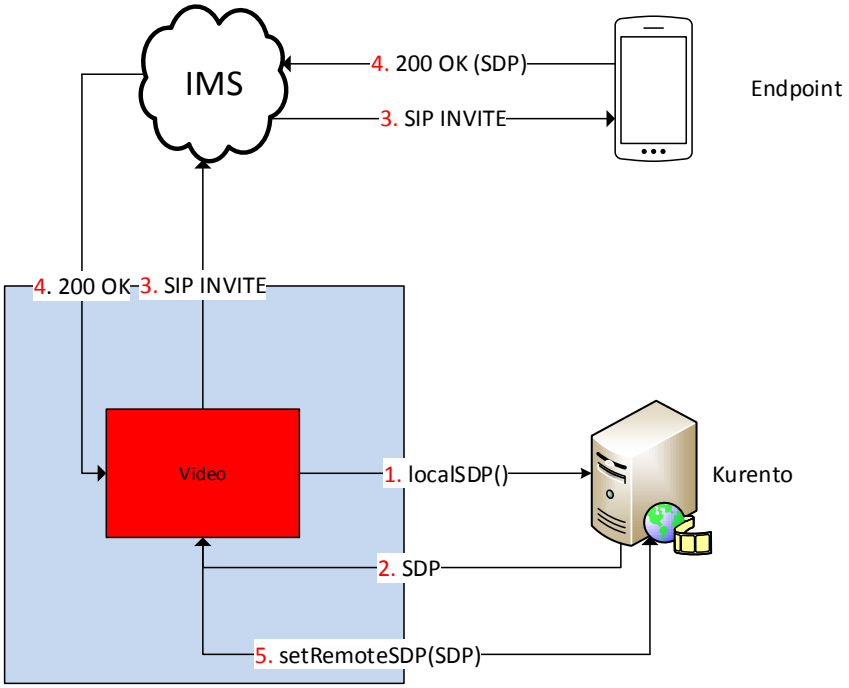

Figure 44 - SDP Negotiation

The SDP negotiation mechanism is initiated when the Video SBB is created, using the Kurento Media API [41] method **getLocalSDP()** (1.). This method sends a JSON request to the media server, and in the response (2.) the SDP of the media server is returned (example of SDP is given in 5.7. Media Server).

The Video SBB must then send this SDP to the Endpoint (3.). This process is the same that is used in the Image Share mechanism. If the Endpoint accepts the video share, its' SDP is returned to the Video SBB in the body of the SIP 200 OK Response (4.). All the SIP messages exchanged in between can be consulted in the use cases chapter, 4.5.3 Use Cases The negotiation process ends by setting the remote SDP in the Kurento Media Server. This is achieved by, once again, using a method offered in the KMS API (5.)

At this point, the Kurento Media Server and the Endpoint are prepared to send, and receive video share, respectively. The API used to control the Kurento Media server offers a simple mechanism for applications to interact with it. In order to start playing the Advert, the Video Sbb only needs to link a video file with a **PlayerEndpoint** (Kurento Java Class), and then call the API method **play()**. If at any time the endpoints cancels the video share, the API method **stop()** is called in order to inform Kurento that it can stop the video streaming.

Similarly to what happens in Image, the video share is **complete** when a SIP BYE request is received by the Video SBB. The forwarding procedure of notifying the parent SBBs that the media share is complete is the same that is used in the Image Share process.

5.8.8. User Media Share

One of the use cases detailed in 4.5.3 Use Cases is that at any point, the user must be able to exchange media between them, in other words, the Advertisement Service must never interfere with the capabilities of Endpoints. To ensure this, the AS contains a **Proxy SBB** that is constantly listening for users Image/Video Share (Figure 35 - Advertisement Server Architecture).

As referred in section 5.8.4. Call Management the User Media Share process is initiated when the Call SBB receives a SIP Invite request inside a call that is already established, and that contains the tags for image or video share in the **ACCEPT-CONTACT** header (Figure 45 - User Media Share).

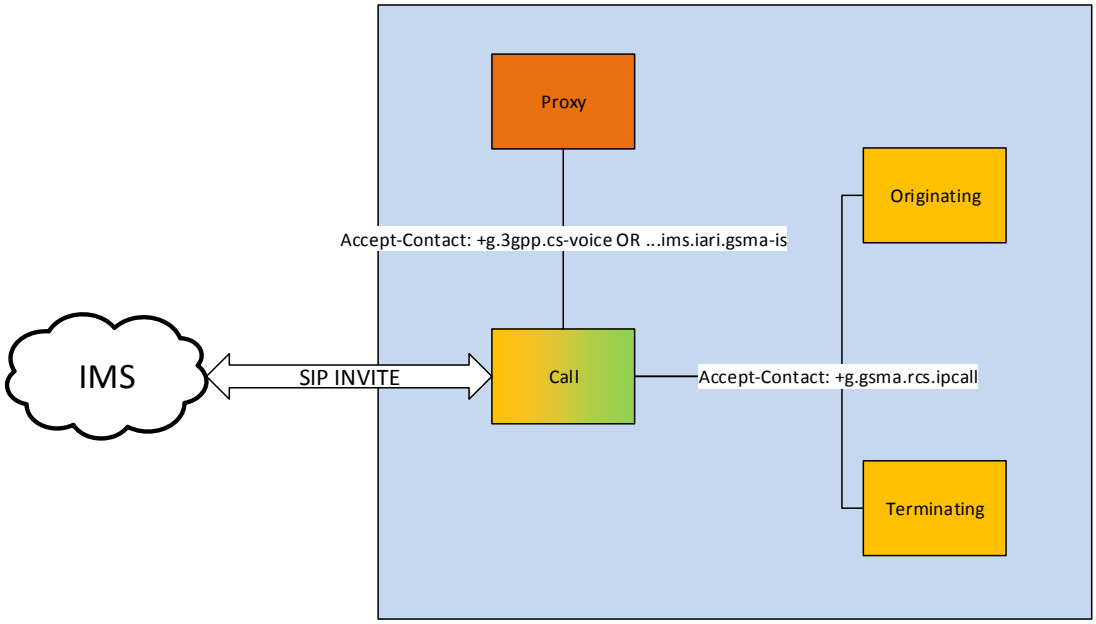

Figure 45 - User Media Share

If the AS detects an Invite for media share for an already established call, in other words, if an invite is forward to the **Proxy SBB**, this SBB checks the **TO** SIP header in the INVITE received in order to understand **which Endpoint** (Originating or Terminating) will **receive** the media share. After this, the Proxy SBB sends an **Event** to the SBB responsible for handling that Endpoint, notifying it **to stop any current Advert** (Figure 46 - Cancel Advertisement).

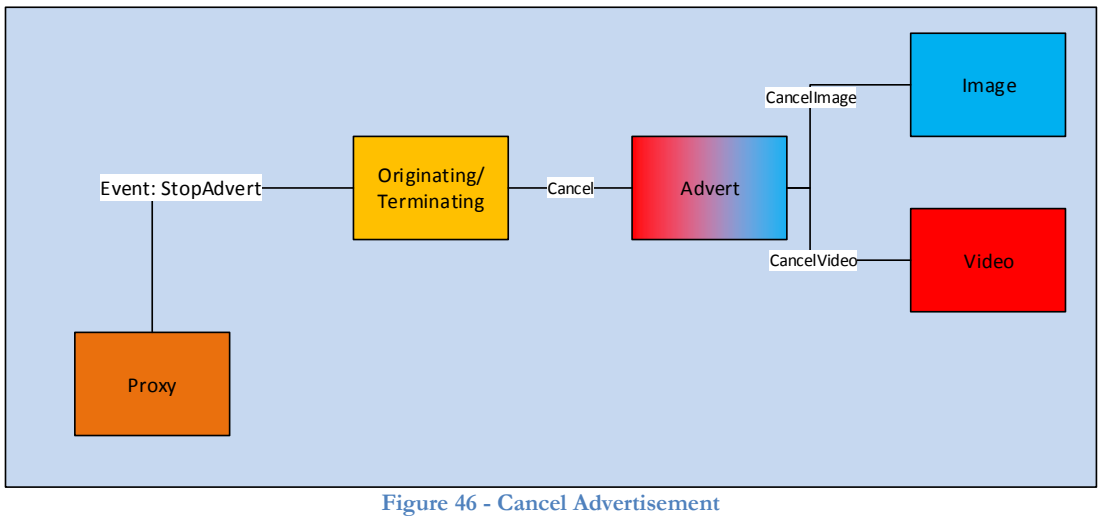

Figure 46 - Cancel Advertisement

After the **StopAdvert** is sent, the Proxy Sbb, like the name suggests, assumes the function of a SIP Proxy Application Server, just **forwarding** all the SIP Messages. It is noteworthy that during the period the User Media Share lasts, **no advert** is sent to the **Endpoint** that is **receiving** that media share, but the other peer can still receive Advertisements (more information in 4.5.3 Use Cases).

When the User media share is completed, another event is sent from the Proxy SBB to the Originating/Terminating SBB notifying it that it can send adverts again (same message flow as illustrated in Figure 46 - Cancel Advertisement).

Chapter 6
Requirements Validation

Besides developing the software, it was necessary to evaluate all the features implemented. In this chapter, all the tests carried out during the internship are presented, and a brief analysis is made by the author in order for the reader to understand if all the requirements (4.2 Requirements Analysis) were achieved.

6.1. Introduction

Software Testing can be defined as the **evaluation** of a software/product that is being developed to check its **capabilities** and **ability** to **deliver** the intended **results**. It is an integral part of the software development life cycle.

The main objectives of Software Testing are to validate and verify that a product:

- Meets the requirements
- Works as expected
- Satisfies the needs of the client.

6.2. Methodology

There are several techniques and methodologies that are in practice nowadays, and different sets of test cases may require different test strategies. However, most of the methodologies used during software testing have the following levels of testing:

- **Unit** – Testing of individual software components or modules. It is typically done by the programmer and not by testers. This level of testing requires detailed knowledge of the internal program design and code. During the internship, this level of testing was made the author during the Implementation phase using **JUnit** [44] whenever necessary.
- **Integration** – Testing of integrated modules to verify combined functionality after integration. This level of testing is especially relevant to client/server and distributed systems. During the internship this level of testing was usually done at the end of each Sprint, in other words, whenever a new feature was added to the system, a battery of tests was made in other to validate its functionality.
- **System** – The entire system is tested as per the requirements specifications. This level covers all combined parts of a system/product. These types of tests were made by the author in during the last phase of development with an "alpha" version of the product.
- **Acceptance** – Usually this type of testing is done by the client, in order to verify that the system meets the customer specified requirements. Due to the fact that the product developed during the Internship was a POC, and there was not a real costumer involved, the Acceptance level of testing was made by the author prior the development phase.

Due to the fact that the development approach taken by the author was similar to Scrum (4.3 Methodology), most of the tests were made during the **Sprints**. Only the System and

Acceptance Testing were made prior to the development. This approach is similar one Testing Methodology called the **V-Model Testing Methodology**.

Although the V-Model Methodology was first design to be an extension to the **Waterfall** model, it integrates well with the approach taken during the internship. Figure 47 - V-Model illustrates all the phases involved in a V-Model Testing project.

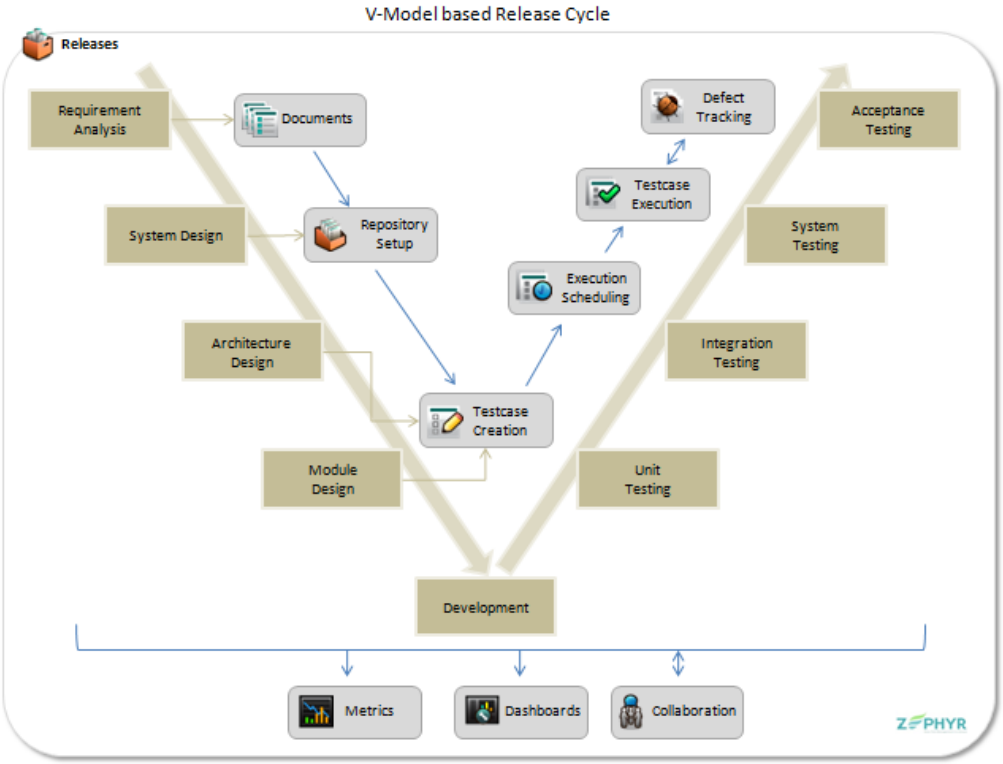

Figure 47 - V-Model

6.3. Functional Tests

In order to become easier to **validate** or **disprove** a requirement, a validation model was design (Table 18- Test Model) for the functional tests. This model was used during all levels of testing, except for the unity tests. To use this model in all unity tests would be very time consuming, and with few improvement to the overall project.

Requirement: Title of the Requirement			
Level: Integration/System/Acceptance			
Description: A small description of the test.			
Action	**Expected**	**Observed**	**Pass/Fail**
Actions that lead to the test	What is expected from the system	What really happened	✔/✖
Problems: A description of what was the problem.			

Table 18- Test Model

Due to the fact that the number of tests is very big, and it would make this document too extensive, only a summary table is presented for each of the Modules that were tested. All the tests can be consulted in Annex C.

6.3.1. BackOffice and Database

As referred, the Back Office developed during the internship is just for testing and not the final product. Due to that fact, the important tests to be done were the **integration** tests between the **Demo BO** and the **Database**. Table 19 summarizes the results obtained.

Test Name	Level	Pass/Fail
User		
Authentication	System	✔
Create Campaign	System	✔
Edit Campaign	System	✔
Delete Campaign	System	✔
Edit User	System	✔
ADMIN		
Add User	System	✔
Search Users	System	✔
Search Campaigns	System	✔
Search Videos	System	✖
Search Images	System	✖
Edit Users	System	✔
Edit Campaigns	System	✔
Statistics	System	✖

Table 19 – BackOffice and Database Test Results

The features that failed to pass the tests were not part of the main use cases necessary to implement in the **BackOffice**, and were considered extras. Due to the lack of time available for the development of the Demo BO, these features were not implemented.

6.3.2. Kurento Media Server

During the integration of the media server with the entire system, it was necessary to test that this Media Server could handle all the necessary codecs (4.2 Requirements Analysis) and transcoding of several video formats. Table 20 summarizes the results obtained.

Test Name	Level	Result
Video Codecs		
H.263	Integration	✖
H.264	Integration	✔
Audio Codecs		
G.711	Integration	✔
GSM	Integration	✔

Table 20 - Kurento Media Server Tests

In the Kurento Media Server documentation [18] the video codec H.263 is listed as one of the supported codecs. However, after some tests, the author realized that this media server did not transcode video files into H.263. After contacting the developers of Kurento, the author was informed that the implementation of this codec has some bugs, and was withdrawn. Nevertheless, it is in the roadmap of Kurento, and should be available in the next version.

6.3.3. Application Server

Like already referred in Chapter 5 – Development, the Application Server is the core element of the entire system. Due to this fact, it was the module that was target of more tests. Also, this modulo communicates with every other module (except the BackOffice), so, the integration tests are of extreme importance. Table 21 summarizes the results obtained.

Capabilities Discovery

Test Name	Level	Pass/Fail
Options IFC	Integration	✔
Discover Capabilities From Options	Integration	✔
Discover Capabilities From Response	Integration	✔
Fire Capabilities Event	Integration	✔
Capabilities Event Received	Integration	✔

Table 21 - Capabilities Discovery Tests

Call Proxy

Test Name	Level	Pass/Fail
Forward Invite	Integration	✔
Forward Responses & communication between SBBs	Integration	✔
Call Declined	Integration	✔
Call Terminated	Integration	✔

Campaigns

Test Name	Level	Pass/Fail
Database Communication	Integration	✔
Use Case Decision	Integration	✔

Image Share

Test Name	Level	Pass/Fail
Invite	Integration	✔
Rejected	Integration	✔
Accepted	Integration	✔
Cancelled	Integration	✔

Video Share

Test Name	Level	Pass/Fail
Get SDP from Media Server	Integration	✔
Invite	Integration	✔
Rejected	Integration	✔
Accepted	Integration	✔
Cancelled	Integration	✔

Use Case – Before Call

Test Name	Level	Pass/Fail
AS Answer Call	System	✔
Media Share	System	✔
Call Forward	System	✔

Use Case – Ringing

Test Name	Level	Pass/Fail
AS Answer Call & Forward Invite	System	✔
Media Share	System	✔
Call Answered by Terminating User	System	✔
Call rejected by Terminating User	System	✔

Use Case – During Call Users Media Share

Test Name	Level	Pass/Fail
Not Sharing Advert	System	✔
Sharing Advert	System	✔

6.3.4. Acceptance Tests

Acceptance tests are usually done by the costumer. However, due to the fact that the main goal of the internship is to implement a POC, these tests were made by the author, and it was decided between the author and the company supervisor, that use cases defined during the planning phase (4.5.3 Use Cases) should be the main acceptance tests. Due to the importance of these tests in the overall project, they are the only ones that are fully presented in this document, and not in Annex.

BackOffice – Create Advertisement Campaign

Requirement: Create Campaign			
Level: Acceptance			
Description: Back Office User creates a new Campaign			
Action	**Expected**	**Observed**	**Pass/Fail**
User access BO and creates a new Campaign	Campaign is added to the Database	Same as Expected	✔
Problems:			

Application Server - Capability Discovery Mechanism

Requirement: Capability Discovery Mechanism			
Level: Acceptance			
Description: AS must be aware of users Capabilities.			
Action	**Expected**	**Observed**	**Pass/Fail**
AS Receive SIP Options	AS updates capabilities whenever necessary.	Same as Expected	✔
Problems:			

Advertisement Server – Database Campaign Retrieve

Requirement: Campaign Retrieve			
Level: Acceptance			
Description: AS connects with Database and Retrieves an active Campaign.			
Action	**Expected**	**Observed**	**Pass/Fail**
New Invite Arrives at AS	Active Campaign Retrieved from Database	Same as Expected	✔
Problems:			

Advertisement Server – Media Share before Call setup

Requirement: Use Case – Media Share before Call setup			
Level: Acceptance			
Description: AS must be able to send Visual Advertisement to the Originating User before the call setup is established			
Action	**Expected**	**Observed**	**Pass/Fail**
Campaign retrieved from Database is active for "Before Call"	AS Invites Originating User to media share before forwarding the call Invite to Terminating User	Same as Expected	✔
Problems:			

Advertisement Server – Media Share during Call setup

Requirement: Use Case – Media Share during Call setup			
Level: Acceptance			
Description: AS must be able to send Visual Advertisement to the Originating User during the call setup is established			
Action	**Expected**	**Observed**	**Pass/Fail**
Campaign retrieved from Database is active for "Ringing"	AS Invites Originating User to media share after forwarding the call Invite to Terminating User. If Terminating user Accepts/Declines call, media Share is cancelled	Same as Expected	✔
Problems:			

Advertisement Server – Media Share during ongoing Call

Requirement: Use Case – Media Share during ongoing Call			
Level: Acceptance			
Description: AS must be able to send Visual Advertisement to both users during ongoing Call.			
Action	**Expected**	**Observed**	**Pass/Fail**
Call is established. Campaign is active for "Call"	AS Invites Originating/Terminating User to media share during call to Terminating User.	Same as Expected	✔
Problems:			

Advertisement Server – Not interfere with users media Share Capabilities

Requirement: Use Case – Allow Users to share media between them			
Level: Acceptance			
Description: Users in an ongoing call must be able to share media between them at any time.			
Action	**Expected**	**Observed**	**Pass/Fail**
AS receives media share Invite from any User	AS cancels any ongoing Advert share with the User that will receive the invite.	Same as Expected	✔
Problems:			

Table 22 - Acceptance Tests

Once the system developed **passed all the acceptance tests**, the author considers that the functional tests requirement is **validated**. However, not all the requirements were accomplished, which leaves space for **future work** to optimize the system.

6.4. Non-Functional Tests

As defined in 4.2 Requirements Analysis, besides the Functional Tests, there were some Non-Functional Tests that should be achieved during the development of the Advertisement System. In the next sections the author details the approach taken, and the tests that were made in order to validate these requirements.

6.4.1. Standard Protocols - IMS11, RCS 5.1

In order to guarantee these requirements, the author followed the specifications of both IMS11 [14], and RCS5.1 [5].

The first one defines how the modules involved in an IMS architecture should communicate with each other, in other words, which interfaces and protocols should be used; in the scope of this internship, the important interfaces are, **ISC** for communication between the SCSCF and the Application Server, and **UT** for communication between the AS and Endpoints (3.3.2 IMS Core). In both this interfaces, **SIP** is the protocol used. This specification also defines that, in order for SIP Messages to be re-directed to the AS, one or more **IFCs** should be configured in the HSS. Following IMS11 specs assures that the system developed can easily be integrated, without any changes, in any IMS network already in use.

Following the Specifications of RCS5.1, the author can guarantee that the product developed will be compatible with any RCS client that also follows these specifications, independently of which company developed that client. RCS5.1 Specs define how Endpoints should communicate with the network, in the scope of the internship, the important part of these specs are the **Capabilities Discovery Mechanism** (3.2.1 Capability Discovery) and the **Content Sharing** (3.2.2 Content Sharing). It is defined MSRP (with chunking) as the transport protocol for Image Share, and the usage of SIP Options exchange for the process of capabilities discovery.

6.4.2. High Throughput

Due to the lack of time, this requirement was not validated. During the planning phase of the internship, it was allocated 2 weeks for the final tests. During that period, it was expected for the author to use the **SIPp** tool [45].

SIPp is an Open Source traffic generator for the SIP protocol. It allows the developer to write multiple XML scenarios. In these scenarios, the developer defines the message flow that should happen, in other words, a SIPp scenario defines the behaviour of a user.

A scenario can then be used as a passed/failed test (unity testing), or the tester can define a number of parameters for each test, such as: number of users, maximum connections per second, retransmissions; that provide a report containing all the statistical information (number of messages exchanged, unexpected messages received, timeouts, errors, average RTT time, and others).

This tool allows the construction of scenarios involving RTP (video share), however, it drastically increases the complexity of the scenarios, and the author was not able to achieve

it during the time available. Furthermore, SIPp does not have any module to test MSRP transfers. That would derail the benchmarking of Image Share.

Due to these reasons, it was not possible to validate the high throughput requirement of the project.

6.4.3. Open-Source Software

All the software used during the internship is **Open-Source** or **Licensed-Free**. However, it is noteworthy that if the POC evolves into a real product, another analysis to the state of the art (Chapter 2 – State of the Art) is needed. This is due to the fact that during the internship only free solutions were analysed, and there might be some products on the market that offers better performance. The main modules that should be analysed in this case are the Media Server and MSRP stack (both solutions chosen during this project are still in beta version and in development).

Chapter 7
Conclusions

The main purpose of this project was to create a Value Added Service (VAS) to an IMS network, capable of delivering image and video advertisement to RCS users. During the first semester of the internship, the main objective was to research how could, and how should the purposed project be accomplished. First it was defined which features should be present in the service, then, it was necessary to analyse, study and test all available technologies. Finally, after defining all the technologies that were going to be used, it was researched and tested how these different components could be brought together in order to build a strong and innovative solution.

In order to achieve the largest number of features as possible, it was decided to use some new technologies, like for example the MSRP Resource Adapter, and the Kurento media server, which were still in development stage. This ambition brought great value to the product, but it also increased the risk of failure. Nevertheless, considering that the aim of this internship was to learn and develop new capabilities, it was thought it was a risk worth taking.

During the **First semester**, the entire original planned was not fulfilled due to a number of facts. First of all, the internship was scheduled to start in the beginning of September, but because of some bureaucracies, it only started in the 21st of October, almost one and a half month later. However, with some extra work done by the author, most of the delay was recovered. By the end of the semester some problems occur while integrating the RCS client (joyn) with the OpenIMS network and the Application Server. Nevertheless, this process was important to fully understand the architecture and functionality of the IMS network, and the differences between a RCS client and softphones.

After the architecture design was made, and the technologies to be used were decided, it was time for Development. This was the main focus of the **Second Semester**. As referred in previous chapters, the author and the company supervisor realized that the time available for development could not be enough to implement all the desired features. So, it was settled that the BackOffice (BO) would be the last feature to be implemented. This is due to the fact that the innovative part of the Internship was in the Application Service and not in the BO. By the end of the semester, the author took some time to perform the necessary **Tests** on the System. These Tests are of extreme importance in order to **validate** all the implemented **Requirements**, they also function as a guideline for the company to perceive if the internship was successful or not.

The Development flowed as planning, however, there were some challenges found. During the integration of the **MSRP Resource Adaptor**, the author realized that one of the key necessary features (**Chunking**) was not developed in the MSRP Stack, so, some time was spent developing this feature. However, the fact that this feature developed by the author was accepted by the developers of the MSRP Stack, and at the time of writing, is planned to be included in the next version brings great value to the work done by the author, and the extra time spent doing it.

Overall, the work developed during this internship was the biggest challenge the author has ever faced throughout his academic life. On a personal level, it contributed greatly to the growth as a person and as a software developer. It enabled the author to experience how the real market works and offered the author a "bridge" between the academic life and a

professional career as a Software Engineer. From the technically point of view, the biggest advantages was the possibility to learn and work with several technologies and products that were unknown to the author, as for example SIP, MSRP, RCS and IMS.

Considering the Advertisement Service developed during the Internship, the author considers that he was able to achieve all the main goals and a functional POC was produced.

Future Work

As referred in previous chapters, not all the proposed requirements were achieved. Furthermore, the author did not have the time to perform a complete benchmark on the Application Server. Taking these factors into account, the author considers that the future work to be done is:

- Develop the final **BackOffice** (Annex C).
- Develop the **Statistics Mechanism** and integrate into the System.
- Perform Benchmark on the Application server in order to validate the **high throughput** requirement.
- Write the patent of the POC (first draft currently being written by the author).

References

[1] "History of telecommunication," [Online]. Available: http://en.wikipedia.org/wiki/History_of_telecommunication. [Accessed 28 09 2013].

[2] "GSM Association," [Online]. Available: http://www.gsma.com/. [Accessed 02 10 2013].

[3] "3rd Generation Partnership Project," [Online]. Available: http://www.3gpp.org. [Accessed 1 10 2014].

[4] "Over-the-Top Application," [Online]. Available: http://www.techopedia.com/definition/29145/over-the-top-application-ott. [Accessed 16 12 2013].

[5] "RCS 5.1 Specifications," [Online]. Available: http://www.gsma.com/futurecommunications/faq/specifications/. [Accessed 18 09 2013].

[6] "Telecommunication," [Online]. Available: http://en.wikipedia.org/wiki/Telecommunication. [Accessed 26 09 2013].

[7] "Comparative Study of 1G, 2G, 3G and 4G," [Online]. Available: http://borjournals.com/Research_papers/Ap_2013/1248IT.pdf. [Accessed 22 11 2013].

[8] "Online Messaging costs," [Online]. Available: http://thetechdeal.blogspot.pt/2012/02/online-messaging-cost-carriers-139.html. [Accessed 03 12 2014].

[9] "The Pudding Targeted Advertising," [Online]. Available: http://techcrunch.com/2007/09/24/thepudding-targeted-advertising-comes-to-phone-calls/. [Accessed 24 12 2013].

[10] "Pudding Image," [Online]. Available: http://www.nytimes.com/2007/09/24/business/media/24adcol.html?ex=1348286400&en=2b872e9e7df0ee8f&ei=5090&partner=rssuserland&emc=rss&_r=0. [Accessed 13 12 2013].

[11] "Jangl and jajah," [Online]. Available: http://mashable.com/2008/01/14/voip-advertising/. [Accessed 14 12 2013].

[12] "Skype Advertising," [Online]. Available: http://blogs.skype.com/2012/06/13/skype-advertising-update/. [Accessed 28 12 2013].

[13] "Google Patents Targeted Advertising in Phone Calls," [Online]. Available: http://www.tomsguide.com/us/google-voip-advertising-phone-calls,news-14457.html.

[Accessed 18 12 2013].

[14] "3GPP IMS Specification Re-11," [Online]. Available: http://www.3gpp.org/DynaReport/23228.htm. [Accessed 01 10 2013].

[15] "Broadband Integrated Services Digital Network," [Online]. Available: http://en.wikipedia.org/wiki/Broadband_Integrated_Services_Digital_Network. [Accessed 28 11 2013].

[16] "Asynchronous Transfer Mode," [Online]. Available: http://en.wikipedia.org/wiki/Asynchronous_Transfer_Mode. [Accessed 26 09 2013].

[17] "Skype, P2PSIP and P2P Networks," [Online]. Available: http://www.disruptivetelephony.com/2010/11/a-brief-primer-on-the-tech-behind-skype-p2psip-and-p2p-networks.html. [Accessed 28 12 2013].

[18] "Kurento Media Server," [Online]. Available: http://www.kurento.com/documentation. [Accessed 10 04 2014].

[19] "VoIP Terms and Definitions," [Online]. Available: http://www.patton.com/manuals/VoIP_Glossary.pdf. [Accessed 04 10 2013].

[20] "Packet-swithced vs. Circuit-Switched Networks," [Online]. Available: http://www.computerworld.com/s/article/41904/Packet_Switched_vs._Circuit_Switched_Networks. [Accessed 26 09 2013].

[21] "VoIP statistics," [Online]. Available: http://blog.businessphonesdirect.com/2012/07/18/3-key-reasons-why-voip-is-a-good-choice-for-your-business/. [Accessed 04 10 2013].

[22] U. O. C. M. I. F. A. R. M. S. Rogier Noldus, IMS Application Developer's Handbook, Academic Press, 2011.

[23] "IMS," [Online]. Available: http://en.wikipedia.org/wiki/IP_Multimedia_Subsystem. [Accessed 02 10 2013].

[24] "JAIN SLEE vs SIP Servlets," [Online]. Available: https://developer.opencloud.com/devportal/download/attachments/29818915/atnac2007_OC_JAIN_SLEE_vs_SIP_Servlet_f.pdf. [Accessed 23 12 2013].

[25] "RFC 2543," [Online]. Available: http://www.ietf.org/rfc/rfc2543.txt. [Accessed 28 09 2013].

[26] "RFC SIP," [Online]. Available: http://tools.ietf.org/html/rfc3261. [Accessed 29 09 2013].

[27] "RFC RTP," [Online]. Available: http://www.ietf.org/rfc/rfc3550.txt. [Accessed 05 11 2013].

[28] "RFC SDP," [Online]. Available: http://tools.ietf.org/html/rfc4566. [Accessed 03 10 2013].

[29] "SDP Image," [Online]. Available: http://www.in2eps.com/fo-abnf/tk-fo-abnf-

sdp.html. [Accessed 06 11 2013].

[30] "RFC MSRP1," [Online]. Available: http://tools.ietf.org/html/rfc4975. [Accessed 04 11 2013].

[31] "RFC MSRP 4976," [Online]. Available: http://tools.ietf.org/html/rfc4976. [Accessed 04 11 2013].

[32] "RFC3435," [Online]. Available: http://www.ietf.org/rfc/rfc3435.txt. [Accessed 30 09 2013].

[33] "RFC MEGACO," [Online]. Available: http://tools.ietf.org/html/rfc3525. [Accessed 03 10 2013].

[34] "Image MEGACO," [Online]. Available: http://en.wikipedia.org/wiki/H.248. [Accessed 20 12 2013].

[35] "Scrum Methodology," [Online]. Available: http://agilemethodology.org/. [Accessed 24 12 2013].

[36] "Waterfall," [Online]. Available: http://learnaccessvba.com/application_development/waterfall_method.htm. [Accessed 19 02 2014].

[37] "Kurento Installation," [Online]. Available: https://github.com/Kurento/kurento-media-server/wiki/Kurento-Media-Server-Binary-Install-Guide#Prerequisites. [Accessed 21 2 2014].

[38] "PostgreSQL," [Online]. Available: http://www.postgresql.org/. [Accessed 13 01 2014].

[39] "Postgresql Install," [Online]. Available: http://www.postgresql.org/download/linux/ubuntu/. [Accessed 13 01 2014].

[40] "Crow's Foot Notation," [Online]. [Accessed 4 01 2014].

[41] "Kurento Media API," [Online]. Available: http://www.kurento.org/docs/4.2.0/kmf-media-api/packages.html. [Accessed 22 04 2014].

[42] "MSRP Library," [Online]. Available: https://msrp.java.net/. [Accessed 8 03 2014].

[43] "MSRP RA," [Online]. Available: https://code.google.com/p/jain-slee/source/checkout?repo=msrp. [Accessed 12 03 2014].

[44] "JUnit," [Online]. Available: http://junit.org/. [Accessed 03 12 2014].

[45] "SIPp installation and using," [Online]. Available: http://sipp.sourceforge.net/doc/reference.html.

[46] "Mobicents MSRP Example," [Online]. Available: https://code.google.com/p/jain-slee/source/browse/?repo=msrp#git%2Fexamples%2Fchat-server%2Fevents%253Fstate%253Dclosed.

[47] "SIPp Examples," [Online]. Available: http://tomeko.net/other/sipp/sipp_cheatsheet.php?lang=pl.

[48] "A brief History of VoIP," [Online]. Available: http://www.joehallock.com/edu/pdfs/Hallock_J_VoIP_Past.pdf.

[49] R. M. Perea, Internet Multimedia Communications Using SIP, Morgan Kaufmann Series, 2008.

[50] A. B. Johnson, SIP Understanding the Session Initiation Protocol, Artech House.

[51] "RFC SIP," [Online]. Available: http://www.ietf.org/rfc/rfc3261.txt.

[52] "Pudding Image," [Online]. Available: http://www.nytimes.com/2007/09/24/business/media/24adcol.html?ex=1348286400&en=2b872e9e7df0ee8f&ei=5090&partner=rssuserland&emc=rss&_r=0.

[53] "SoftwareQuality," [Online]. Available: http://searchsoftwarequality.techtarget.com/answer/What-are-the-different-software-testing-methodologies. [Accessed 20 05 2014].

[54] "New Instructions Software QoS," [Online]. Available: http://www.newinstruction.com/testing-definitions.pdf. [Accessed 26 05 2014].

[55] "Software Testing Methodologies," [Online]. Available: http://www.buzzle.com/articles/software-testing-methodologies.html. [Accessed 22 05 2014].

[56] "SipP," [Online]. Available: http://sipp.sourceforge.net/. [Accessed 20 11 2013].

[57] "Content Sharing - So hot right now," [Online]. Available: http://www.quirk.biz/cms/1125.howto-partfour.pdf. [Accessed 10 12 2013].

[58] "H.248 Protocol," [Online]. Available: http://www.cisco.com/en/US/docs/voice_ip_comm/pgw/9/feature/module/9.7_3_/h_248.pdf. [Accessed 03 12 2013].

[59] "History of Telecommunications 2," [Online]. Available: http://www.cclab.com/billhist.htm. [Accessed 28 09 2013].

[60] "IMS - Open-Source Projects," [Online]. Available: http://ims.no.sapo.pt/opensource.htm. [Accessed 17 11 2013].

[61] "IMS Alternatives," [Online]. Available: http://www.slideshare.net/Gredmill/session-1-ims-alternatives. [Accessed 12 12 2013].

[62] "IMS and Multimedia Services," [Online]. Available: http://catis-blog.com/. [Accessed 21 12 2013].

[63] "JAIN SIP API," [Online]. Available: http://www.oracle.com/technetwork/articles/entarch/introduction-jain-sip-090386.html. [Accessed 28 12 2013].

[64] "JAIN SIP Tutorial," [Online]. Available: http://www.slideshare.net/rajibdk/jain-sip-tutorial. [Accessed 28 12 2013].

[65] "JAIN SLEE 1.1 specification," [Online]. Available: https://www.jcp.org/en/jsr/detail?id=240. [Accessed 25 11 2013].

[66] "JAIN SLEE Architecture," [Online]. Available: http://conan.diei.unipg.it/WEB-RTM/lucidiRCM/Slee.pdf. [Accessed 10 10 2013].

[67] "Jitsi," [Online]. Available: https://jitsi.org/. [Accessed 13 12 2013].

[68] "List of SIP Responses," [Online]. Available: http://en.wikipedia.org/wiki/List_of_SIP_response_codes. [Accessed 20 09 2013].

[69] "Media Gateway Control Protocol (MGCP) Technology," [Online]. Available: http://web.uct.ac.za/depts/commnetwork/eee5026/note/eee5026-06-620mgcp.pdf. [Accessed 10 12 2013].

[70] "Megaco vs MGCP," [Online]. Available: http://hive1.hive.packetizer.com/users/packetizer/papers/ipmc/MEGACOvsMGCP_v3.pdf. [Accessed 20 12 2013].

[71] "Mobicents B2BUA example," [Online]. Available: http://docs.jboss.org/mobicents/jain-slee/2.7.0.FINAL/examples/sip-b2bua/user-guide/en-US/html/source_code_overview.html. [Accessed 05 01 2014].

[72] "Mobicents Custom Events Example," [Online]. Available: http://docs.jboss.org/mobicents/jain-slee/2.7.0.FINAL/tools/eclipslee/user-guide/en-US/pdf/Mobicents_SLEE_Tools_EclipSLEE_User_Guide.pdf. [Accessed 03 01 2014].

[73] "Mobicents Group," [Online]. Available: https://groups.google.com/forum/#!forum/mobicents-public. [Accessed 03 11 2013].

[74] "Mobicents MSRP Example," [Online]. Available: https://code.google.com/p/jain-slee/source/browse/?repo=msrp#git%2Fexamples%2Fchat-server%2Fevents%253Fstate%253Dclosed. [Accessed 28 10 2013].

[75] "Open Mobile Alliance," [Online]. Available: http://en.wikipedia.org/wiki/Open_Mobile_Alliance. [Accessed 01 11 2013].

[76] "Overview of Voice over IP," [Online]. Available: http://www.coe.montana.edu/ee/rwolff/EE548/papers/VoIP/upenn_Overview_VoIP.pdf. [Accessed 20 09 2013].

[77] "Peer-to-peer SIP," [Online]. Available: http://en.wikipedia.org/wiki/Peer-to-peer_SIP. [Accessed 20 12 2013].

[78] "Project Clearwater," [Online]. Available: http://www.projectclearwater.org/. [Accessed 12 12 2013].

[79] "Relation among Session, Dialog, Transaction & Message," [Online]. Available:

http://www.siptutorial.net/SIP/relation.html. [Accessed 27 09 2013].

[80] "RFC 3261 Examples Explained," [Online]. Available: http://www.in2eps.com/fo-sip/tk-fo-sip-archi.html#siparch-01. [Accessed 08 10 2013].

[81] "RFC Addressing Record-Route Issues in SIP," [Online]. Available: http://tools.ietf.org/search/rfc5658. [Accessed 29 12 2013].

[82] "RFC GRUU," [Online]. Available: http://tools.ietf.org/html/rfc5627. [Accessed 28 11 2013].

[83] "RFC Message Session Relay Protocol," [Online]. Available: http://tools.ietf.org/html/rfc4975. [Accessed 03 10 2013].

[84] "Rich Communication Services," [Online]. Available: http://www.gsma.com/futurecommunications/rcs/. [Accessed 24 09 2013].

[85] "Scrum Image," [Online]. Available: http://epf.eclipse.org/wikis/scrumpt/Scrum/guidances/supportingmaterials/scrum_overview_610E45C2.html. [Accessed 24 12 2013].

[86] "SDP Offer/Answer Mechanism to Enable File Transfer," [Online]. Available: http://tools.ietf.org/html/draft-ietf-mmusic-file-transfer-mech-08. [Accessed 06 09 2013].

[87] "SIP and other VoIP protocols and applications," [Online]. Available: http://openlife.cc/system/files/FSWC+Henrik+Ingo+Article+SIP,+VoIP+and+FLOSS.pdf. [Accessed 27 12 2013].

[88] "Telecom Advertising," [Online]. Available: http://www.telecomadvertising.com/. [Accessed 28 09 2013].

[89] "Telecommunications Evolution and Future," [Online]. Available: http://www.pearsonhighered.com/samplechapter/0130281360.pdf. [Accessed 26 09 2013].

[90] "The Evolution of Communication Networks," [Online]. Available: http://blogs.cisco.com/cle/the-evolution-of-communication-networks/. [Accessed 26 09 2013].

[91] "VoIP Providers," [Online]. Available: http://voip-service-review.toptenreviews.com/. [Accessed 04 10 2013].

[92] "What is GRUU," [Online]. Available: http://blog.greenl.ee/2011/11/16/gruu/. [Accessed 04 12 2013].

[93] "Wit Joyn," [Online]. Available: http://www.wit-software.com/solutions/rcs-rcs-e/. [Accessed 24 09 2013].

[94] "X-Lite," [Online]. Available: http://www.counterpath.com/x-lite.html. [Accessed 04 12 2013].